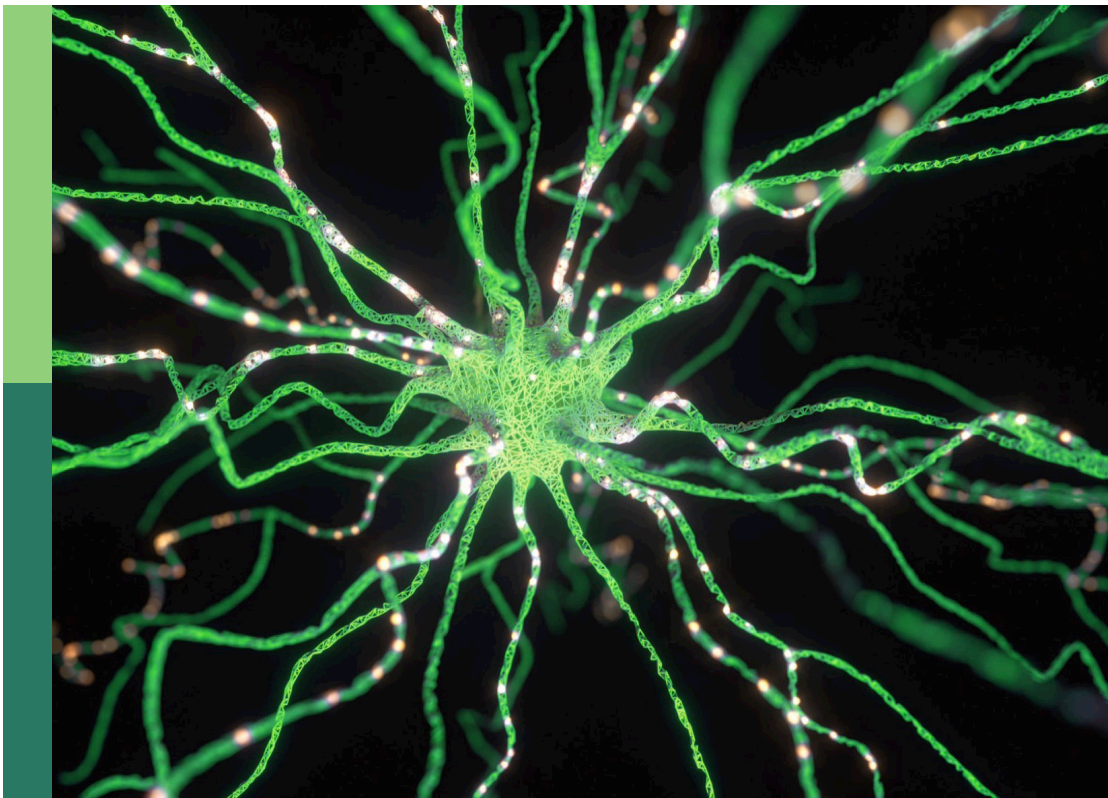
frontiers | Research Topics

Advancing neural network-based intelligent algorithms in robotics: challenges, solutions, and future perspectives

Edited by
Long Jin and Xin Ma

Published in
Frontiers in Neurorobotics



ISSN 1664-8714
ISBN 978-2-8325-6708-1
DOI 10.3389/978-2-8325-6708-1

Generative AI statement
Any alternative text (Alt text) provided alongside figures in the articles in this ebook has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

About Frontiers

Frontiers is more than just an open access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers journal series

The Frontiers journal series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the *Frontiers journal series* operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews. Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the *Frontiers journals series*: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area.

Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers editorial office: frontiersin.org/about/contact

Advancing neural network-based intelligent algorithms in robotics: challenges, solutions, and future perspectives

Topic editors
Long Jin — Lanzhou University, China
Xin Ma — The Chinese University of Hong Kong, China

Citation
Jin, L., Ma, X., eds. (2025). *Advancing neural network-based intelligent algorithms in robotics: challenges, solutions, and future perspectives*. Lausanne: Frontiers Media SA.
doi: 10.3389/978-2-8325-6708-1

Table of contents

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 12 March 2024
DOI 10.3389/fnbot.2024.1382305

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Yizhi Chen,
Royal Institute of Technology, Sweden
Jialiang Fan,
University of Georgia, United States

*CORRESPONDENCE
Wenyu Bian
✉ 36815337@qq.com
Liangxiong Chen
✉ 38393626@qq.com

RECEIVED 05 February 2024
ACCEPTED 26 February 2024
PUBLISHED 12 March 2024

CITATION
Li M, Bian W, Chen L and Liu M (2024) HiDeS: a higher-order-derivative-supervised neural ordinary differential equation for multi-robot systems and opinion dynamics. *Front. Neurorobot.* 18:1382305. doi: 10.3389/fnbot.2024.1382305

HiDeS: a higher-order-derivative-supervised neural ordinary differential equation for multi-robot systems and opinion dynamics

Meng Li[1,2], Wenyu Bian[1]*, Liangxiong Chen[1]* and Mei Liu[1]

[1]Zhangjiajie College, Zhangjiajie, China, [2]School of Public Administration, Hunan University, Changsha, China

This paper addresses the limitations of current neural ordinary differential equations (NODEs) in modeling and predicting complex dynamics by introducing a novel framework called higher-order-derivative-supervised (HiDeS) NODE. This method extends traditional NODE frameworks by incorporating higher-order derivatives and their interactions into the modeling process, thereby enabling the capture of intricate system behaviors. In addition, the HiDeS NODE employs both the state vector and its higher-order derivatives as supervised signals, which is different from conventional NODEs that utilize only the state vector as a supervised signal. This approach is designed to enhance the predicting capability of NODEs. Through extensive experiments in the complex fields of multi-robot systems and opinion dynamics, the HiDeS NODE demonstrates improved modeling and predicting capabilities over existing models. This research not only proposes an expressive and predictive framework for dynamic systems but also marks the first application of NODEs to the fields of multi-robot systems and opinion dynamics, suggesting broad potential for future interdisciplinary work. The code is available at https://github.com/MengLi-Thea/HiDeS-A-Higher-Order-Derivative-Supervised-Neural-Ordinary-Differential-Equation.

KEYWORDS

neural ordinary differential equations, multi-robot systems, opinion dynamics, robotics, neural networks

1 Introduction

As a learnable model parameterized by $\boldsymbol{\theta} \in \mathbb{R}^n$, a standard neural ordinary differential equation (NODE) $\dot{\boldsymbol{x}} = \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$ is particularly adept at representing complex and nonlinear dynamics (Chen et al., 2018; Liufu et al., 2024), where $\boldsymbol{x} \in \mathbb{R}^d$ is the state at time t, $\dot{\boldsymbol{x}} = \mathrm{d}\boldsymbol{x}/\mathrm{d}t$ denotes the time derivative of $\boldsymbol{x}$, and $\boldsymbol{\phi}(\boldsymbol{x}, t)$ is a vector field with $\boldsymbol{\phi} \in (\mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d)$ being a function of $\boldsymbol{x}$ and t. Its strength lies in processing time-variant data and adaptively learning from it. This modeling flexibility renders NODEs great potential for the intricate nature of dynamic systems (Hua et al., 2023; Wang et al., 2023; Jin et al., 2024), enabling a more nuanced understanding of complex dynamic systems.

Despite these strengths, the standard NODE encounters expressivity limitations, failing to model functions like NOT operations (Kidger, 2021; Xu et al., 2023). The NOT operation [i.e., $(0,1) \rightarrow (1,0)$] involves trajectories that necessarily intersect, presenting a challenge for standard NODEs that cannot model intersecting trajectories due to their first-order nature. The NODE with momentum, which can be regarded as a second-order ODE, improves the expressive capability (Sander et al., 2021): $\ddot{\boldsymbol{x}} = c_0\dot{\boldsymbol{x}} + c_1\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$, where c_0 and c_1 are constants. Nonetheless, it can only express limited dynamics due to the linear relationship of $\dot{\boldsymbol{x}}$ and $\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$. Besides, it cannot model interactions between $\dot{\boldsymbol{x}}$ and $\boldsymbol{x}$. The second-order NODE (SONODE) presented in Norcliffe et al. (2020) seeks to address this limitation by modeling the interactions between $\dot{\boldsymbol{x}}$ and $\boldsymbol{x}$. However, SONODE cannot model interactions between higher-order derivatives and $\boldsymbol{x}$, and the supervised signal used in training is only the ground-truth value of $\boldsymbol{x}$, which confines its scope and limits its prediction capability.

To surmount these challenges, we propose a higher-order-derivative-supervised NODE (HiDeS NODE) that is able to model interactions between higher-order derivatives and $\boldsymbol{x}$. This approach not only expands the expressive range of NODEs but also enhances predictive ability through employing the state vector and its higher-order derivatives as supervised signals, surpassing the modeling and predicting performance of existing NODEs.

This paper evaluates the effectiveness of the HiDeS NODE in the realms of multi-robot systems and opinion dynamics, key areas of dynamic systems, both domains that inherently involve complex interactions and communication (Granha et al., 2022). In multi-robot systems, conventional analytic solutions fall short in high-dimensional control tasks (Károly et al., 2021), such as multi-robot grasping and motion control. NODEs, in contrast, offer a promising avenue for modeling and controlling complex dynamic interactions in a continuous, efficient, and adaptable manner in multi-robot systems. Regarding opinion dynamics research, the primary objective is to decipher the underlying mechanisms and influences that catalyze shifts in opinions. Existing methodologies for learning opinion dynamics overlook the critical prior knowledge that opinion dynamics can be described as an ODE formulated as $\dot{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x}, t)$. ODEs are particularly well-suited for modeling the fluid nature of opinion dynamics due to their inherent capacity to capture the dynamics of evolving systems. However, contemporary models employed in learning opinion dynamics underutilize this foundational knowledge. This oversight hampers their ability to effectively capture the nuanced and intricate nature of opinion evolution. Furthermore, the complexities inherent in the evolution of opinions present considerable challenges to the application of existing NODEs in both modeling and forecasting the trajectories of opinion dynamics. The HiDeS NODE conquers these aspects, providing a more effective tool for understanding and predicting opinion evolution.

To bridge these gaps, we propose a new NODE, termed HiDeS NODEs, for modeling and predicting tasks in multi-robot control and opinion dynamics. Figure 1 illustrates the framework of the HiDeS NODE, and Table 1 qualitatively demonstrates the HiDeS NODE's superiority compared with existing NODEs.

The contributions of this paper are demonstrated as follows:

- We propose the HiDeS NODE, a novel approach for modeling the intricacies of dynamics. The HiDeS NODE excels in modeling and predicting interactions among higher-order derivatives within dynamic systems. This advancement provides a more accurate and nuanced representation of dynamic systems.
- The HiDeS NODE integrates higher-order derivatives as supervised signals, significantly enhancing the ability to predict dynamical behaviors.
- We examine the versatility and effectiveness of the proposed HiDeS NODE through its application in two distinct yet complex fields: Multi-robot systems and opinion dynamics. In these fields, the model's ability to capture and predict intricate system dynamics is evaluated.
- To our knowledge, this is the first time that the NODE is introduced for opinion dynamics and multi-robot-system control. Application of the proposed HiDeS NODE to these fields unveils new avenues for both the advancement of NODE methodologies and the nuanced modeling of opinion dynamics and multi-robot-system control.

2 Related work

In this section, we briefly review three lines of research that are close to our work: NODEs, multi-robot-system control methods, and opinion dynamics modeling.

2.1 NODEs

The intersection of neural networks and differential equations, especially interpreting residual networks (ResNets) as discretized ODEs, spurs the development of NODEs (Weinan, 2017; Cui et al., 2023; Ruiz-Balet and Zuazua, 2023). NODEs integrate black-box ODE solvers and neural networks to parameterize the hidden state's derivative. This integration substantially advances time-series modeling, offering robust function approximation and handling of irregular data (Chen et al., 2018; Kidger, 2021). However, standard NODEs encounter representational constraints without dimensionality augmentation, constraining their universal approximation capabilities for certain functions (Dupont et al., 2019).

Research pivots toward higher-dimensional NODEs to overcome these limitations. Momentum-enhanced ResNets, representing second-order NODE extensions, exhibit enhanced capability in modeling non-homeomorphic dynamics and demonstrate improved convergence properties (Sander et al., 2021). In parallel, augmented NODEs, by expanding the solution space, facilitate the learning of more complex functions through simpler dynamic flows, thereby sidestepping the limitations of the vector field's general-representation property (Kidger, 2021). Nonetheless, augmented NODEs introduce challenges in interpretability and alter the loss landscape's structure (Norcliffe et al., 2020). A specific iteration of augmented NODEs, termed second-order NODEs (SONODEs) (Norcliffe et al., 2020), captures more intricate behaviors by integrating second-order dynamics, effectively combining the principles of coupled augmented NODEs.

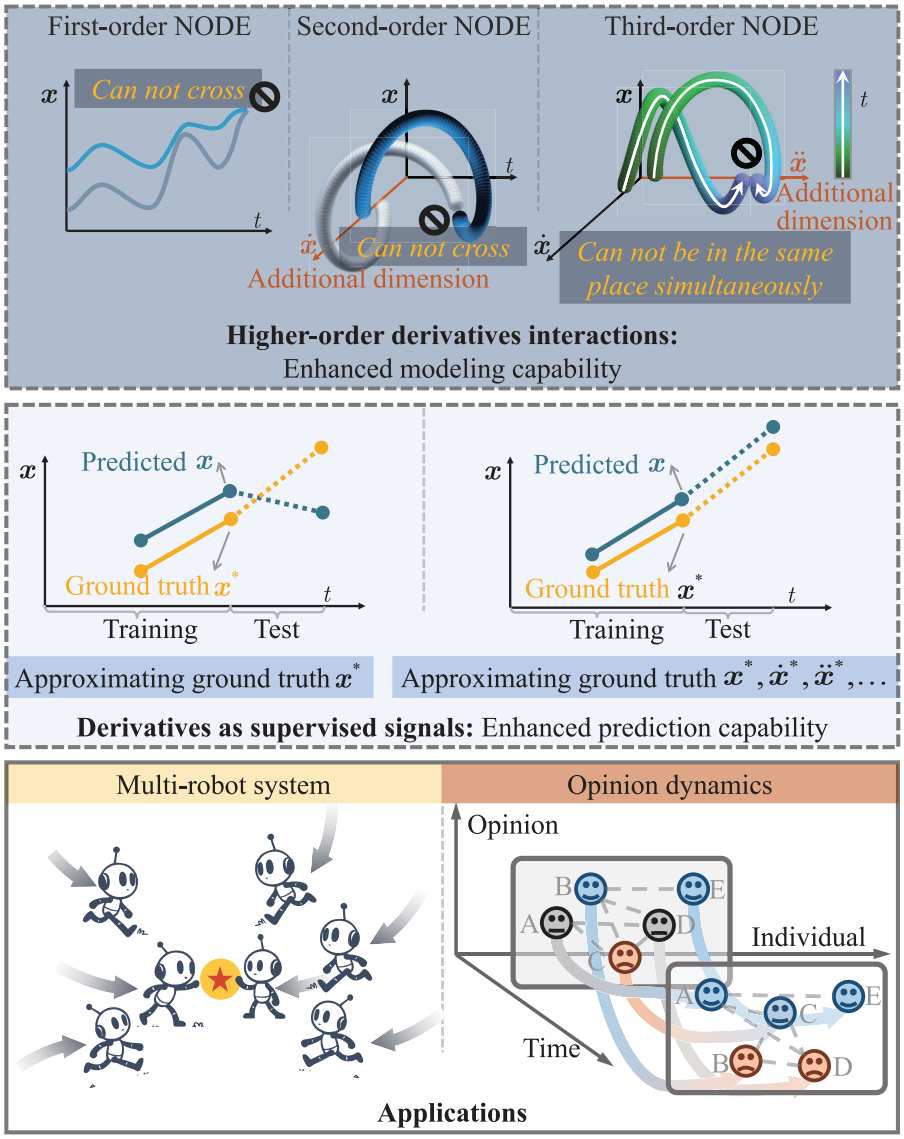


FIGURE 1
Framework of this paper. **(Top)** Evolution from first to third-order NODEs, highlighting their progressively sophisticated ability to model complex trajectories where higher orders allow for more intricate behaviors. **(Middle)** Predicting precision with and without using higher-order derivatives as supervised signals, showing the latter's superior approximation of ground truth. **(Bottom)** Practical applications of HiDeS NODEs in multi-robot systems and opinion dynamics.

Additionally, the advent of heavy ball NODEs (HBNODEs) (Xia et al., 2021) marks a significant advancement. HBNODEs incorporate the classical momentum accelerated gradient descent method and adeptly mitigate the vanishing gradient problem, thereby enhancing the model's capacity in learning long-term dependencies in sequential data (Xia et al., 2021).

2.2 Multi-robot-system control

Multi-robot systems provide significant benefits in tasks that demand the duplication of effort, risk reduction, or adaptability, offering distinct advantages over single-robot systems (Hichri et al., 2022; Kwa et al., 2022). Multi-robot-system control methods can be categorized into deterministic methods with fixed forms and learning-based methods (Pierpaoli et al., 2021). However, deterministic methods lack flexibility and adaptability in dynamic or unpredictable environments (Liu et al., 2023). In order to overcome these defects, learning methods are increasingly applied to multi-robot control problems. Adaptation methods, for instance, are proposed to enhance trajectory prediction efficiency in multi-agent systems (Aydemir et al., 2023). Furthermore, the parameter-adaptive learning methods are improved through iterative parametric learning controllers (Yu and Chen, 2023). Additionally, neural network-based adaptive learning methods are utilized to learn unknown fault functions, ensuring cooperative tracking in distributed multi-robot systems (Khalili et al., 2020). Despite these advancements, existing methods often fall short in

TABLE 1 Comparisons among different NODEs.

Models	Year	Modeling higher-order dynamics	Modeling derivative interactions of each order	Applied to opinion dynamics	Supervised signals
Standard NODE (Chen et al., 2018)	2018	×	×	×	$\boldsymbol{x}(t)$
Momentum NODE (Sander et al., 2021)	2021	Only 2nd order	×	×	$\boldsymbol{x}(t)$
SONODE (Norcliffe et al., 2020)	2020	Only 2nd order	√	×	$\boldsymbol{x}(t)$
HiDeS NODEs	2024	√	√	√	$\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}$

naturally and efficiently modeling the dynamics that often can be described as an ODE, a gap that NODEs can potentially fill.

2.3 Opinion dynamics modeling

Opinion dynamics studies how opinions form and evolve over time through interactions with individuals and environments. Researchers propose various mathematical models to understand and predict the dynamics of opinions. These include continuous-time models such as the DeGroot model (Wu et al., 2023), Hegselmann-Krause model, and bounded confidence model (Kolarijani et al., 2021), as well as discrete-time models like the Ising model, Voter model, and Friedkin and Johnsen model (Baumann et al., 2020; Ao and Jia, 2023; Peng et al., 2023). However, these models, with their fixed forms, lack the flexibility to model the evolution of opinions independently.

In response to these limitations, researchers leverage advances in neural networks to utilize their nonlinear relation approximation ability for learning complex opinion dynamics. An early approach introduces a linear influence model that learns edge influence strength from real data (De et al., 2014). Unlike traditional models, this linear model represents a foundational step in opinion dynamics learning methods, but its simplicity fails to capture the complexity of societal opinion dynamics. Furthering this exploration, SLANT (De et al., 2016; Zhu et al., 2020) introduces a linear model of latent opinions driven by stochastic differential equations (SDEs) using historical, fine-grained event data. Subsequently, SLANT+ (Kulkarni et al., 2017) extends this model with a nonlinear generative model and a network-guided recurrent neural network (RNN) architecture. This model underscores the importance of nonlinearity in designing opinion dynamics models. However, the RNN architecture it relies on faces the challenge of the vanishing gradient problem, hindering long-term predictions of opinion flow. Learnable opinion dynamics model (LODM) (Monti et al., 2020) emerges as a learnable generalization of an opinion dynamics model, combining the causal interpretability of traditional agent-based models with data-driven approaches. Additionally,Okawa and Iwata (2022) introduces the sociologically-informed neural network (SINN), a novel hybrid approach that integrates sociological and social psychological theories with data-driven neural networks to model and predict opinion dynamics in social networks. Despite these advances, current models do not fully exploit the prior knowledge of differential equations in opinion evolution, nor do they effectively model higher-order derivatives.

TABLE 2 Main symbols and notations.

Symbol	Description
$\boldsymbol{x}$	State vector representing opinions of individuals
t	Time variable
$\dot{\boldsymbol{x}}$	First-order time derivatives of $\boldsymbol{x}$
$\ddot{\boldsymbol{x}}$	Second-order time derivatives of $\boldsymbol{x}$
$\dddot{\boldsymbol{x}}$	Third-order time derivatives of $\boldsymbol{x}$
c	Order of the highest derivative in HiDeS NODE
$\boldsymbol{x}^{(c)}$	c-th order time derivative of $\boldsymbol{x}$
$\boldsymbol{\theta}$	Parameters of a neural network
$\boldsymbol{\phi}_{\boldsymbol{\theta}}(\cdot)$	Vector field (neural network) parameterized by $\boldsymbol{\theta}$
$\mathbb{R}^d$	d-dimensional real space
$\mathbb{R}^n$	n-dimensional real space
$\boldsymbol{\omega}$	Extended state vector in HiDeS NODE
Δt	Time step for numerical approximation
$\boldsymbol{\vartheta}$	Alternate set of parameters for the neural network
$\check{t}, \hat{t}$	Time interval boundaries

3 Materials and methods

In this section, formal descriptions and analyses of the proposed HiDeS NODE are provided. Table 2 presents the main symbols and notations used throughout this paper to ensure clarity and ease of understanding.

3.1 Formulation of the HiDeS NODE

The HiDeS NODE has two unique features for modeling and predicting opinion evolution. The first is that the HiDeS NODE is a higher-order NODE that is able to model interactions of higher-order derivatives of the opinion variable $\boldsymbol{x}$. The second is that the HiDeS NODE adopts higher-order derivatives as supervisory

signals to predict opinion evolution better. The HiDeS NODE is described as Equation (1):

$$\begin{bmatrix} \dot{\boldsymbol{x}}(t) \\ \ddot{\boldsymbol{x}}(t) \\ \dddot{\boldsymbol{x}}(t) \\ \vdots \\ \boldsymbol{x}^{(c)}(t) \end{bmatrix} = \boldsymbol{\phi}_{\boldsymbol{\theta}} \left(\begin{bmatrix} \boldsymbol{x}(t) \\ \dot{\boldsymbol{x}}(t) \\ \ddot{\boldsymbol{x}}(t) \\ \vdots \\ \boldsymbol{x}^{(c-1)}(t) \end{bmatrix}, t \right), \tag{1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^d$ is a time-varying vector representing the opinion of d individuals; $t \in [\check{t}, \hat{t}]$ is the time; Vectors $\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t)$, and $\boldsymbol{x}^{(c)}(t)$ correspond to the first, second, third, and c-th order time derivatives of $\boldsymbol{x}$, respectively; The function $\boldsymbol{\phi} : \mathbb{R}^{cd} \times \mathbb{R} \to \mathbb{R}^{cd}$ is parameterized by a neural network with the parameter $\boldsymbol{\theta} \in \mathbb{R}^n$. Note that $[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c)}(t)]^\top \in \mathbb{R}^{cd}$ is a concatenation of higher-order derivatives, where the superscript $^\top$ means a transpose of a vector, and we call a HiDeS NODE with up to c-th order time derivatives in this concatenation as the HiDeS-c NODE. To enhance readability and avoid redundancy, we may omit "(t)" in certain contexts where the time dependency is understood and does not affect the meaning or clarity of the mathematical expressions.

Remark 1. One advantage of a HiDeS NODE is that it is able to model nonlinear interactions between higher-order derivatives and $\boldsymbol{x}$. In practice, multiple higher-order derivatives and $\boldsymbol{x}$ can interact with each other. For example, there can be terms like $\dot{\boldsymbol{x}} \otimes \ddot{\boldsymbol{x}}$ in the vector field, where $\otimes$ is the Hadamard product.

It can be seen that the standard NODE (Chen et al., 2018) is a HiDeS-1 NODE, and if we just focus on the formulation, SONODE (Norcliffe et al., 2020) can be regarded as a HiDeS-2 NODE. In fact, the HiDeS-2 NODE distinguishes itself from SONODE due to its unique training process.

3.2 Training of the HiDeS NODE

Existing variants of the NODE utilize the ground-truth value of $\boldsymbol{x}(t)$ as the label for training. Differently, the HiDeS NODE adopts the entire $[\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top$ as the label (the model's prediction $[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c)}(t)]^\top$ is integrated first to get $[\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top$). This approach is beneficial for predicting the future evolution of $\boldsymbol{x}(t)$. Training the entire $[\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top$ gives accurate approximations of all these variables. Since the prediction of the next time step for the HiDeS NODE relies on the entire $[\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top$, the training strategy of the HiDeS NODE leads to a better prediction performance compared to only training with the ground-truth value of $\boldsymbol{x}(t)$. When the model is predicting the next $\boldsymbol{x}(t)$, utilizing only the ground-truth value of $\boldsymbol{x}(t)$ as the label may lead to an inaccurate result because the basic information it relies on is inaccurate. The inclusion of the derivatives ensures that the model is sensitive to not just the position or condition at a given time but also to the trends and patterns of change, which are critical for forecasting. An explanation is illustrated in Figure 1.

3.3 Inexpressible trajectories of the HiDeS NODE

The superior expressive capability of the HiDeS NODE comes from two aspects.

The first is that lower-order NODEs have limitations in modeling trajectories that require the representation of higher-order dynamics. Consider a trajectory that requires an abrupt change in its acceleration (second derivative of $\boldsymbol{x}$), which is not expressible in a first-order system but can be expressed in a second-order system $[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t)]^\top = \boldsymbol{\phi}_{\boldsymbol{\theta}}([\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t)]^\top, t)$. Similarly, trajectories requiring changes in the third derivative (jerk) are not expressible in a second-order system but can be captured in a third-order system, and so on. As a result, there exist trajectories that can not be expressed by $[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c)}(t)]^\top = \boldsymbol{\phi}_{\boldsymbol{\theta}}\left([\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top, t\right)$ but can be expressed by $[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c+1)}(t)]^\top = \boldsymbol{\phi}_{\boldsymbol{\theta}}\left([\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c)}(t)]^\top, t\right)$.

The second origin of the superior expressive capability of the HiDeS NODE is that it alleviates the restriction that trajectories cannot cross. One major limitation of the standard NODE is that trajectories under different initial conditions cannot intersect, which constrains its expressive capability. In the following, we show how this constraint is able to be eliminated by the HiDeS NODE.

Theorem 1 (Inexpressible trajectories of a HiDeS-c NODE). Assume that the function $\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{\omega}, t) : \mathbb{R}^{cd} \times \mathbb{R} \to \mathbb{R}^{cd}$ with $t \in [\check{t}, \hat{t}]$ is Lipschitz continuous w.r.t. $\boldsymbol{\omega} \in \mathbb{R}^{cd}$. Consider a HiDeS-$c$ NODE governed by Equation (2):

$$\begin{aligned} &[\dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c)}(t)]^\top \\ &\qquad = \boldsymbol{\phi}_{\boldsymbol{\theta}}\left([\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top, t\right), \end{aligned} \tag{2}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^d$ is the state vector, $\boldsymbol{\phi}_{\boldsymbol{\theta}} : \mathbb{R}^{cd} \times \mathbb{R} \to \mathbb{R}^{cd}$ is a continuously differentiable function parameterized by neural network parameters $\boldsymbol{\theta}$. For any two initial conditions $\boldsymbol{\omega}(\check{t}) = [\boldsymbol{x}(\check{t}), \dot{\boldsymbol{x}}(\check{t}), \ldots, \boldsymbol{x}^{(c-1)}(\check{t})]^\top$ and $\tilde{\boldsymbol{\omega}}(\check{t}) = [\tilde{\boldsymbol{x}}(\check{t}), \dot{\tilde{\boldsymbol{x}}}(\check{t}), \ldots, \tilde{\boldsymbol{x}}^{(c-1)}(\check{t})]^\top$, trajectories that require $\boldsymbol{\omega}(t)$ and $\tilde{\boldsymbol{\omega}}(t)$ to cross over the interval $[\check{t}, \hat{t}]$ are inexpressible by a HiDeS-c NODE.

Proof. Define the extended state vector $\boldsymbol{\omega} \in \mathbb{R}^{cd}$ as $\boldsymbol{\omega}(t) = [\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)]^\top$. The HiDeS-$c$ NODE can be represented as Equation (3):

$$\dot{\boldsymbol{\omega}} = \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{\omega}(t), t). \tag{3}$$

Given that $\boldsymbol{\phi}_{\boldsymbol{\theta}}$ is Lipschitz continuous w.r.t. $\boldsymbol{\omega}$, the Picard-Lindelöf theorem (Anil Kumar et al., 2022; Zhang et al., 2023) assures the existence of a unique solution $\boldsymbol{\omega}(t)$ for a given initial condition $\boldsymbol{\omega}(0)$. This uniqueness implies that for any two distinct initial conditions $\boldsymbol{\omega}(\check{t})$ and $\tilde{\boldsymbol{\omega}}(\check{t})$, the resulting trajectories $\boldsymbol{\omega}(t)$ and $\tilde{\boldsymbol{\omega}}(t)$ do not cross over $[\check{t}, \hat{t}]$. As a result, trajectories that require $\boldsymbol{\omega}(t)$ and $\tilde{\boldsymbol{\omega}}(t)$ to cross over the interval $[\check{t}, \hat{t}]$ are inexpressible by a HiDeS-c NODE. The proof is thus completed. □

Remark 2. From Theorem 1, it can be seen that as the order c increases, the degree of freedom for avoiding the crossing of $\boldsymbol{x}(t)$ and $\tilde{\boldsymbol{x}}(t)$ increases. In practice, there are some trajectories cross

in the x-t space or in the phase space [i.e., x-$\dot{x}$-$\ddot{x}$-...-$x^{(c-1)}$-t space], so the HiDeS-c NODE provides a better capability to model these dynamics compared with the standard NODE (a HiDeS-1 NODE) (Chen et al., 2018) and other second-order NODEs (HiDeS-2 NODEs) (Norcliffe et al., 2020; Sander et al., 2021). An intuitive understanding is that additional dimensions provide new directions to make trajectories elude each other, which is illustrated in Figure 1. Consider two actual evolution trajectories under different initial conditions, $x^*(t)$ and $\tilde{x}^*(t)$, which may intersect at some time points if there are no restrictions. However, for $c = 1$, the trajectories $x(t)$ and $\tilde{x}(t)$ generated by the standard NODE cannot intersect due to its nature of first-order ODEs. This limitation means they cannot accurately approximate $x^*(t)$ and $\tilde{x}^*(t)$ in cases where the actual trajectories intersect. Theorem 1 from our manuscript implies similar limitations for higher-order NODEs, but with increasing order c, the trajectories have more freedom, reducing the limitations.

3.4 The HiDeS NODE's utilization of historical information

Due to the introduction of higher-order derivatives, the HiDeS NODE implicitly uses historical state information for predicting the next state. The reason is that higher-order derivatives can be approximated by historical states. For example, the first derivative $\dot{x}$ at the k-th moment t_k can be approximated as $\dot{x}(t_k) \approx (x(t_k) - x(t_{k-1}))/\Delta t$. Similarly, the second derivative $\ddot{x}$ can be approximated as Equation (4):

$$\ddot{x}(t_k) \approx \frac{\left(\frac{x(t_k)-x(t_{k-1})}{\Delta t}\right) - \left(\frac{x(t_{k-1})-x(t_{k-2})}{\Delta t}\right)}{\Delta t}. \tag{4}$$

Iteratively, we have Equation (5):

$$x^{(c)}(t_k) \approx \sum_{i=0}^{c} (-1)^i \binom{c}{i} x(t_{k-i})/(\Delta t^c), \tag{5}$$

where the binomial coefficient $\binom{c}{i} = c!/(i!(c-i)!)$ represents the combinatorial number of ways to choose i elements from a set of c elements. Consequently, $\boldsymbol{\phi}_{\boldsymbol{\theta}}\left([x(t), \dot{x}(t), \ddot{x}(t), \ldots, x^{(c-1)}(t)]^\top, t\right)$ in the HiDeS NODE (Equation 1) can be approximated as a function of historical states as in Equation (6):

$$\begin{aligned}\left[\dot{x}(t), \ddot{x}(t), \dddot{x}(t), \ldots, x^{(c)}(t)\right]^\top &= \boldsymbol{\phi}_{\boldsymbol{\theta}}\left(\left[x(t), \dot{x}(t), \ddot{x}(t), \ldots, x^{(c-1)}(t)\right]^\top, t\right) \\ &\approx \boldsymbol{\phi}_{\boldsymbol{\vartheta}}\left(x(t_k), x(t_{k-1}), \ldots, x(t_{k-c}), t_k\right),\end{aligned} \tag{6}$$

where $\boldsymbol{\phi}_{\boldsymbol{\vartheta}} : \mathbb{R}^d \times \mathbb{R}^d \times \cdots \times \mathbb{R} \to \mathbb{R}^{cd}$ is a function parameterized by $\boldsymbol{\vartheta}$. The HiDeS NODE's utilization of historical information could enhance the prediction of the next state.

3.5 Implementation

We provide Algorithm 1 to show the process of constructing, training, and using a HiDeS NODE. Besides, structures of HiDeS-3 NODEs for multi-robot systems and for opinion dynamics are shown in Figures 2A, B, respectively. In Figure 2, the inputs to the system are a concatenation of the initial state $x(0)$, the initial velocity $\dot{x}(0)$, and the initial acceleration $\ddot{x}(0)$, which are fed into an ODE solver alongside the parametrized function $\boldsymbol{\phi}_{\boldsymbol{\theta}}([x(t), \dot{x}(t), \ddot{x}(t)]^\top)$ to compute the state $x(t)$, velocity $\dot{x}(t)$, and acceleration $\ddot{x}(t)$ at time t.

Input: Time series data, initial conditions $x(0)$, $\dot{x}(0), \ddot{x}(0), \ldots, x^{(c-1)}(0)$
Output: Predicted states $x(t)$, $\dot{x}(t), \ddot{x}(t), \ldots, x^{(c)}(t)$
Initialize neural network parameters $\boldsymbol{\theta}$
procedure CONSTRUCTING A PARAMETERIZED VECTOR FIELD
  Construct a neural network $\boldsymbol{\phi}_{\boldsymbol{\theta}}([x(t), \dot{x}(t), \ddot{x}(t), \ldots, x^{(c-1)}(t)]^\top, t)$ to represent the parameterized vector field
end procedure
procedure TRAINING
  for each epoch **do**
    for each batch **do**
      Apply an ODE solver to get predicted trajectories $[x(t), \dot{x}(t), \ddot{x}(t), \ldots, x^{(c-1)}(t)]^\top$ at various time t based on the parameterized vector field and initial conditions
      Compute the loss between the predicted trajectories and actual trajectories
      Backpropagate loss and update parameters $\boldsymbol{\theta}$
    end for
  end for
end procedure
procedure PREDICTION
  Set initial conditions $x(0)$, $\dot{x}(0), \ddot{x}(0), \ldots, x^{(c-1)}(0)$
  Use the trained $\boldsymbol{\phi}_{\boldsymbol{\theta}}$ to compute future states and derivatives
  return predicted $x(t)$
end procedure

Algorithm 1. Algorithm of HiDeS NODE.

4 Results

In this section, we conduct experiments to evaluate the effectiveness of our models, HiDeS-2 NODE and HiDeS-3 NODE, by comparing them with baseline models [standard NODE (Chen et al., 2018) and SONODE (Norcliffe et al., 2020)] on two applications: multi-robot control and opinion dynamics. These baseline models have the same configurations in terms of network architecture, optimizer, epochs, and learning rate, ensuring a fair comparison. Notably, our models utilize higher-order derivatives as supervised signals, crucial for accurately capturing the intricate, nonlinear evolution of opinions over time. The implementation details of our experiments are as follows.

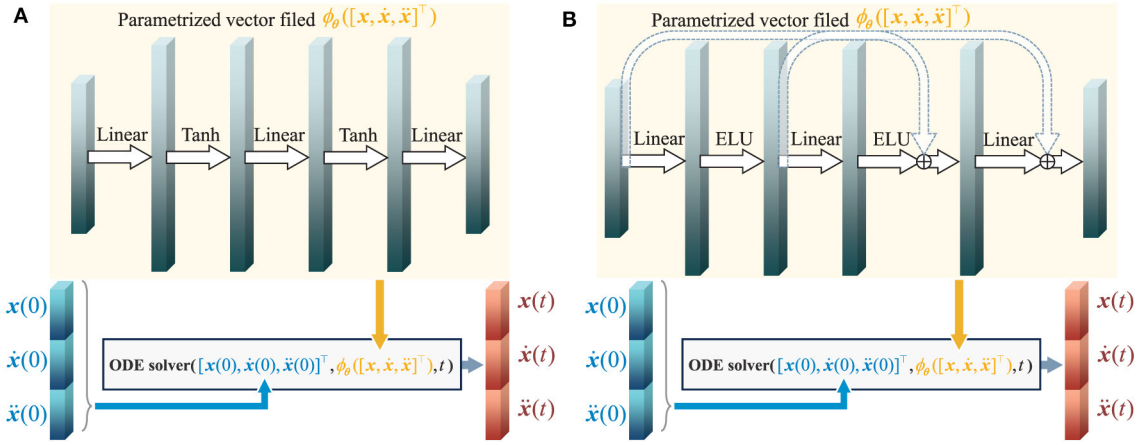


FIGURE 2
Structure of HiDeS-3 NODEs. **(A)** Structure of HiDeS-3 NODE for multi-robot systems. **(B)** Structure of HiDeS-3 NODE for opinion dynamics.

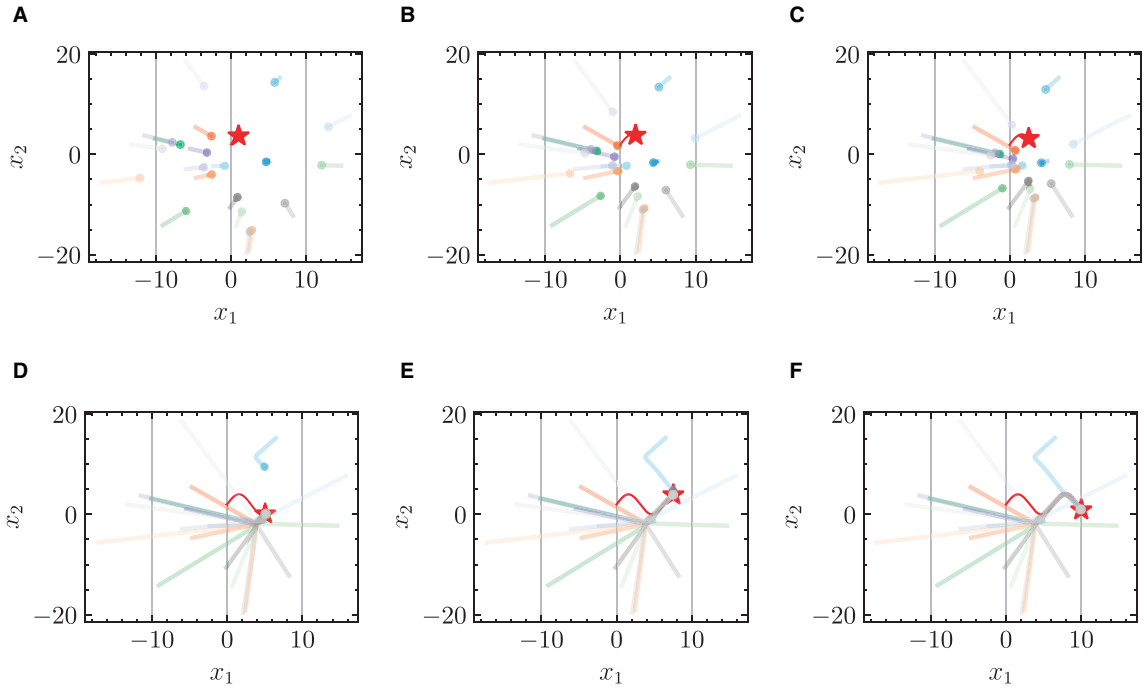


FIGURE 3
Trajectories of multiple robots using standard NODE in target-chasing task without energy constraint. This figure depicts the paths of multiple robots (indicated by dots) as they follow a dynamically moving target (denoted by stars) after 1,000 epochs of training. **(A)** $t = 2$ s. **(B)** $t = 4$ s. **(C)** $t = 5$ s. **(D)** $t = 10$ s. **(E)** $t = 15$ s. **(F)** $t = 20$ s.

4.1 Experimental settings

4.1.1 Settings for multi-robot-system control

In multi-robot-system control, each NODE block is composed of three fully connected layers, each succeeded by a Tanh activation function, as shown in Figure 2A. The NODE block undergoes forward propagation 200 times, evolving from $t = 0$ to $t = 20$, in order to develop a deep model. A weighted loss ℓ that emphasizes the trajectory's later stages is applied: $\ell_W = \sum_{k=0}^{\hat{k}} (k/\hat{k})^p \ell(\boldsymbol{\omega}(t_k), \boldsymbol{\omega}^*(t_k))$, where $\hat{k}$ is the total number of steps, $p > 0$ is a scalar, $\ell(\cdot,\cdot)$ is a loss function, and $\boldsymbol{\omega}^*(t_k)$ is the ground truth of $\boldsymbol{\omega}(t_k)$. In simulations, p is taken as 4. All models are trained for 1,000 epochs using the Adam optimizer and a cosine annealing scheduler with a base learning rate of 0.01.

4.1.2 Settings for opinion dynamics

In simulations of opinion dynamics, each block of NODEs consists of three fully connected layers, each followed by an exponential linear unit (ELU) activation function, as shown in

10.3389/fnbot.2024.1382305

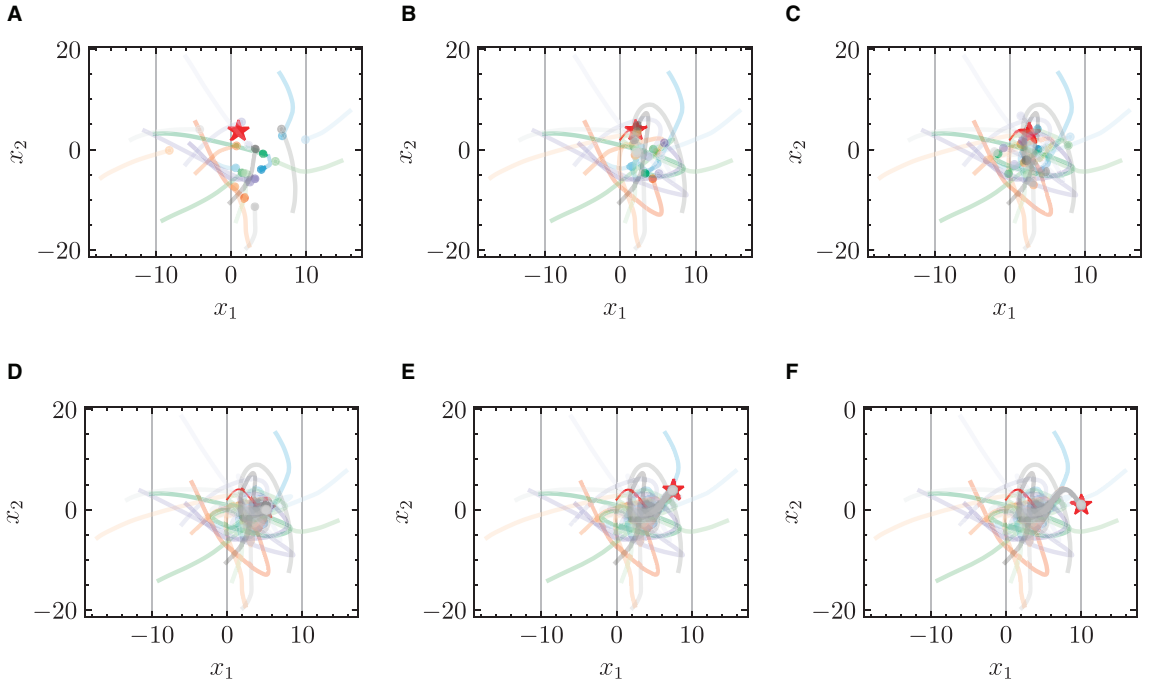


FIGURE 4
Trajectories of multiple robots using HiDeS-3 NODE (ours) in target-chasing task without energy constraint. This figure depicts the paths of multiple robots (indicated by dots) as they follow a dynamically moving target (denoted by stars) after 1,000 epochs of training. **(A)** $t = 2$ s. **(B)** $t = 4$ s. **(C)** $t = 5$ s. **(D)** $t = 10$ s. **(E)** $t = 15$ s. **(F)** $t = 20$ s.

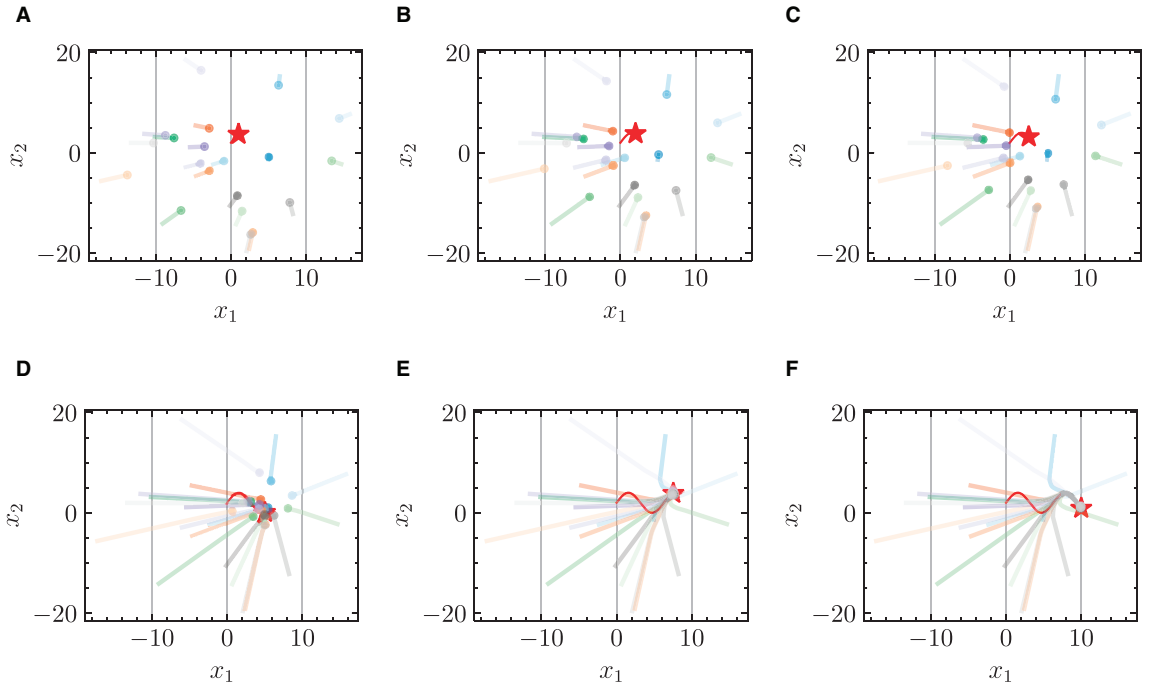


FIGURE 5
Trajectories of multiple robots using standard NODE in target-chasing task with energy constraint. This figure depicts the paths of multiple robots (indicated by dots) as they follow a dynamically moving target (denoted by stars) after 1,000 epochs of training. **(A)** $t = 2$ s. **(B)** $t = 4$ s. **(C)** $t = 5$ s. **(D)** $t = 10$ s. **(E)** $t = 15$ s. **(F)** $t = 20$ s.

10.3389/fnbot.2024.1382305

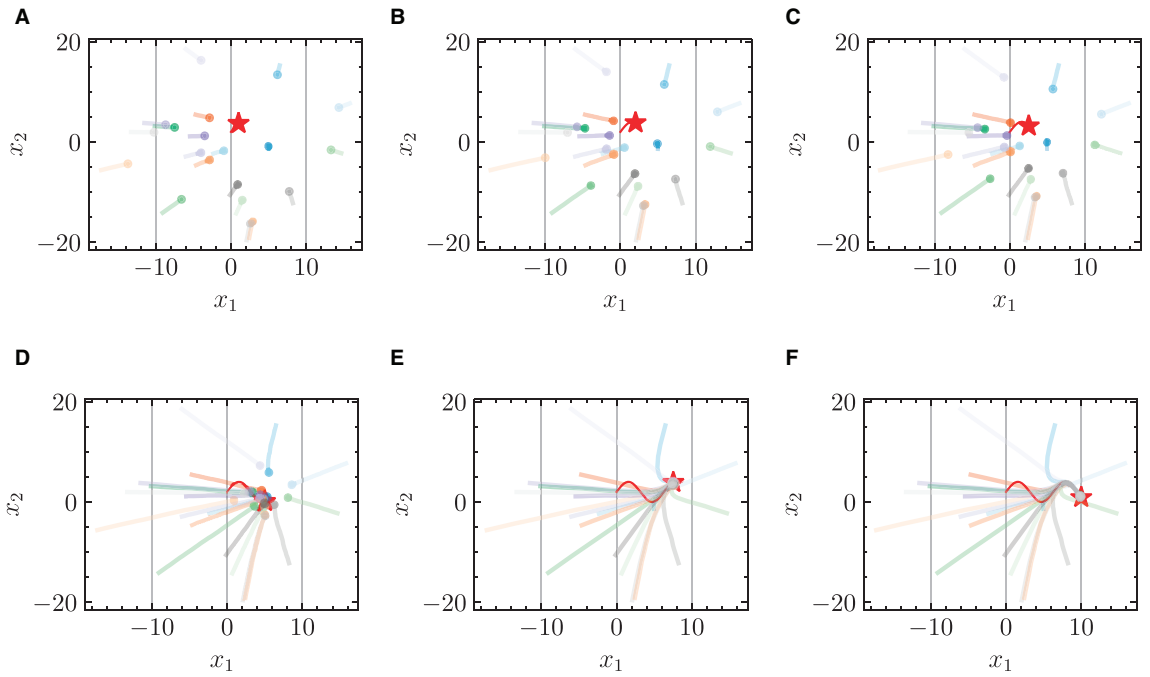


FIGURE 6
Trajectories of multiple robots using HiDeS-3 NODE (ours) in target-chasing task with energy constraint. This figure depicts the paths of multiple robots (indicated by dots) as they follow a dynamically moving target (denoted by stars) after 1,000 epochs of training. **(A)** $t = 2$ s. **(B)** $t = 4$ s. **(C)** $t = 5$ s. **(D)** $t = 10$ s. **(E)** $t = 15$ s. **(F)** $t = 20$ s.

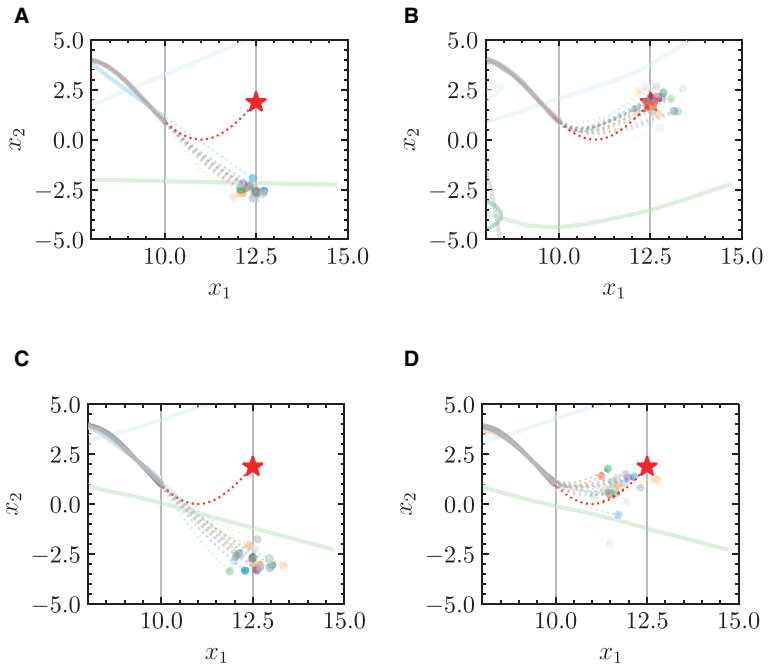


FIGURE 7
Trajectories of multiple robots in target-chasing task after losing information of target. This figure depicts the paths of multiple robots (indicated by dots) as they follow a dynamically moving target (denoted by stars). Solid curves denote training phase (trajectory of target is known), and dashed curves denote test phase (trajectory of target is unknown). **(A)** Standard NODE; without energy constraint. **(B)** HiDeS-3 NODE (ours); without energy constraint. **(C)** Standard NODE; with energy constraint. **(D)** HiDeS-3 NODE (ours); with energy constraint.

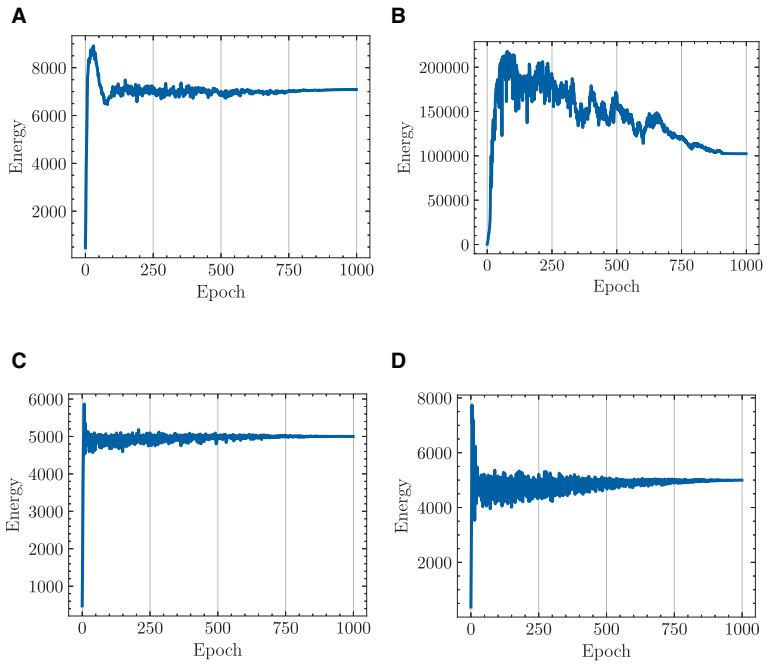


FIGURE 8
Total energy cost of multiple robots in target-chasing task during 1,000 epochs of training. Energy constraint is set such that total energy cost ≤5,000 J. **(A)** standard NODE; without energy constraint. **(B)** HiDeS-3 NODE (ours); without energy constraint. **(C)** Standard NODE; with energy constraint. **(D)** HiDeS-3 NODE (ours); with energy constraint.

Figure 2B. The block of NODEs loops in the forward propagation 50 times from $t = 0$ to $t = 5$ to form a deep model. To enhance the training stability, we incorporate residual connections. The optimizer employed is NAdam (Dozat, 2016; Li et al., 2022), with an initial learning rate of 0.01, modulated using a cosine annealing learning rate scheduler (Jin et al., 2022). We train the models over 1,000 and 2,000 epochs, respectively. The loss function is the mean square error between the predicted and actual $\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(c-1)}(t)$. Given the higher dimensions of HiDeS NODEs and SONODEs compared to standard NODEs, we introduce an auxiliary loss to ensure a fair comparison. The auxiliary loss is evaluated based on predicted and actual $\boldsymbol{x}(t)$, and it is only used for comparisons rather than for training purposes.

4.2 Target chasing of multi-robot system

In this section, the NODE models are used to control a multi-robot system to chase a moving target. The target's trajectory is described by $x_1 = 0.5t$ and $x_2 = 2\sin(0.5t + 2)$. During the training phase, this trajectory serves as the ground truth to minimize the total distance between the robots and the target. In the test phase, the target location is unknown. The transition from training to test simulates a scenario in tracking processes where, despite initially having knowledge of the target location, the information regarding the target's position is lost from a certain moment onward.

As tracking problems in reality often occur under finite energy consumption, we introduce an inequality constraint on the total energy consumption of all agents: $e \leq e_{\max}$, where e represents the energy and $e_{\max}$ is the predetermined energy ceiling. This constraint is implemented during training through a regularization term as in Equation (7):

$$\check{\ell} = \ell(\boldsymbol{\omega}, \boldsymbol{\omega}^*) + \max\{e_{\max}, e\}. \tag{7}$$

During training, if $e > e_{\max}$, the term $\max\{e_{\max}, e\}$ encourages a reduction in e; If $e \leq e_{\max}$, then this term does not affect e. Since the tracking occurs on a horizontal plane, potential energy is not considered; thus, $e = \sum_i^r m_i v_i^2/2$, where r is the total number of robots. In the simulations, we set each robot's mass as equal, with the total mass being 2 kg (therefore, $e = \sum_i^r v_i^2 = \sum_{j=0}^d \sum_{i=0}^r \dot{x}_{ij}^2$), and set $e_{\max} = 5,000$ J.

4.2.1 Chasing trajectories with given target trajectory

Figures 3 and 4 respectively illustrate the trajectories of multiple robots and a chased target at different moments in time without energy constraints. By comparing Figures 3A and 4A, it can be observed that the chasing speed of the proposed HiDeS-3 NODE is significantly faster than the standard NODE. At the end of the tracking phase (e.g., at $t = 20$ s), both the standard NODE and HiDeS-3 NODE successfully reach the target. Examination of the various subfigures in Figure 4 reveals that the trajectories of the HiDeS-3 NODE exhibit typical characteristics of high-order dynamic systems similar to those seen with higher-order optimizers (Su et al., 2016; An et al., 2018) and proportional-integral-derivative (PID) controllers (Huba et al., 2023), such as rapid convergence and overshooting. This is attributed to HiDeS-3 NODE being

10.3389/fnbot.2024.1382305

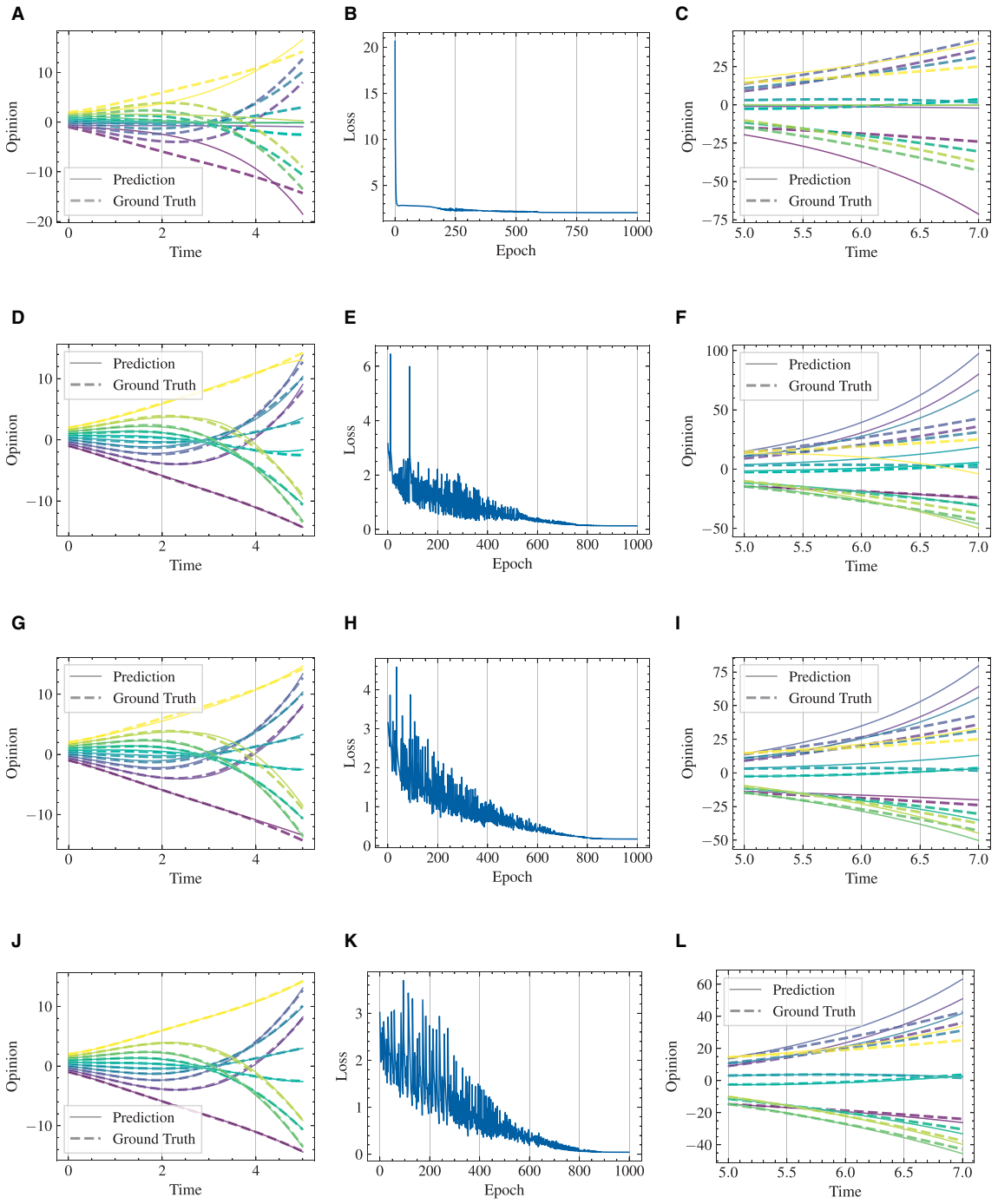


FIGURE 9
Visualization of learning results with standard NODE, SONODE, HiDeS-2 NODE, and HiDeS-3 NODE. All NODEs are trained for 1,000 epochs. **(A)** Standard NODE; opinion evolution; training. **(B)** Training loss w.r.t. epoch. **(C)** Standard NODE; opinion evolution; test. **(D)** SONODE; Opinion evolution; training. **(E)** Training loss w.r.t. epoch. **(F)** SONODE; opinion evolution; test. **(G)** HiDeS-2 NODE; opinion evolution; training. **(H)** Training loss w.r.t. epoch. **(I)** HiDeS-2 NODE; opinion evolution; test. **(J)** HiDeS-3 NODE; opinion evolution; training. **(K)** Training loss w.r.t. epoch. **(L)** HiDeS-3 NODE; opinion evolution; test.

a high-order dynamic system, as demonstrated by Equation 1. Figures 5 and 6 present a superficially similar performance between the standard NODE and the HiDeS NODE when subject to energy constraints. However, a distinct contrast emerges during the subsequent testing phase, which operates without a predefined target position.

4.2.2 Predicted trajectories without given target position

In the test phase, the target position is not given to the model. Figure 7 presents a comparison of the predicted trajectories of multi-robots in a target-chasing scenario where the target position is not provided. The comparison is between the trajectories

10.3389/fnbot.2024.1382305

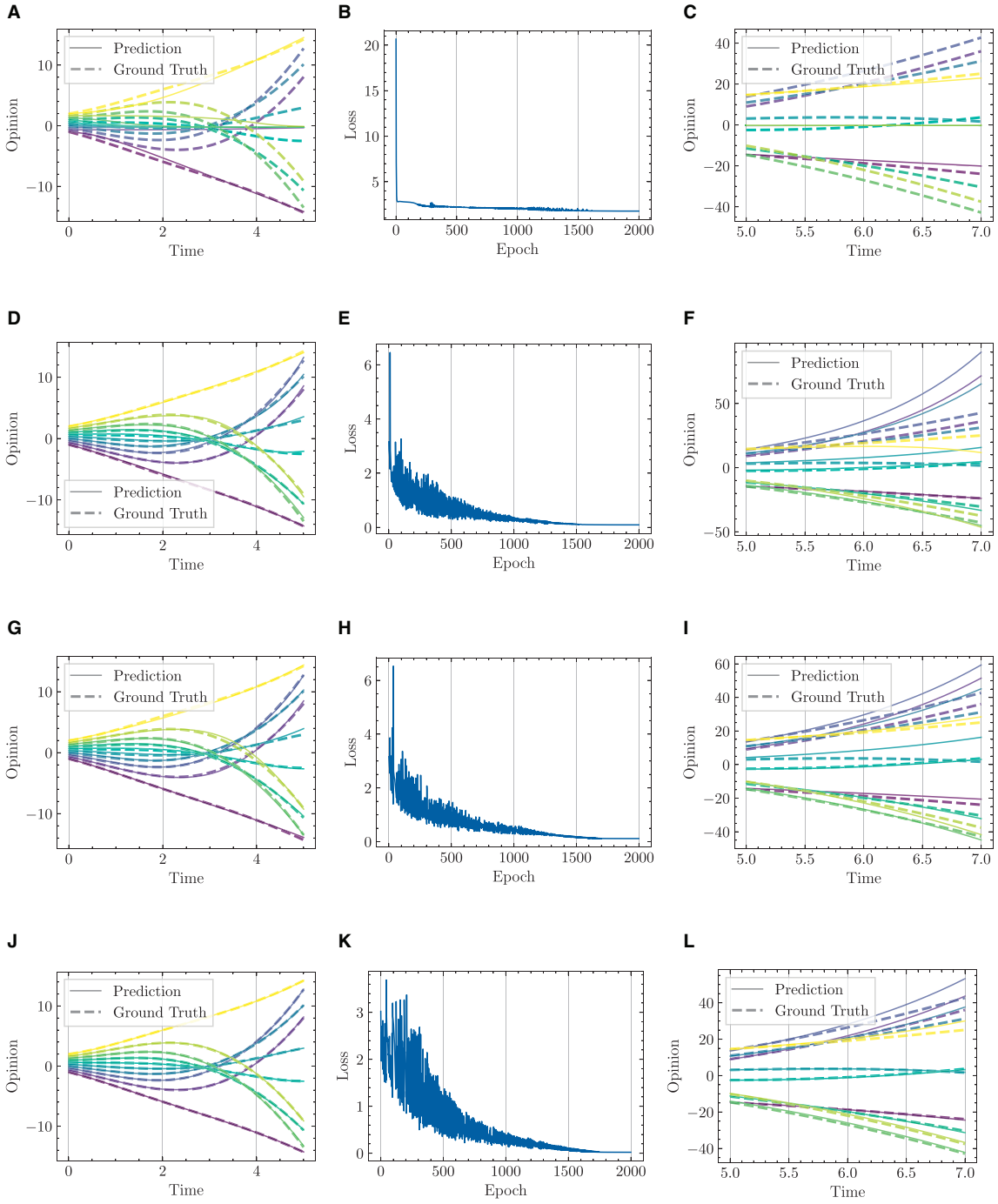


FIGURE 10
Visualization of learning results with standard NODE, SONODE, HiDeS-2 NODE, and HiDeS-3 NODE. All NODEs are trained for 2,000 epochs. **(A)** Standard NODE; opinion evolution; training. **(B)** Training loss w.r.t. epoch. **(C)** Standard NODE; opinion evolution; test. **(D)** SONODE; Opinion evolution; training. **(E)** Training loss w.r.t. epoch. **(F)** SONODE; opinion evolution; test. **(G)** HiDeS-2 NODE; opinion evolution; training. **(H)** Training loss w.r.t. epoch. **(I)** HiDeS-2 NODE; opinion evolution; test. **(J)** HiDeS-3 NODE; opinion evolution; training. **(K)** Training loss w.r.t. epoch. **(L)** HiDeS-3 NODE; opinion evolution; test.

generated by the standard NODE and the HiDeS NODE, with and without the imposition of energy constraints. The figure clearly demonstrates that the HiDeS NODE offers superior performance over the standard NODE. Specifically, the trajectories predicted by the standard NODE show significant deviations from the target (Figures 7A, C). In contrast, those predicted by the HiDeS NODE closely align with the target's trajectory. Although the HiDeS NODE with energy constraints shows slight deviations due to restricted velocity, it still significantly outperforms the standard NODE (Figures 7B, D). This suggests that the advantages of the HiDeS NODE may not solely be attributed to increased velocity but could also derive from additional information, such as curvature, which

TABLE 3 Comparisons of training and test losses among SOTA NODEs and HiDeS NODE.

# Epochs	Loss	Standard NODE (Chen et al., 2018)	SONODE (Norcliffe et al., 2020)	HiDeS-2 NODE (ours)	HiDeS-3 NODE (ours)
1,000	Training	2.05	0.13	0.17	0.04
	Test	17.50	6.97	4.42	2.03
2,000	Training	1.78	0.09	0.12	0.02
	Test	14.70	5.21	2.63	1.08

is inferred by the high-order supervised signals, as illustrated in the middle of Figure 1.

4.2.3 Energy cost

Figure 8 draws a parallel of the total energy expenditure of multiple robots engaged in a target-chasing task, contrasting the standard NODE with the HiDeS NODE. The figure shows that the implementation of energy constraints has a significant impact. Without energy constraints, both the standard NODE and the HiDeS NODE incur substantial energy costs after training, reaching up to 7,000 and 100,000 J, respectively (Figures 8A,B). However, with the application of energy constraints, both NODEs manage to keep the energy expenditure no more than 5,000 J after 1,000 epochs of training (Figures 8C, D).

4.3 Modeling and predicting of opinion dynamics

In this section, simulations of NODEs on modeling and predicting opinion dynamics are conducted.

4.3.1 One dimension, multiple initial conditions

We respectively present experimental results conducted over 1,000 and 2,000 epochs in Figures 9, 10, and complemented by Table 3. The results indicate that our HiDeS NODEs surpass the standard NODE and SONODE in capturing the subtleties of opinion dynamics. Note that the SONODE and HiDeS-2 have the same hidden layer dimensions. This superiority is evident from the more accurate approximations of actual opinion dynamics during both training and testing phases (Figures 9 and 10) and lower auxiliary losses in consistent iterations (Table 3). For instance, Figure 9 shows that the prediction curves of standard NODE diverge from the ground truth traces in both training and testing since the trajectories from different initial conditions can not cross. While SONODE performs better than standard NODE in the learning stage, its predictive ability remains inferior in the testing phase. In contrast, our models' prediction curves closely align with the ground truth. Extending the epochs to 2,000 shows that both models perform better in the learning stage than under 1,000 epochs, with our models demonstrating remarkable superiority in the testing phase. This empirical observation aligns with our theoretical analyses that using higher-order derivatives as supervised signals enhances the predictive capacity. Furthermore, as Table 3 shows, the proposed HiDeS-3 NODE exhibits significantly lower training and testing losses compared to the baseline models. Although the HiDeS-2 NODE exhibits a slightly higher training loss than SONODE, its test loss is substantially lower, indicating superior generalization ability, a key goal of neural networks.

4.3.2 Multiple dimension, one initial condition

Figure 11 presents modeling and predicting results on diverse types of individuals' intra- and inter-group interactions for opinion dynamics. The top row (Figures 11A–D) represents ground truth values, while the bottom row (Figures 11E, F) shows predictions generated by the HiDeS-3 NODE. Each subfigure illustrates different combinations of consensus and dissensus within and between groups, highlighting the model's performance in capturing extensive opinion dynamics.

4.3.3 Training dynamics

The learning results of opinion evolution are shown in Figure 12 with the increase of epochs during a 2,000-epoch training. It is clear that the learning results get more accurate and fine-grained with the increasing epochs. Specifically, the predicted results show significant deviation from the ground truth under 1–200 epochs (Figures 12B, C), while the more granular learning results are presented with the increase of epochs (Figures 12D–H), achieving more accurate predictions for opinion evolution.

5 Discussion

This paper has proposed the HiDeS NODE, a higher-order-derivative-supervised NODE, as a novel approach for modeling and predicting complex dynamics in multi-robot systems and opinion dynamics. This framework excels in capturing interactions among higher-order derivatives and the state vector, significantly enhancing modeling precision over existing NODE methodologies. The introduction of higher-order derivatives as supervised signals in the HiDeS NODE brings a superior predicting ability. Applications of the HiDeS NODE in multi-robot systems and opinion dynamics have demonstrated its effectiveness. To our knowledge, this is the first initiative that introduces NODEs into multi-robot systems and opinion dynamics. Applying the HiDeS NODE to these fields opens new avenues for broader applications in various intricate and dynamic systems.

The broader impact of the HiDeS NODE extends into numerous fields where dynamic systems play a crucial role. For

10.3389/fnbot.2024.1382305

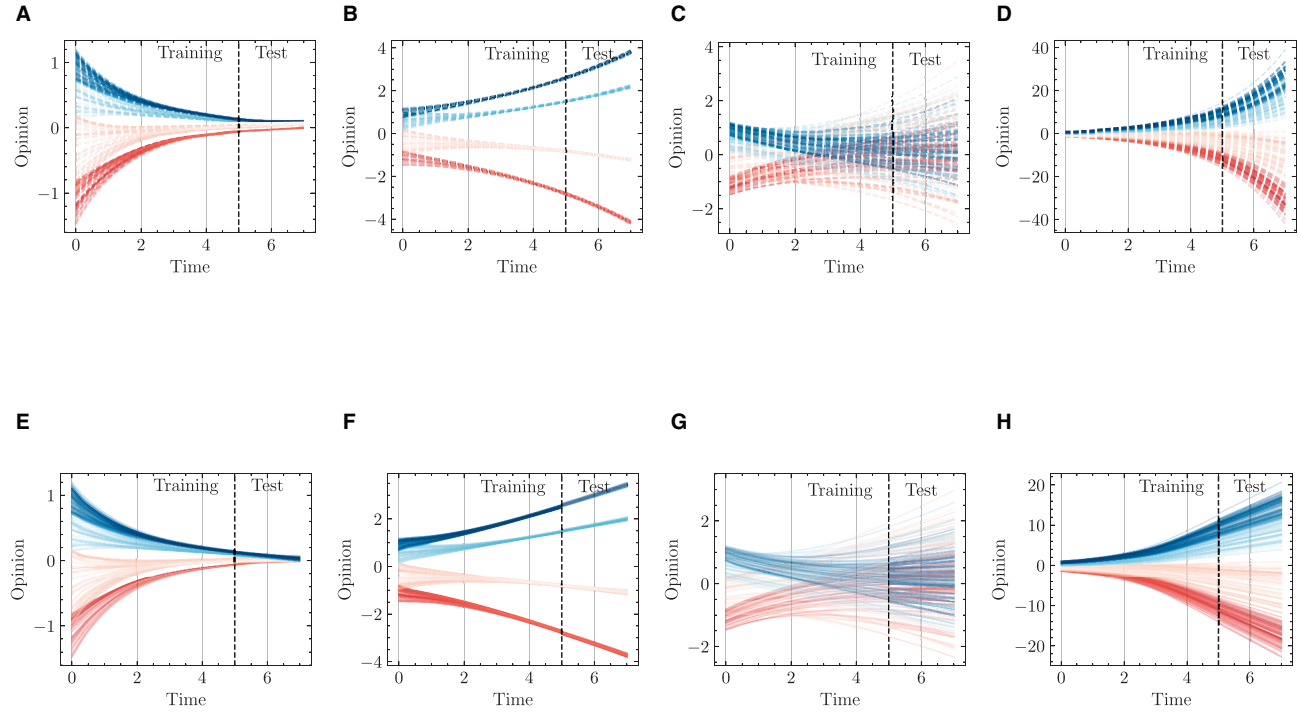


FIGURE 11
Opinion dynamics modeling and predicting using HiDeS-3 NODE for different consensus and dissensus scenarios. **(A)** Intra-group consensus and inter-group consensus, ground truth. **(B)** Intra-group consensus and inter-group dissensus, ground truth. **(C)** Intra-group dissensus and inter-group consensus, ground truth. **(D)** Intra-group dissensus and inter-group dissensus, ground truth. **(E)** Intra-group consensus and inter-group consensus, predicted by HiDeS-3 NODE. **(F)** Intra-group consensus and inter-group dissensus, predicted by HiDeS-3 NODE. **(G)** Intra-group dissensus and inter-group consensus, predicted by HiDeS-3 NODE. **(H)** Intra-group dissensus and inter-group dissensus, predicted by HiDeS-3 NODE.

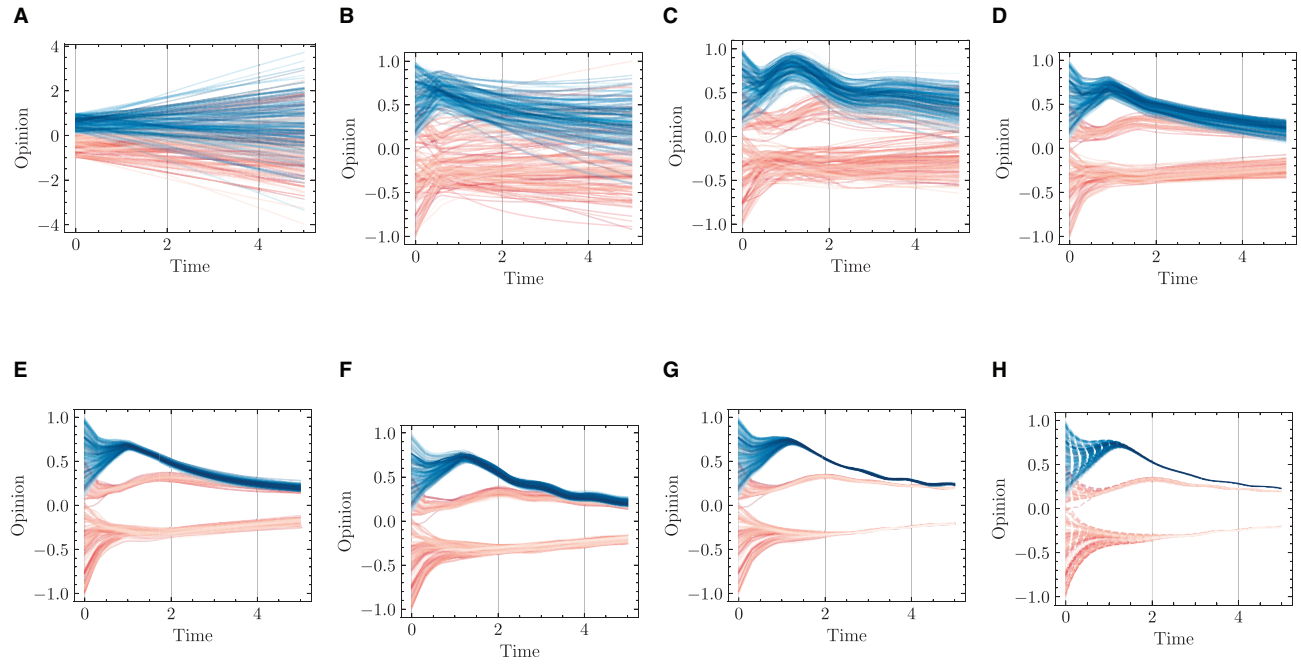


FIGURE 12
Opinion dynamics learning results during 2,000-epoch training. **(A)** Epoch = 1. **(B)** Epoch = 100. **(C)** Epoch = 200. **(D)** Epoch = 500. **(E)** Epoch = 1,000. **(F)** Epoch = 1,100. **(G)** Epoch = 2,000. **(H)** Ground truth.

instance, it has the potential to offer refined predictions of climate change effects or pollution dispersion. In healthcare, the HiDeS NODE could lead to breakthroughs in understanding the dynamics of disease spread or patient response to treatments, enabling personalized medicine. The adaptability and advanced modeling capabilities of the HiDeS NODE position it as a versatile tool capable of addressing complex problems across various domains.

Despite its potential, the HiDeS NODE faces limitations such as computational demands, particularly as the order of derivatives increases, making real-time applications challenging. The model's accuracy is heavily reliant on the quality and quantity of data, which can be a significant constraint in environments where data is sparse or noisy. Addressing these challenges will be essential for the HiDeS NODE's successful application across different fields.

A valuable future direction is to utilize real-world data to validate our model's performance in practical scenarios. Additionally, enhancing the robustness of the HiDeS NODE to noisy data presents a promising direction for future research.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

MLi: Conceptualization, Formal analysis, Methodology, Software, Writing – original draft. WB: Data curation, Funding acquisition, Resources, Supervision, Validation, Visualization, Writing – review & editing. LC: Funding acquisition, Project administration, Resources, Software, Validation, Visualization, Writing – review & editing. MLiu: Data curation, Formal analysis, Investigation, Resources, Supervision, Validation, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The handling editor LJ declared a shared affiliation with the author MLi at the time of review.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

An, W., Wang, H., Sun, Q., Xu, J., Dai, Q., and Zhang, L. (2018). "A PID controller approach for stochastic optimization of deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Los Alamitos, CA: IEEE Computer Society), 8522–8531.

Anil Kumar, N., Patrick, S., Hong, W., and Hur, P. (2022). Control framework for sloped walking with a powered transfemoral prosthesis. *Front. Neurorobot.* 15:790060. doi: 10.3389/fnbot.2021.790060

Ao, Y., and Jia, Y. (2023). Agents attraction competition in an extended Friedkin-Johnsen social network. *IEEE Transact. Cont. Netw. Syst.* 10, 1100–1112. doi: 10.1109/TCNS.2022.3220709

Aydemir, G., Akan, A. K., and Güney, F. (2023). "Adapt: efficient multi-agent trajectory prediction with adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (Los Alamitos, CA: IEEE Computer Society), 8295–8305.

Baumann, F., Lorenz-Spreen, P., Sokolov, I. M., and Starnini, M. (2020). Modeling echo chambers and polarization dynamics in social networks. *Phys. Rev. Lett.* 124:048301. doi: 10.1103/PhysRevLett.124.048301

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. *Adv. Neural Inf. Process. Syst.* 31, 6572–6583. doi: 10.5555/3327757.3327764

Cui, W., Zhang, H., Chu, H., Hu, P., and Li, Y. (2023). On robustness of neural ODEs image classifiers. *Inf. Sci.* 632, 576–593. doi: 10.1016/j.ins.2023.03.049

De, A., Bhattacharya, S., Bhattacharya, P., Ganguly, N., and Chakrabarti, S. (2014). "Learning a linear influence model from transient opinion dynamics," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (New York, NY: ACM), 401–410.

De, A., Valera, I., Ganguly, N., Bhattacharya, S., and Gomez Rodriguez, M. (2016). Learning and forecasting opinion dynamics in social networks. *Adv. Neural Inf. Process. Syst.* 29, 397–405. doi: 10.5555/3157096.3157141

Dozat, T. (2016). "Incorporating Nesterov momentum into Adam," in *International Conference on Learning Representations* (OpenReview.net).

Dupont, E., Doucet, A., and Teh, Y. W. (2019). Augmented neural ODEs. *Adv. Neural Inf. Process. Syst.* 32, 3140–3150. doi: 10.5555/3454287.3454569

Granha, M. F., Vilela, A. L., Wang, C., Nelson, K. P., and Stanley, H. E. (2022). Opinion dynamics in financial markets via random networks. *Proc. Nat. Acad. Sci. U. S. A.* 119:e2201573119. doi: 10.1073/pnas.2201573119

Hichri, B., Gallala, A., Giovannini, F., and Kedziora, S. (2022). Mobile robots path planning and mobile multirobots control: a review. *Robotica* 40, 4257–4270. doi: 10.1017/S0263574722000893

Hua, C., Cao, X., Liao, B., and Li, S. (2023). Advances on intelligent algorithms for scientific computing: an overview. *Front. Neurorobot.* 17:1190977. doi: 10.3389/fnbot.2023.1190977

Huba, M., Vrancic, D., and Bistak, P. (2023). Series PID control with higher-order derivatives for processes approximated by IPDT models. *IEEE Transact. Automat. Sci. Eng.* doi: 10.1109/TASE.2023.3296201. [Epub ahead of print].

Jin, J., Zhao, L., Chen, L., and Chen, W. (2022). A robust zeroing neural network and its applications to dynamic complex matrix equation solving and robotic manipulator trajectory tracking. *Front. Neurorobot.* 16:1065256. doi: 10.3389/fnbot.2022.1065256

Jin, L., Liu, L., Wang, X., Shang, M., and Wang, F.-Y. (2024). Physical-informed neural network for MPC-based trajectory tracking of vehicles with noise considered. *IEEE Transact. Intell. Vehicl.* doi: 10.1109/TIV.2024.3358229. [Epub ahead of print].

Károly, A. I., Galambos, P., Kuti, J., and Rudas, I. J. (2021). Deep learning in robotics: survey on model structures and training strategies. *IEEE Transact. Syst. Man Cybernet. Syst*. 51, 266–279. doi: 10.1109/TSMC.2020.3018325

Khalili, M., Zhang, X., Cao, Y., Polycarpou, M. M., and Parisini, T. (2020). Distributed fault-tolerant control of multiagent systems: an adaptive learning approach. *IEEE Transact. Neural Netw. Learn. Syst*. 31, 420–432. doi: 10.1109/TNNLS.2019.2904277

Kidger, P. (2021). *On Neural Differential Equations* (PhD thesis). Oxford: University of Oxford.

Kolarijani, M. A. S., Proskurnikov, A. V., and Esfahani, P. M. (2021). Macroscopic noisy bounded confidence models with distributed radical opinions. *IEEE Trans. Automat. Contr*. 66, 1174–1189. doi: 10.1109/TAC.2020.2994284

Kulkarni, B., Agarwal, S., De, A., Bhattacharya, S., and Ganguly, N. (2017). "SLANT+: a nonlinear model for opinion dynamics in social networks," in *IEEE International Conference on Data Mining* (Piscataway, NJ: IEEE), 931–936.

Kwa, H. L., Leong Kit, J., and Bouffanais, R. (2022). Balancing collective exploration and exploitation in multi-agent and multi-robot systems: a review. *Front. Robot. AI* 8:771520. doi: 10.3389/frobt.2021.771520

Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J. (2022). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transact. Neural Netw. Learn. Syst*. 33, 6999–7019. doi: 10.1109/TNNLS.2021.3084827

Liu, L., Wang, X., Yang, X., Liu, H., Li, J., and Wang, P. (2023). Path planning techniques for mobile robots: review and prospect. *Expert Syst. Appl*. 227:120254. doi: 10.1016/j.eswa.2023.120254

Liufu, Y., Jin, L., Shang, M., Wang, X., and Wang, F.-Y. (2024). ACP-incorporated perturbation-resistant neural dynamics controller for autonomous vehicles. *IEEE Transact. Intell. Vehicl*. doi: 10.1109/TIV.2023.3348632. [Epub ahead of print].

Monti, C., De Francisci Morales, G., and Bonchi, F. (2020). "Learning opinion dynamics from social traces," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (New York, NY: ACM), 764–773.

Norcliffe, A., Bodnar, C., Day, B., Simidjievski, N., and Liò, P. (2020). On second order behaviour in augmented neural ODEs. *Adv. Neural Inf. Process. Syst*. 33, 5911–5921. doi: 10.5555/3495724.3496220

Okawa, M., and Iwata, T. (2022). "Predicting opinion dynamics via sociologically-informed neural networks," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (New York, NY: ACM), 1306–1316.

Peng, Y., Zhao, Y., and Hu, J. (2023). On the role of community structure in evolution of opinion formation: a new bounded confidence opinion dynamics. *Inf. Sci*. 621, 672–690. doi: 10.1016/j.ins.2022.11.101

Pierpaoli, P., Li, A., Srinivasan, M., Cai, X., Coogan, S., and Egerstedt, M. (2021). A sequential composition framework for coordinating multirobot behaviors. *IEEE Transact. Robot*. 37, 864–876. doi: 10.1109/TRO.2020.3036628

Ruiz-Balet, D., and Zuazua, E. (2023). Neural ODE control for classification, approximation, and transport. *SIAM Rev*. 65, 735–773. doi: 10.1137/21M14 11433

Sander, M. E., Ablin, P., Blondel, M., and Peyré, G. (2021). "Momentum residual neural networks," in *International Conference on Machine Learning* (London: PLMR), 9276–9287.

Su, W., Boyd, S., and Candès, E. J. (2016). A differential equation for modeling Nesterov's accelerated gradient method: theory and insights. *J. Mach. Learn. Res*. 17, 1–43. doi: 10.5555/2946645.3053435

Wang, T., Wang, X., Lee-Sarwar, K. A., Litonjua, A. A., Weiss, S. T., and Sun, Y. (2023). Predicting metabolomic profiles from microbial composition through neural ordinary differential equations. *Nat. Mach. Intell*. 5, 284–293. doi: 10.1038/s42256-023-00627-3

Weinan, E. (2017). A proposal on machine learning via dynamical systems. *Commun. Math. Stat*. 1, 1–11. doi: 10.1007/s40304-017-0103-z

Wu, Z., Zhou, Q., Dong, Y., Xu, J., Altalhi, A. H., and Herrera, F. (2023). Mixed opinion dynamics based on degroot model and hegselmann-krause model in social networks. *IEEE Transact. Syst. Man Cybernet. Syst*. 53, 296–308. doi: 10.1109/TSMC.2022.3178230

Xia, H., Suliafu, V., Ji, H., Nguyen, T., Bertozzi, A., Osher, S., et al. (2021). Heavy ball neural ordinary differential equations. *Adv. Neural Inf. Process. Syst*. 34, 18646–18659. doi: 10.48550/arXiv.2110.04840

Xu, S., Xu, T., Li, D., Yang, C., Huang, C., and Wu, X. (2023). A robot motion learning method using broad learning system verified by small-scale fish-like robot. *IEEE Trans. Cybern*. 53, 6053–6065. doi: 10.1109/TCYB.2023. 3269773

Yu, X., and Chen, T. (2023). Distributed iterative learning control of nonlinear multiagent systems using controller-based dynamic linearization method. *IEEE Transact. Cybernet*. doi: 10.1109/TCYB.2023.3281479. [Epub ahead of print].

Zhang, Y., Li, P., Xu, C., Peng, X., and Qiao, R. (2023). Investigating the effects of a fractional operator on the evolution of the enso model: bifurcations, stability and numerical analysis. *Fract. Fract*. 7:602. doi: 10.3390/fractalfract 7080602

Zhu, L., He, Y., and Zhou, D. (2020). Neural opinion dynamics model for the prediction of user-level stance dynamics. *Inf. Process. Manag*. 57:102031. doi: 10.1016/j.ipm.2019.03.010

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 22 April 2024
DOI 10.3389/fnbot.2024.1396979

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Ming Wan,
Liaoning University, China
Yan Pei,
University of Aizu, Japan
Shangce Gao,
University of Toyama, Japan

*CORRESPONDENCE
Jun Wang
✉ tasi0923@163.com

RECEIVED 06 March 2024
ACCEPTED 02 April 2024
PUBLISHED 22 April 2024

CITATION
Zhang X, Ge Y, Wang Y, Wang J, Wang W and Lu L (2024) Residual learning-based robotic image analysis model for low-voltage distributed photovoltaic fault identification and positioning.
Front. Neurorobot. 18:1396979.
doi: 10.3389/fnbot.2024.1396979

Residual learning-based robotic image analysis model for low-voltage distributed photovoltaic fault identification and positioning

Xudong Zhang[1], Yunlong Ge[1], Yifeng Wang[1], Jun Wang[2]*, Wenhao Wang[2] and Lijun Lu[2]

[1]State Grid Hebei Electric Power Company, Shijiazhuang, China, [2]Henan XJ Metering Co., Ltd., Xuchang, China

With the fast development of large-scale Photovoltaic (PV) plants, the automatic PV fault identification and positioning have become an important task for the PV intelligent systems, aiming to guarantee the safety, reliability, and productivity of large-scale PV plants. In this paper, we propose a residual learning-based robotic (UAV) image analysis model for low-voltage distributed PV fault identification and positioning. In our target scenario, the unmanned aerial vehicles (UAVs) are deployed to acquire moving images of low-voltage distributed PV power plants. To get desired robustness and accuracy of PV image detection, we integrate residual learning with attention mechanism into the UAV image analysis model based on you only look once v4 (YOLOv4) network. Then, we design the sophisticated multi-scale spatial pyramid fusion and use it to optimize the YOLOv4 network for the nuanced task of fault localization within PV arrays, where the Complete-IOU loss is incorporated in the predictive modeling phase, significantly enhancing the accuracy and efficiency of fault detection. A series of experimental comparisons in terms of the accuracy of fault positioning are conducted, and the experimental results verify the feasibility and effectiveness of the proposed model in dealing with the safety and reliability maintenance of low-voltage distributed PV systems.

KEYWORDS

low-voltage distributed photovoltaics, photovoltaic identification, positioning technology, unmanned aerial vehicle imagery, horizontal comparison experiment

1 Introduction

In recent years, the Photovoltaic (PV) energy has experienced a fast development process, and increasingly, it plays an important role in our daily life due to the advantages of easy installation, cleaning, reasonable return period, and short construction period (Akram et al., 2020; Maka et al., 2021). Compared to conventional power generation approaches, PV power generation shows more superiorities such as safety, reliability, noiselessness, environmentally friendliness, resource distribution, and has a high energy quality and short construction period (Ali et al., 2020; Stiubiener et al., 2020). Especially, with the increasing energy demand, large-scale PV installations have received a surge of attentions, and this leads to the establishment of a mature PV market and technological

innovation in the PV industry (Ali et al., 2020; Ma et al., 2022b). With the rapid development of large-scale PV power plants, automatic identification and localization of PV system faults are of critical importance to improving the safety, reliability, and productivity of PV systems. Faults in PV systems can reduce system efficiency and pose potential safety hazards, necessitating the proactive detection of potential faults and the implementation of appropriate corrective measures.

Typically, a large-scale PV system contain massive solar modules, which are variously interconnected with each other. Once one of them fail to work well, the efficiency of the PV system will be largely degraded. Therefore, it is urgently needed to actively detect any potential faults such that suitable corrective measures can be employed before the disruptions happen. To reach this goal, various artificial intelligence (AI) and robotic techniques have been utilized in the fault positioning/diagnosis of PVPP to enhance the intelligent processing capability (Ma et al., 2022b; Wang et al., 2024). Especially, due to the advantages of mobility, flexibility, programmability and large-area coverage, the unmanned aerial vehicles (UAVs) have been widely employed in the PV plant to capture moving images of various PV modules in the distributed power plants. Through intelligent image analysis, we can inspect specific PV faults in an efficient way, which is key to improve the operation and maintenance (O&M) level of the power plant (Atsu et al., 2020; Chen et al., 2020; Abubakar et al., 2021; Navid et al., 2021; Cui et al., 2022).

In fact, many deep learning models have been utilized for the improvement of PV-system fault diagnosis, such as the residual neural network, convolutional neural network, and semi-supervised ladder network (Atsu et al., 2020; Aziz et al., 2020). However, how to develop an effective AI-based approach, which can fully exploit useful fault feature information in UAV images, still is a challenging issue to enhance the fault diagnosis performance.

Regarding the on-site deployment of UAVs for inspection purposes, data used for inspection should be collected and processed first, including geographic information, environmental features, and specific requirements of the expected task. According to the characteristics of the target scene and the requirements of the task, the flight route, inspection frequency and data collection method of the UAV should be designed and implemented. The quality of PV images is challenging to guarantee, and issues such as under-exposure, high noise, and blurred details are frequently encountered. Moreover, the diversity of PV faults and the significant difference in image features among different faults pose significant challenges to the performance of image analysis algorithms.

In this paper, we propose a residual learning-based UAV image analysis model for low-voltage distributed PV fault identification and positioning. In our target PV power plants, the UAVs are deployed to acquire moving images of low-voltage distributed PV products. Based on YOLOv4 network, we integrate residual learning with attention mechanism into the UAV image analysis model, aiming to improve the robustness and accuracy of PV image detection. Then, we propose a sophisticated multi-scale spatial pyramid fusion method and use it to optimize the YOLOv4 network for the nuanced task of fault localization within PV arrays, where the Complete-IOU loss is used in the predictive modeling phase, which is able to significantly enhance the accuracy and efficiency of fault detection. To augment the size of the dataset, this paper adopts data augmentation techniques to flip and adjust the original samples. The image data used comes from three real power plants. The experimental results on a series of datasets verify the effectiveness of the proposed model.

The contributions of this paper are as follows:

1. Novel residual learning-based UAV image analysis model: A residual learning-based UAV image analysis model is proposed for PV fault recognition, where the residual learning network is constructed based on attention mechanism. This model is able to well exploit useful fault feature from UAV moving images, and then significantly boosts the accuracy and efficiency of PV fault identification and localization.
2. Enhanced YOLOv4 optimization for precise fault localization: An improved optimization method is designed to optimize YOLOv4 network, where a sophisticated multi-scale spatial pyramid fusion aims to optimize the model for the nuanced task of fault localization within PV arrays, while the Complete-IOU loss is used in the predictive modeling phase to enhance the accuracy of fault diagnosis in PV systems.
3. Extensive validation on real-world datasets: The proposed model has been trained and tested on a set of datasets to detect its application capability of PV fault detection and diagnosis.

The rest of the paper is organized as follows: section 2 reviews the related work on PV identification and fault positioning research. Section 3 describes the proposed methods in detail. Section 4 reports the experimental result and analysis. Section 5 represents the conclusion and future work.

2 Related work

Currently, many deep learning techniques have been successfully applied to real-world scenarios and effectively solved various challenging problems. For example, Jin et al. (2024) proposed a physical-informed neural network model with model predictive control controller and a perturbation-resistant neural dynamics controller equipped with the noise-suppression ability (Liufu et al., 2024). These two models addressed the challenges faced by autonomous vehicle control systems when dealing with unpredictable external environments and internal system noise disturbances. In the field of PV identification and fault localization, deep learning also has many applications.

Photovoltaic identification and fault localization are important components of PV O&M. It aims to timely discover and eliminate various faults in PV systems, thus improving the performance and safety of systems. Possible faults in PV systems include open-circuit faults, short-circuit faults, hot spot faults, etc. PV faults can be classified according to their occurrence time, severity, persistence, and cause, such as infant failures, midlife failures, wear-out failures, acute failures, chronic failures, permanent failures, and temporary failures (Hong and Pula, 2022). These faults will cause the output power of PV systems to decrease, and even lead to serious consequences such as fire. Therefore, effective methods are needed to diagnose and locate faults in PV systems.

Fault detection methods based on electrical data are a type of method that uses historical or real-time data of voltage, current, power, etc. in PV systems to extract feature information through data mining, machine learning and artificial intelligence technologies. This

feature information is then used to establish fault identification and location classifiers or regressors. Some researchers used data analysis techniques such as feature extraction, clustering, classification, and regression to estimate the faults location and severity of PV arrays based on the voltage and current data in PV systems (Alajmi and Abdel-Qader, 2016; Chen and Wang, 2017; Fadhel et al., 2019; Patil and Hinge, 2019). Fadhel et al. proposed a framework for PV faults detection and localization based on hybrid data, which combines voltage–current sensor network and environmental sensors, such as temperature, humidity, and irradiance. It improves the accuracy and robustness of PV faults detection and localization (Alajmi et al., 2018). These methods have the advantages of being independent of model parameters and adaptable to complex environmental conditions. However, it also has the disadvantages of requiring additional sensors, instruments and circuits, and being dependent on environmental conditions.

Computer vision-based fault detection methods use artificial intelligence technologies such as deep learning to process and interpret large amounts of visible or infrared images in PV systems, thus achieving the automation and intelligence of PV fault identification and localization. Some researchers proposed a PV fault detection and localization method based on convolutional neural networks. This method used the thermal imaging data of PV systems to achieve the detection and localization of open circuit faults, short circuit faults, hot spot faults, and shadow faults in PV arrays through image processing, feature extraction, and classifier training (Ali et al., 2020; Herraiz et al., 2020; Alves et al., 2021). Korkmaz et al. proposed a PV fault classification method based on transfer learning and multi-scale convolutional neural networks. This method used the visible light image data of PV systems to achieve the classification of open circuit faults, short circuit faults, hot spot faults, and shadow faults of PV modules through image preprocessing, feature extraction, and classifier training (Korkmaz and Acikgoz, 2022). These methods have the advantages of being able to handle complex nonlinear problems, as well as having self-adaptation and learning ability. However, they also have the disadvantages of requiring a lot of training data, computing resources and algorithm optimization.

This section provides an overview of research concerning photovoltaic identification and fault localization. It primarily explores fault types, diagnostic methods, localization techniques and deep learning models related to photovoltaic modules. However, existing fault Identification and positioning techniques for PV systems based on current location suffer from poor robustness, especially when environmental conditions change. Computer vision-based fault Identification and positioning methods may not achieve the expected accuracy when the data are insufficient or the training is inadequate. In addition, the processing range of image processing methods is narrow, which may not cover all types of faults in PV systems. In response to these limitations, we propose methods of photovoltaic identification and fault localization in low-voltage distributed PVPP.

3 Methods

This section elaborates on the methods of photovoltaic identification and fault localization in low-voltage distributed PVPP, emphasizing the integration and optimization of deep learning technology, especially in the context of drone imaging. The structure of the method presented in this paper is shown in Figure 1.

The primary aim of this framework is to enhance photovoltaic identification and fault localization using deep learning techniques, in order to improve the operational efficiency and safety of photovoltaic power plants. First, the relevant information of low-voltage distributed photovoltaic power plants is acquired. Then, the deep learning model undergoes analysis, and the identification model earmarked for optimization is determined. The model is improved by combining residual U-shaped module composed of dilated convolution and residual network, achieving accurate identification of the photovoltaic area. Finally, the photovoltaic fault localization technique is refined using multi-scale spatial pyramid fusion, Complete-IOU loss and self-attention mechanism, thereby achieving precise fault positioning in photovoltaic power plants.

3.1 PV identification and fault positioning of low-voltage distributed PVPP

A PVPP is a facility that uses light energy to convert it into electricity. It is composed of solar panels and inverters. Solar panels convert sunlight into DC electricity. Inverters convert DC energy into alternating current energy, which is fed into the grid. PVPPs can be divided into centralized and distributed according to their nature. Distributed PVPPs are often installed in factories, residential roofs, fish ponds, and other small ground or building areas, generally connected to the grid through 380 V voltage. It operates flexibly and independently of the grid under the right conditions (Gallardo-Saavedra et al., 2020). The advantages of PVPPs include no emissions, low maintenance costs, and long life. It can effectively reduce environmental pollution and energy consumption. However, the disadvantage is that it depends on the weather and light intensity, and the construction and maintenance costs are relatively high (Gao et al., 2021). As a PV system with rapid growth in installed capacity in the next few years, more and more PV power distribution and installation urgently need efficient and low-cost PVPP health inspection methods to detect the function of PV modules and ensure the normal operation of the system (Jie et al., 2020). Figure 2 shows the actual picture of the low-voltage distributed PVPP.

In a complete PV-EG system, after connecting several PV modules in series, a PV string with DC output is formed, which is called a string. The PV brackets used in PVPP can be divided into fixed and tracking brackets. The tracking brackets can automatically adjust the direction to maximize production capacity (Khalil et al., 2020). In the traditional PV bracket unit design, the PV modules are arranged in vertical double rows or horizontal three and four rows. A bracket unit is usually equipped with one or two strings. The number of modules is determined by the number of modules connected in series in the PV string. PV modules as the core EG module, from top to bottom: glass, upper encapsulant, cell, lower encapsulant, backplane, frame, and junction box (Li et al., 2021).

When a hot spot fails on the cell, highlighted areas appear in the infrared image. Therefore, for defect detection, infrared image defect diagnosis is one of the most widely used defect detection methods. Infrared thermal imaging cameras can detect faults caused by internal defects in PV modules and judge the severity of the fault based on the temperature in a non-invasive way. Inspection UAVs can obtain

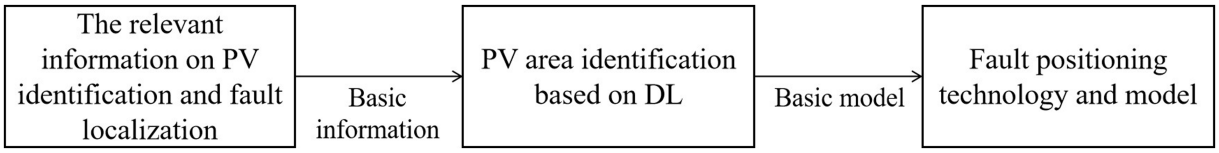


FIGURE 1
Methodological framework of the study.

FIGURE 2
Real view of low-voltage distributed PVPP.

information, including altitude and temperature, by integrating UAVs, sensors and infrared cameras (Liang et al., 2020a). The detection process is shown in Figure 3.

3.2 PV area identification based on DL

Photovoltaic identification identifies important information, such as the brand and model of the solar panel, by extracting and classifying the features of the solar panel. PV identification plays an important role in the assessment, supervision, and management of PVPPs. Traditional PV identification identifies the transmitted back image through UAV. For most infrared images obtained by PVPP, the PV area absorbs more heat and has a more obvious temperature difference from the ground. The grayscale histogram will have obvious peaks. The PV area can be obtained using a suitable gray threshold method (Liang et al., 2020b). For single-peaked prominent pictures, simple pre-and post-scene separation is done. The maximum inter-class variance threshold segmentation algorithm is considered to find the best threshold for segmentation (Liu et al., 2020). The principle is to convolve the original image through a convolution check of a specific shape and size. When a point pixel of the original image and its surrounding pixels can form a convolution kernel shape, it is retained. If not, it is deleted. The expansion operation, on the contrary, can be used to enlarge the objects in the image or recombine the separated pixels (Lyu et al., 2020).

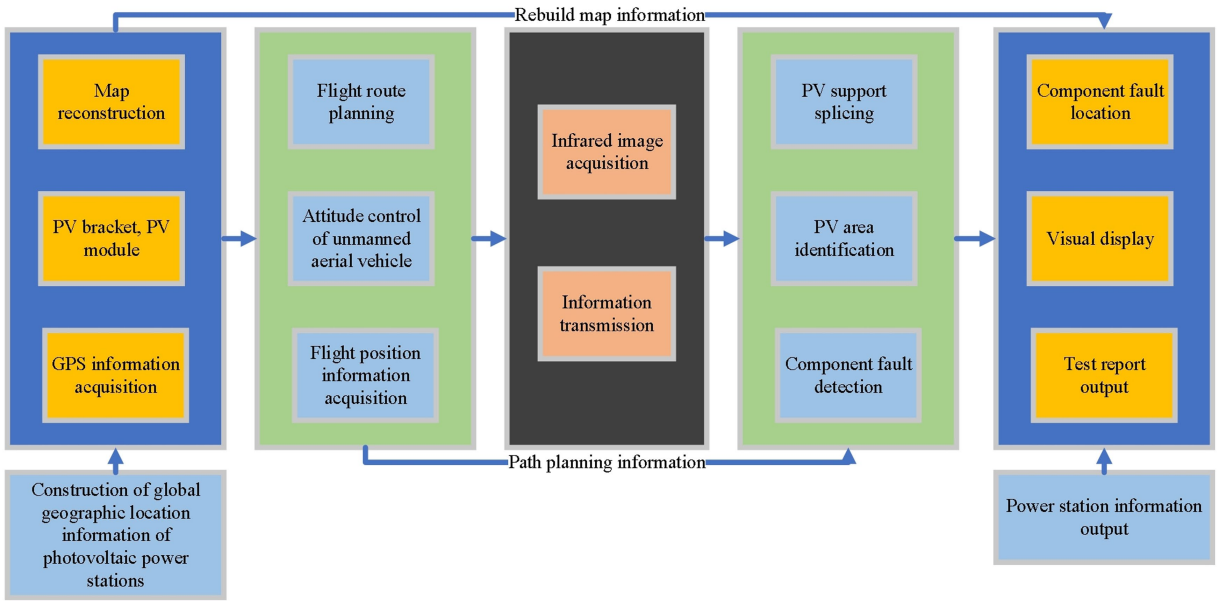


FIGURE 3
UAV infrared image fault diagnosis process.

For traditional image processing algorithms, the process is intuitive, and the calculation is simple and effective. However, the processing scenario is single, and the robustness is low. When the ambient temperature and the PV area temperature are similar, or there is interference in the environment, it is difficult to determine the threshold by the maximum inter-class variance threshold segmentation algorithm in the grayscale histogram. The algorithm directly fails (Manno et al., 2021). The PV region in the infrared image has a typical visual salience, which can be regarded as a binary pixel classification problem in semantic segmentation. The purpose of the semantic segmentation task is to correctly classify each pixel of an image. Each pixel can be classified into a corresponding category by manually determining the semantic label of each pixel and learning (Marqusee et al., 2021). In the PV area visual recognition task, the pixels in the infrared image can be divided into two categories. The target pixel is the PV area pixel, and all the remaining pixels are classified as the background. So, the U-Network (U-Net) semantic segmentation model is introduced. Figure 4 displays the structure of the U-Net model.

The left side of Figure 4 is the downsampling part. It is found that the feature map decreases, and the number of channels increases. The same convolution kernel on the right side is upsampled using the bilinear interpolation method. At present, the U-Net model is affected by the size of the convolution kernel, resulting in small receptive fields and incomplete information capture. Therefore, the receptive field is increased by pooling and dilated convolution. The pooling operation is accompanied by decreased resolution while increasing the receptive field. Dilated convolution can increase the receptive field without increasing the number of parameters and reducing the convolution kernel.

$$k' = k + (k-1) * (d-1) \tag{1}$$

$$RF_{i+1} = RF_i + \left[(k'-1) * \prod_{i=1}^{i} Stride_i \right] \tag{2}$$

In Equations (1, 2), k is the convolution kernel size, and k' is the dilated convolution kernel size. d is the dilation rate hyper-parameter. RF_i is the receptive field of the previous layer. RF_{i+1} is the current receptive field. $\prod_{i=1}^{i} Stride_i$ is the stride product, and i is the number of layers. As the network deepens, degradation and gradient vanishing problems may occur, resulting in deep neural networks that are difficult to train. Although many studies have optimized network architectures, the problem has not been fully solved yet (Li et al., 2023). So, residual learning is introduced. Residual learning is easier than direct learning, so errors can be calculated through residual learning.

$$F(x) = H(x) - x \tag{3}$$

In Equation (3), x is the input information, $F(x)$ is the result of residual learning, and $H(x)$ is the result of direct learning. Deeper structures under reasonable calculation can be constructed through the residual U-shaped module composed of dilated convolution and residual network to obtain multi-scale features. The residual U-shaped module retrieves edge features through feature joining, alleviating the problem that the receptive field of simple convolution is too small to capture global information. The overall structure of the optimized model is shown in Figure 5.

Its main architecture is a classical U-Net network similar to Encoder-Decoder, a simple and effective saliency object detection network that can be used for the semantic segmentation of two

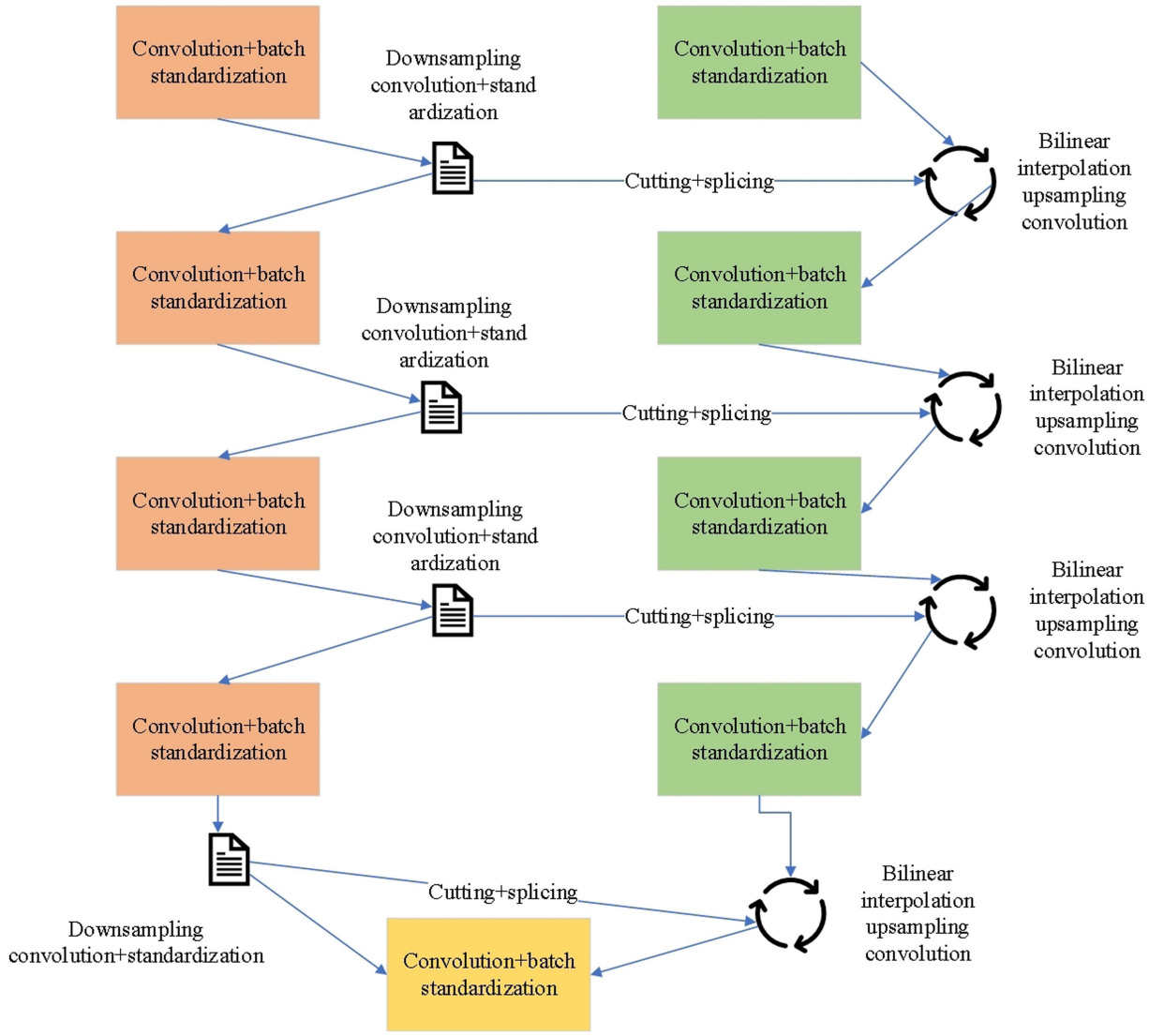


FIGURE 4
Net semantic segmentation model.

classifications. Each stage comprises residual U-shaped modules, which can more effectively capture the global features of multi-scale features. Specific residual U-shaped modules are used in different encoder and decoder stages. From the first layer to the seventh layer, the information structure of the picture has been analyzed and deconstructed. Residual learning enables the model to construct deeper structures and extract multi-scale features with reasonable computation, which facilitates effective visual identification of PV areas under various power plant scenarios and environmental interferences. It also empowers the model with better adaptability and learning ability when dealing with complex non-linear problems. The advantage of this is that more and more feature information will be extracted. The key components of the optimized model is shown in Table 1.

The process of the proposed optimized semantic segmentation model can be as follows:

Step 1: The input image is fed into the input layer of the network for feature extraction.

Step 2: In encoder stage 1, the residual U-shaped module performs a preliminary feature analysis on the image, extracting low-level edge and texture information.

Step 3: In encoder stage 2, the enhanced residual U-shaped module conducts a deeper feature analysis on the image, extracting high-level semantic and structural information.

Step 4: At the intermediate layer, a complex feature integration method combines features at different scales and resolutions, resulting in a global feature representation.

Step 5: In decoder stage 1, bilinear interpolation upsampling and convolution operations reconstruct the features, while concatenation operations connect the features from encoder stage 1 with the features from decoder stage 1, achieving global information transfer.

Step 6: In decoder stage 2, bilinear interpolation upsampling and convolution operations further reconstruct the features, while concatenation operations connect the features from the input layer with the features from decoder stage 2, achieving detail information recovery.

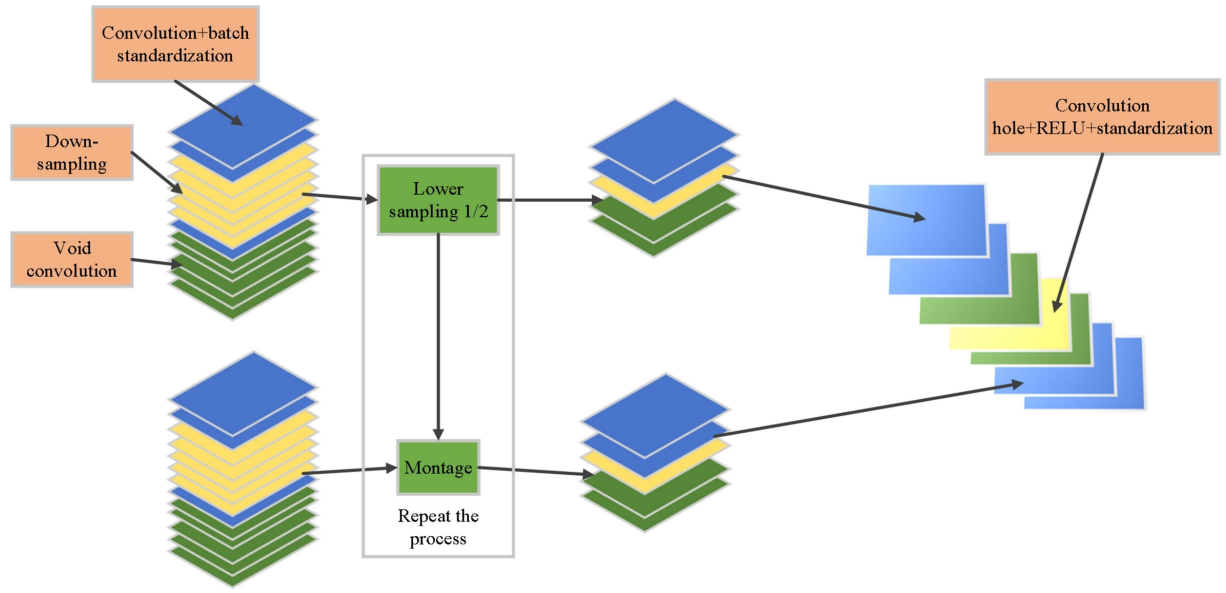


FIGURE 5
Optimized semantic segmentation model.

TABLE 1 The key components of the optimized model.

Layer number	Component	Description
1	Input layer	Initial layer receiving the input image.
2	Encoder stage 1	First stage of the encoder with residual U-shaped module for initial feature extraction.
3	Encoder stage 2	Second stage of the encoder with enhanced residual U-shaped module for deeper feature analysis.
4	Intermediate layer	Central layer of the network, bridging encoder and decoder, with complex feature integration.
5	Decoder stage 1	First stage of the decoder, reconstructing features and integrating global information.
6	Decoder stage 2	Second stage of the decoder, refining the feature reconstruction and detail enhancement.
7	Output layer	Final layer producing the segmented image output, representing the analyzed and processed features.

Step 7: At the output layer, convolution operations transform the features into a binary semantic segmentation image, representing the analyzed and processed features.

3.3 Fault positioning technology and model of low-voltage distributed PVPP

For the infrared image fault diagnosis system of PVPP based on UAV inspection, the health of the components is the most concerned item of the power plant after obtaining infrared images. The corresponding PV module information can be used to guide subsequent O&M after making judgments in infrared fault diagnosis (Mellit and Kalogirou, 2021). In the UAV inspection proposed here, the shooting height can be set in the path planning stage. The field of view of the infrared picture can be locked by collecting the original power plant information and camera angle. The main PV mounts are adjusted horizontally to ensure m complete brackets in one infrared image or one complete bracket in n continuous images (Qais et al., 2020). For the resulting profile composed of several points, the vertex information can be obtained by quadrilateral fitting by the Ramer-Douglas-Peucker algorithm (Quiles et al., 2020). This method, also known as the iterative endpoint fitting algorithm, is an algorithm that approximates the curve as a series of points and reduces the number of points. The contour points obtained are a subset of the original contour (Ramadan et al., 2020).

The Line Segment Detector method is a linear detection method with low time complexity. It forms a horizontal line field by first calculating the horizontal line angle within the eight neighborhoods of each pixel. The vertical angle of the gradient direction of this pixel is the horizontal line angle, and the horizontal line field is a matrix corresponding to the points in the image one by one (Ridha et al., 2020). After obtaining the horizontal line field, the area growth method is used to generate several connected domains according to the horizontal line angle. The threshold t is set to 22.5 degrees, and the horizontal line angle change of all pixels in each connected domain cannot exceed t. The connected domain obtained at this time is called the line support area. Each line support area is a candidate for segment detection. Each line support area corresponds to a matrix, represented by its smallest circumscribed rectangle (Ridha et al., 2021). The line support area's spindle direction is the matrix's major axis direction, and the rectangle covers the entire area. The smallest external rectangle represents the straight-line information. The confirmation

line can be obtained by filtering the fitted rectangle information. The complete single PV module can be obtained by simple area extraction (Romero-Fiances et al., 2022).

Common fault positioning classification models are convolutional neural networks (CNNs) and Transformer, and the dominant CNNs benefit from their inductive bias. For example, the adjacent area has adjacent and translation invariance characteristics. The implied visual prior knowledge can be effectively used to extract information. Therefore, there is good ingestion even for small data, although many studies have been conducted to optimize the CNN architecture (Ma et al., 2023a), which may still lead to performance degradation (Tsanakas et al., 2020). In sequence-related tasks in natural language processing (NLP) neighborhoods, recurrent neural networks (RNNs) with fixed structure memory units simplify the difficulty of long-distance learning and significantly outperform other RNNs (Veerasamy et al., 2021). However, with the introduction of the attention mechanism, it broke through the field of traditional NLP and became the model with the best performance. Figure 6 demonstrates the structure of multi-headed self-attention mechanism.

Compared with CNNs, calculating the association between two positions in the AM only requires calculating the association weight between the two pairs. In contrast, multi-layer convolution is required in CNNs to obtain the relationship between distant positions. However, the model based on the self-attention mechanism needs to calculate more parameters and lacks the inductive bias brought by convolution. CNN has more advantages when there is less data. The target detection network is the dual information determination task of target position and category information (Yang et al., 2021). Traditional detection modes use a combination of candidate boxes and feature extraction. For example, manually designed features are extracted in each window by sliding windows. Then, the features are obtained with a simple classifier (Zghaibeh et al., 2022; Ma et al., 2022a, 2023b). Here, the you only look once v4 (YOLOv4) network is needed. Its overall loss function consists of regression box loss, confidence loss, and classification loss. A threshold is set for confidence. For lower thresholds, no classification loss occurs. This method can solve the problem of sample imbalance well, which is suitable for the detection target required here. The formula of Complete-IOU loss for YOLOv4 is shown below:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2\left(b,b^{gt}\right)}{c^2} + \alpha v \tag{4}$$

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2 \tag{5}$$

$$\alpha = \frac{v}{(1 - IoU) + v} \tag{6}$$

In Equations (4–6), IoU is the overlap between the predicted bounding box and the ground truth bounding box. $\frac{\rho^2\left(b,b^{gt}\right)}{c^2}$ is the Euclidean distance between the centers of the predicted bounding

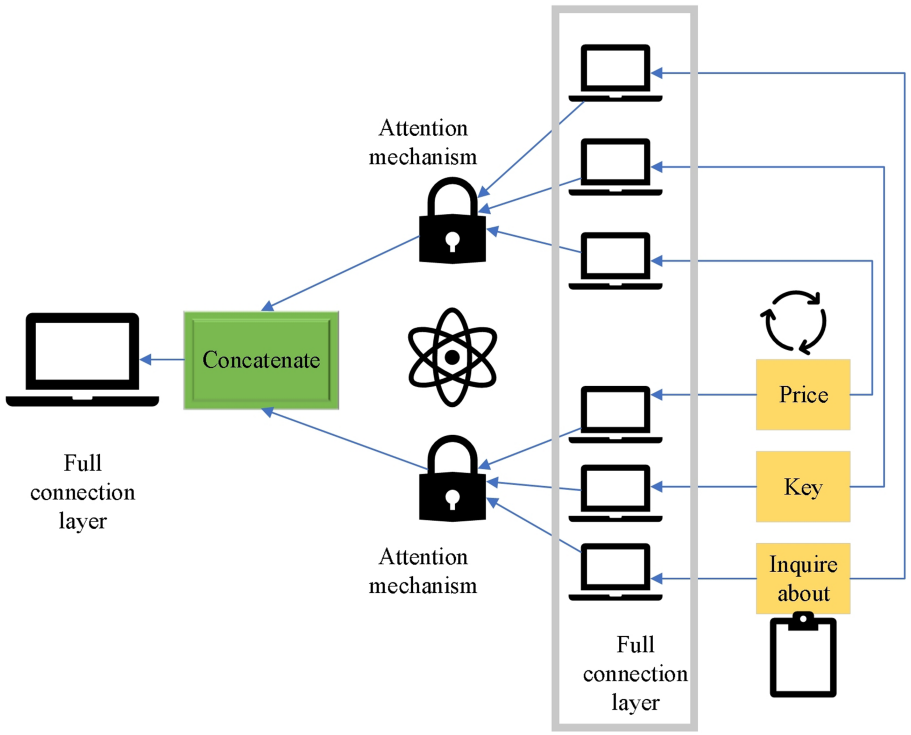


FIGURE 6
Multi-headed self-attention mechanism.

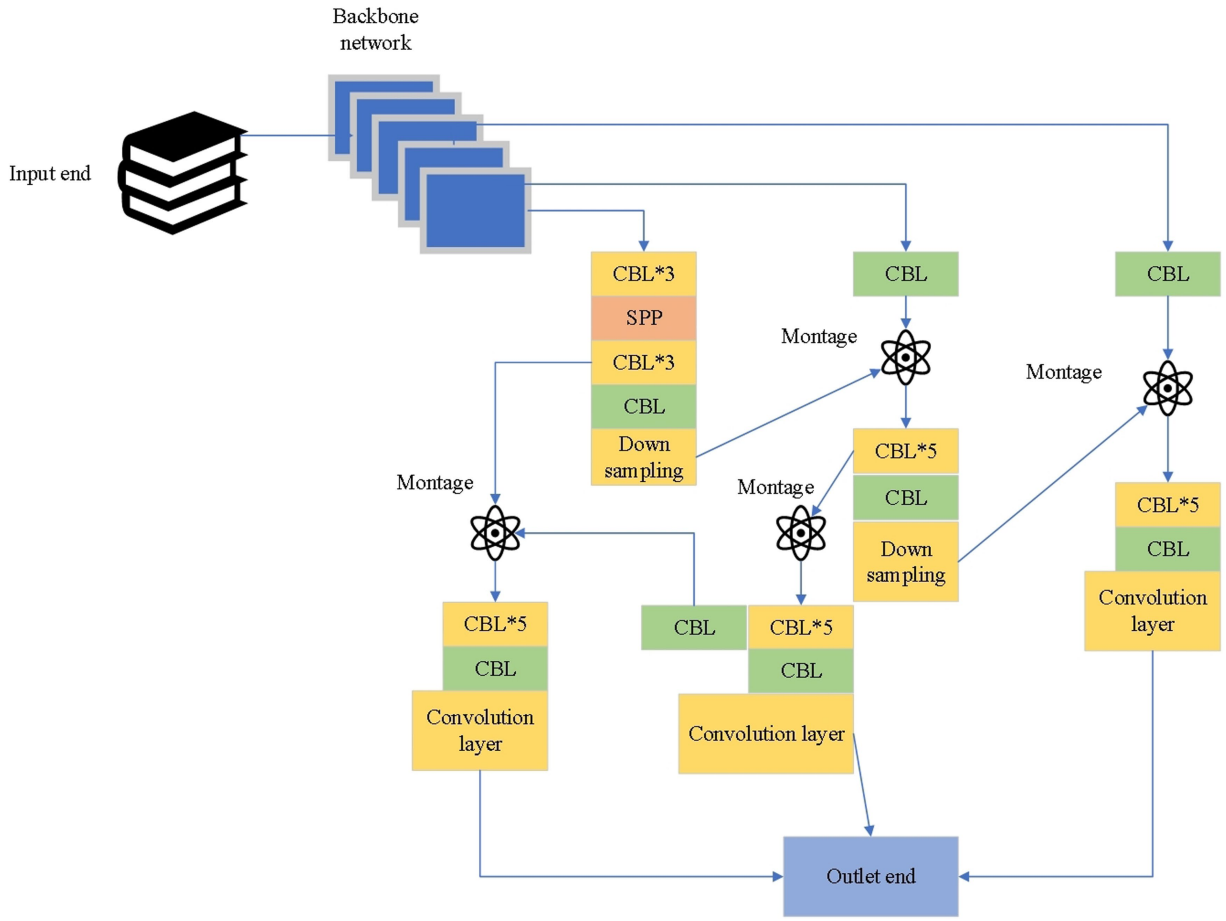


FIGURE 7
Optimized structure of YOLOv4.

box (b) and the ground truth bounding box (b^{gt}), normalized by the diagonal length of the smallest enclosing box covering both boxes. c^2 is the diagonal length of the smallest enclosing box covering both the predicted and ground truth bounding boxes. αv is a value that represents the aspect ratio difference between the predicted bounding box and the ground truth bounding box. The optimized YOLOv4 network structure is given in Figure 7.

Based on the optimized structure of YOLOv4 shown above, a PV positioning method of low-voltage distributed PVPP is proposed, and its process is as follows:

Step 1: The feature map obtained from the backbone network is fed into the prediction network. The CBL includes three components: convolution (Conv), batch normalization (Batch Norm), and leaky rectified linear unit (Leaky ReLU).

Step 2: Introduce additional layers in the prediction network to optimize the algorithm performance, enhance the feature detection capabilities, and increase the sensitivity to the nuances of the PV array images.

Step 3: After using a multi-scale fusion of spatial pyramid pooling (SPP), YOLOv4 improves the feature extraction capability through the feature pyramid and path aggregation network fusion structure. The feature pyramid network, in conjunction with the path aggregation network, forms a fusion structure that effectively consolidates features at various scales and resolutions.

Step 4: Complete-IOU loss is used in the prediction, which integrates the prediction box boundary non-coincidence, center distance information, and aspect ratio information so that the regression operation of the prediction box obtains fast speed and high accuracy. The utilization of Complete-IOU loss results in a more nuanced and detailed regression operation of the prediction box.

This enhances both the speed and accuracy of the algorithm, allowing for rapid processing without compromising the quality of fault detection. The accurate detection of boundaries, combined with the precise localization of faults, makes our optimized YOLOv4 algorithm particularly effective for PV fault positioning. This level of accuracy is critical in the context of PV maintenance, where the timely and precise identification of faults can significantly impact the efficiency and longevity of PV installations.

4 Experimental result and analysis

4.1 Comparative analysis of experimental results of PV area identification

The image data used in the experimental training set comes from three real power plants. They represent three types of power plants: mountain PV, fishery-solar complementary, and agro-solar

TABLE 2 PV identification dataset.

Power plant	Number of samples	Number of samples (data expansion)
Training set 1	120	1,200
Training set 2	195	1,950
Training set 3	494	4,940
Test set 1	97	X
Test set 2	42	X
Test set 3	43	X

complementary. They are named training set 1, training set 2, and training set 3. The test data set is also selected from three real power plants, called test set 1, test set 2, and test set 3. The original samples of the dataset are flipped and adjusted for broadening. We set the ratio of the training sets to the tests set at 8:2. The specific sample number of the dataset is shown in Table 2.

The models participating in the comparative experiment include Otsu, Grabcut, U-Net, Recurrent Residual CNN-based U-Net (RES U-Net), U2-Net, and the optimized model proposed here. The structure and hyperparameter settings of the algorithm used in the experiment are shown in Table 3.

TABLE 3 Structure and hyperparameter settings.

Hyperparameter	Value
Initial learning rate	0.001
Momentum term	0.949
Convolution Kernel size (k)	5×5
Batch size	64
Dilation rate	3

There are three experimental evaluation indicators: accuracy, recall, and Intersection over Union (IoU). Accuracy represents the proportion of correctly identified PV area pixels to the total pixels. Recall represents the proportion of correctly identified PV area pixels to the actual PV area pixels. IoU represents the ratio of the intersection of correctly identified PV area pixels and actual PV area pixels to their union. Figure 8 reveals the results.

From Figure 8, in the comparative experiments of accuracy and IoU, the corresponding data of the optimized model are the highest, 96.5 and 95.07%, respectively. This indicates that the model can effectively distinguish between the PV area and the background area, and has a high degree of matching with the real PV area. Regarding recall, the optimized model has the highest recall rate of 98.46%, except that it is not as high as the Grabcut model of 98.54%. Its recall rate is only 0.08% lower than the Grabcut model. This indicates that the model can cover most of the real PV areas, but there are also a few cases of missed detection. Comparative experiments can show that the optimized model has high adaptability to different power plant

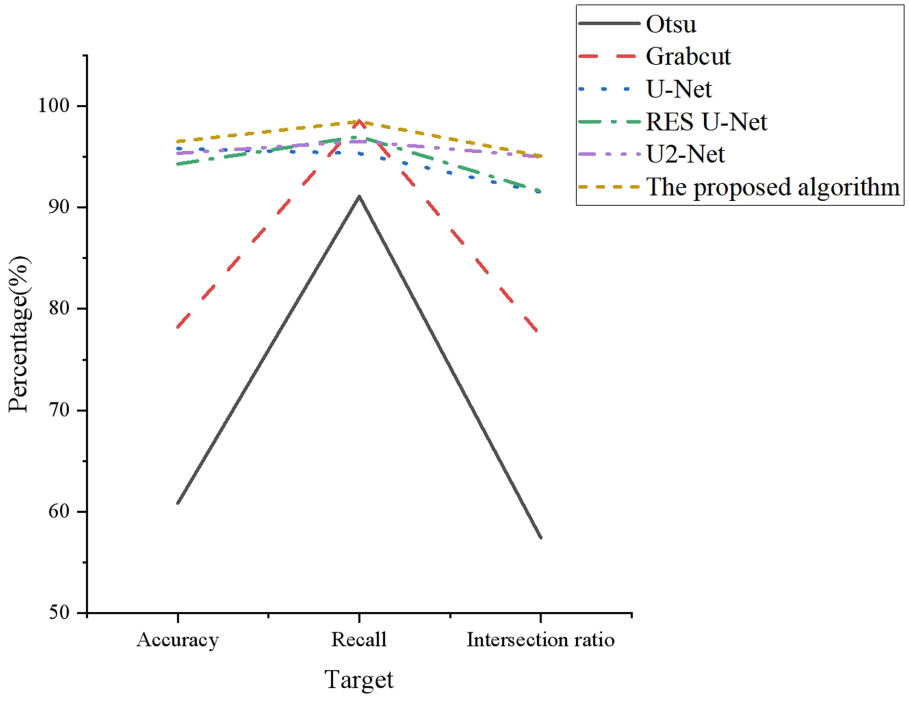


FIGURE 8
Statistics of identification results.

scenarios and environmental disturbances. Additionally, it has a high degree of attention to the data at the edge of the PV power plant, which can effectively carry out visual identification of the PV area from the image.

The optimized U-Network semantic segmentation model proposed here uses residual U-shaped module composed of dilated convolution and residual network, which can build deeper structures and obtain multi-scale features under reasonable computation. The model extract edge features through feature joining, avoiding the problems of gradient disappearance and degradation in neural network, and alleviating the problem of small receptive field and inability to capture global information in simple convolution. This can effectively achieve accurate identification of the photovoltaic area. The introduction of residual learning enhances feature extraction ability of the model, enabling the model to adapt to different power station scenarios and environmental disturbances, thus effectively performing visual recognition of the PV area.

Based on the above improvements, the proposed algorithm not only effectively addresses the limitations of limited receptive fields and inability to capture global information inherent in simple convolutions. More importantly, it exhibits remarkable adaptability in handling PV area identification tasks under diverse power plant scenarios and environmental interferences. This is the reason why the optimized model proposed in this aper can achieve good performance.

4.2 Experimental analysis of fault positioning in PV area

The dataset of the fault positioning experiment uses the dataset in PV identification, from which 2,000 sheets are extracted to construct the training set. The PV images in datasets are shown in Figure 9.

The number of the fault and normal components is classified. The data of the target detection dataset is presented in Table 4.

The evaluation indicators are precision, recall, true positive (TP), false positive (FP), false negative (FN) and average precision. The experimental results are plotted in Figure 10.

From Figure 10, the optimized YOLOv4 has played a very high performance. The recall rate of normal components has reached 100%, and the precision has gained 99.9% by testing normal components. When testing hot spot fault components, it is found that the recall rate of hot spot fault components also reaches 99.53%, and the precision rate reaches 98.73%. In the identification of normal components, the data of FN is 0. In addition to high precision and recall, the proposed model has great advantages in terms of operation time. Models chosen for comparison include single shot multibox detector (SSD), Faster region based convolutional neural network (Faster R-CNN), YOLO, YOLOv2, YOLOv3, and YOLOv4. The experimental results are shown in Figure 11.

FIGURE 9
PV images in datasets.

10.3389/fnbot.2024.1396979

TABLE 4 Target detection dataset.

Data set	Number of pictures	Number of normal components	Number of hot spot fault components	Number of diode conducting components
Training set	2,000	99,444	3,623	322
Test set	809	36,589	3,439	422

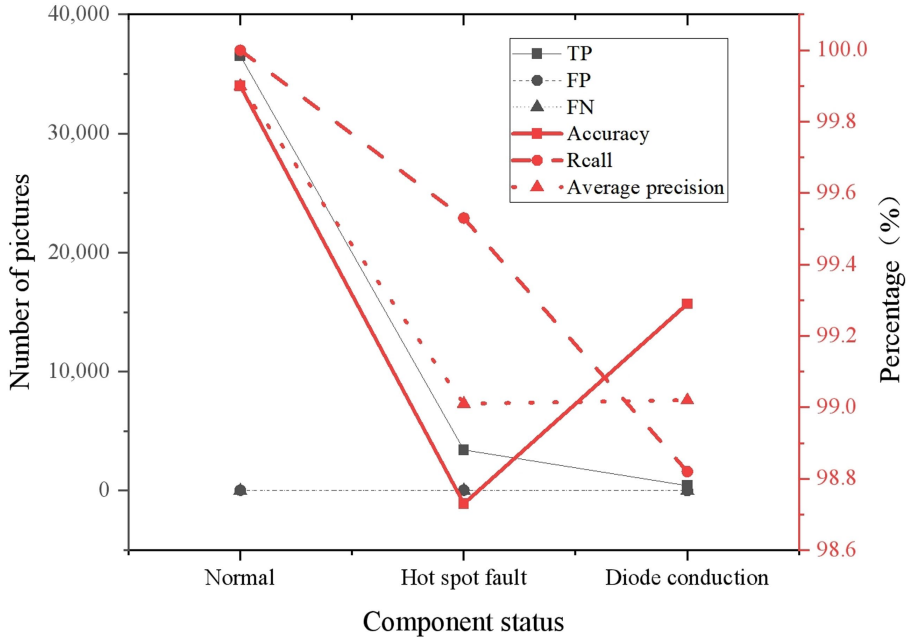


FIGURE 10
Statistics of positioning results.

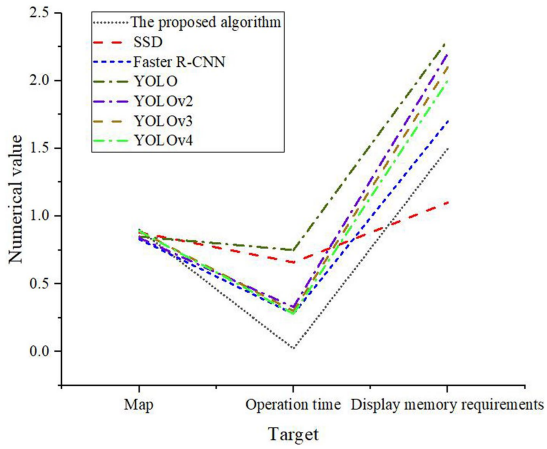


FIGURE 11
Horizontal comparison of the model.

Through the comparison of Figure 11, the average accuracy (Map) of the optimized model is the highest, at 90%. The memory requirement is 1.5G, and its operation time is only 0.022 s. This shows that the optimized YOLOv4 model has lower computational complexity and higher running speed, which can quickly process a large number of UAV infrared images, saving O&M costs and time. Through horizontal comparison, the prediction time of the proposed model is shorter than that of other traditional models, and the memory requirement is small. This shows that the optimized YOLOv4 model occupies less resources and is more suitable for deployment and running on devices with limited resources. Combined with the previous experiments, the overall accuracy and recall of the optimized model are much higher than those of the traditional model for PV identification and fault positioning. Therefore, the rationality and effectiveness of the proposed model can be verified by comparative experiments.

From the above analysis, it is evident that the optimized YOLOv4 model has a significant advantage in the fault positioning task of PV components, not only performing well in terms of precision and recall, but also having obvious optimization in prediction time and memory requirement. The optimized YOLOv4 proposed here improves the robustness and generalization ability of the model by effectively integrating features of different scales and resolutions through multi-scale fusion of spatial pyramids. This can make the model to recognize both macro and micro features in PV array images. The Complete-IOU loss function takes into account the non-overlap, center distance and aspect ratio of prediction box. This can refine the regression process for prediction boxes, and enhance localization precision and recall rates, and facilitating precise fault localization in PV systems. These improvements enable the optimized YOLOv4 model to better capture and analyze the details and features of the PV component images, thereby enhancing the performance of fault localization.

Experimental results show that the proposed PV identification model has high accuracy and IoU. In PV identification, the accuracy rate of the optimized model can reach 96.5%, and the IoU is 95.07%.

Among the six models compared, the IoU is also the highest. Moreover, the recall rate of the optimized model is only 0.08% lower than that of the Grabcut model, which has the highest recall rate. It verifies the effectiveness of the PV identification model proposed here. In the experiment of fault positioning, precision, recall, TP, FP, FN, and average precision are used as performance indicators. The fault positioning model achieves 100% recall and precision of 99.9% in testing normal components. For the hot spot fault component test, the recall rate of the hot spot fault component also reaches 99.53%, and the precision rate reaches 98.73%. The model performs well. Meanwhile, a horizontal comparison is added at the end of the experiment. Compared with the traditional model, the average accuracy value of the optimized model is the highest, 90%. The memory requirement is 1.5G, and its operation time is only 0.022 s, significantly exceeding the operation time of other models.

The experimental results demonstrate that the proposed model has high practical value and effectiveness in the O&M of real PV power plants. It can provide strong support for the safe, reliable and efficient operation of PV power plants. Specifically, the model can accurately locate faults in PV modules, helping O&M personnel to timely discover and solve problems, thereby improving the operation efficiency and long-term stability of PV power plants. For example, in the actual application of a certain PV power plant, the model successfully identified a batch of aging PV modules and replaced them in time, avoiding accidents at the power plant.

5 Conclusion

As the installed capacity of PV power generation increases rapidly, how to detect abnormalities and faults of PV modules in an efficient manner has become a key challenge to maintain the safety, reliability, and productivity of large-scale PV plants. With the consideration that all the fault information of the PV module exists in the moving images of UAVs, we propose an improved residual learning model to extract useful fault feature from the UAV moving images, and then use it for low-voltage distributed PV fault identification and positioning. This way works in an end-to-end way, and it can not only detect single faults, but also identify the existence of hybrid PV faults. First, we integrate residual learning with attention mechanism into the UAV image analysis model, aiming to improve the robustness and accuracy of PV image detection. Then, we propose a sophisticated multi-scale spatial pyramid fusion method for the optimization of the YOLOv4 network, targeting at the nuanced task of fault localization within PV arrays, where the Complete-IOU loss is used in the predictive modeling phase, significantly enhancing the accuracy and efficiency of fault detection. The proposed novel residual learning model and optimized YOLOv4 network were applied to fault identification and localization in low-voltage distributed PV systems. The models were trained and tested on a real-world dataset, demonstrating their application potential in fault detection and diagnosis of low-voltage distributed PV systems. This research is of great significance to ensure the safety, reliability and productivity of large-scale PV power plants.

The current training dataset is restricted to a limited number of fault categories, hindering direct applicability to fault identification and positioning of equipment in other types or domains. To enhance generalization capability and robustness of the proposed algorithm, we will focus on expanding the dataset to encompass a broader spectrum of fault types in future research. Moreover, considering that many novel and effective computer vision and deep learning methods emerge rapidly, we will adopt state-of-the-art intelligent models rather than the original neural network models in the target PV fault identification and positioning. In addition, we attempt to optimize the scheduling problem after fault localization using multi-objective optimization algorithms.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

XZ: Conceptualization, Formal analysis, Methodology, Validation, Writing – original draft, Writing – review & editing. YG: Formal analysis, Methodology, Validation, Writing – review & editing. YW: Data curation, Formal analysis, Validation, Writing – review & editing. JW: Formal analysis, Methodology, Validation, Writing – review & editing. WW: Data curation, Validation, Writing – review & editing. LL: Data curation, Validation, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

XZ, YG, and YW were employed by State Grid Hebei Electric Power Company. JW, WW, and LL were employed by Henan XJ Metering Co., Ltd.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Abubakar, A., Almeida, C. F. M., and Gemignani, M. (2021). Review of artificial intelligence-based failure detection and diagnosis methods for solar photovoltaic systems. *Mach. Des.* 9:328. doi: 10.3390/machines9120328

Akram, M. W., Li, G., Jin, Y., Chen, X., Zhu, C., and Ahmad, A. (2020). Automatic detection of photovoltaic module defects in infrared images with isolated and develop-model transfer deep learning. *Sol. Energy* 198, 175–186. doi: 10.1016/j.solener.2020.01.055

Alajmi, M., and Abdel-Qader, I. (2016). Fault detection and localization in solar PV arrays using the current-voltage sensing framework[C]//2016 IEEE international conference on electro information technology (EIT). *IEEE*, 0307–0312. doi: 10.1109/EIT.2016.7535257

Alajmi, M., Aljasem, O., Ali, N., Alqurashi, A., and Abdel-Qader, I., et al. (2018). Fault detection and localization in solar PV arrays framework: hybrid methods of data-analysis and a network of voltage-current sensors[C]//2018 IEEE international conference on electro/information technology (EIT). *IEEE*, 0404–0410. doi: 10.1109/EIT.2018.8500264

Ali, M. U., Khan, H. F., Masud, M., Kallu, K. D., and Zafar, A. (2020). A machine learning framework to identify the hotspot in photovoltaic module using infrared thermography. *Sol. Energy* 208, 643–651. doi: 10.1016/j.solener.2020.08.027

Alves, R. H. F., de Deus, J. G. A., Marra, E. G., et al. (2021). Automatic fault classification in PV modules using convolutional neural networks. *Renew. Energy* 179, 502–516. doi: 10.1016/j.renene.2021.07.070

Atsu, D., Seres, I., Aghaei, M., and Farkas, I. (2020). Analysis of long-term performance and reliability of PV modules under tropical climatic conditions in sub-Saharan. *Renew. Energy* 162, 285–295. doi: 10.1016/j.renene.2020.08.021

Aziz, F., Haq, A. U., Ahmad, S., Mahmoud, Y., Jalal, M., and Ali, U. (2020). A novel convolutional neural network-based approach for fault classification in photovoltaic arrays. *IEEE Access* 8, 41889–41904. doi: 10.1109/ACCESS.2020.2977116

Chen, L., and Wang, X. (2017). Adaptive fault localization in PV systems. *IEEE Trans. Smart Grid* 9, 6752–6763. doi: 10.1109/TSG.2017.2722821

Chen, S. Q., Yang, G. J., Gao, W., and Guo, M. F. (2020). Photovoltaic fault diagnosis via semisupervised ladder network with string voltage and current measures. *IEEE J. Photovolt.* 11, 219–231. doi: 10.1109/JPHOTOV.2020.3038335

Cui, F., Tu, Y., and Gao, W. (2022). A photovoltaic system fault identification method based on improved deep residual shrinkage networks. *Energies* 15:3961. doi: 10.3390/en15113961

Fadhel, S., Delpha, C., Diallo, D., Bahri, I., Migan, A., Trabelsi, M., et al. (2019). PV shading fault detection and classification based on IV curve using principal component analysis: application to isolated PV system. *Sol. Energy* 179, 1–10. doi: 10.1016/j.solener.2018.12.048

Gallardo-Saavedra, S., Hernández-Callejo, L., del Carmen, A.-G. M., Morales-Aragonés, J. I., Alonso-Gómez, V., and Martínez-Sacristán, O. (2020). Nondestructive characterization of solar PV cells defects by means of electroluminescence, infrared thermography, I–V curves and visual tests: experimental study and comparison. *Energy* 205:117930. doi: 10.1016/j.energy.2020.117930

Gao, S., Wang, K., Tao, S., Jin, T., Dai, H., and Cheng, J. (2021). A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models. *Energy Convers. Manag.* 230:113784. doi: 10.1016/j.enconman.2020.113784

Herraiz, Á. H., Marugán, A. P., and Márquez, F. P. G. (2020). PV plant condition monitoring using thermal images analysis by convolutional neural network-based structure. *Renew. Energy* 153, 334–348. doi: 10.1016/j.renene.2020.01.148

Hong, Y. Y., and Pula, R. A. (2022). Methods of PV fault detection and classification: a review. *Energy Rep.* 8, 5898–5929. doi: 10.1016/j.egyr.2022.04.043

Jie, Y., Ji, X., Yue, A., Deng, Y., Chen, J., and Zhang, Y. (2020). Combined multi-layer feature fusion and edge detection method for distributed photovoltaic power station identification. *Energies* 13:6742. doi: 10.3390/en13246742

Jin, L., Liu, L., Wang, X., Shang, M., and Wang, F. Y. (2024). Physical-informed neural network for MPC-based trajectory tracking of vehicles with noise considered. *IEEE Trans. Intellig. Vehicles*, 1–10. doi: 10.1109/TIV.2024.3358229

Khalil, I. U., Ul-Haq, A., Mahmoud, Y., Aamir, M., Ahsan, M. U., and Mehmood, K. (2020). Comparative analysis of photovoltaic faults and performance evaluation of its detection techniques. *IEEE Access* 8, 26676–26700. doi: 10.1109/ACCESS.2020.2970531

Korkmaz, D., and Acikgoz, H. (2022). An efficient fault classification method in solar PV modules using transfer learning and multi-scale convolutional neural network. *Eng. Appl. Artif. Intell.* 113:104959. doi: 10.1016/j.engappai.2022.104959

Li, N., Ma, L., Guo, Y., Xue, B., Zhang, M., and Jin, Y. (2023). Survey on evolutionary deep learning: principles, algorithms, applications and open issues. *ACM Comput. Surv.* 56, 1–34. doi: 10.1145/3603704

Li, K., Yan, J., Hu, L., Wang, F., and Zhang, N. (2021). Two-stage decoupled estimation approach of aggregated baseline load under high penetration of behind-the-meter PV system. *IEEE Trans. Smart Grid* 12, 4876–4885. doi: 10.1109/TSG.2021.3105747

Liang, J., Ge, S., Qu, B., Liu, F., Yang, H., and Li, Z. (2020a). Classified perturbation mutation base particle swarm optimization algorithm for parameters extraction of photovoltaic models. *Energy Convers. Manag.* 203:112138. doi: 10.1016/j.enconman.2019.112138

Liang, J., Qiao, K., Yu, K., Qu, B., Xu, R., and Li, K. (2020b). Parameters estimation of solar photovoltaic models via a self-adaptive ensemble-based differential evolution. *Sol. Energy* 207, 336–346. doi: 10.1016/j.solener.2020.06.100

Liu, Y., Chong, G., Heidari, A. A., Liang, G., Ye, X., and Wang, M. (2020). Horizontal and vertical crossover of Harris hawk optimizer with Nelder-Mead simplex for parameter estimation of photovoltaic models. *Energy Convers. Manag.* 223:113211. doi: 10.1016/j.enconman.2020.113211

Liufu, Y., Jin, L., Shang, M., Wang, X., and Wang, F. Y. (2024). ACP-incorporated perturbation-resistant neural dynamics controller for autonomous vehicles. *IEEE Trans. Intellig. Vehicles*, 1–12. doi: 10.1109/TIV.2023.3348632

Lyu, Y., Fairbrother, A., Gong, M., Gu, X., Kempe, M., and Boyce, K. (2020). Impact of environmental variables on the degradation of photovoltaic components and perspectives for the reliability assessment methodology. *Sol. Energy* 199, 425–436. doi: 10.1016/j.solener.2020.02.020

Ma, L., Huang, M., Yang, S., Wang, R., and Wang, X. (2022a). An adaptive localized decision variable analysis approach to large-scale multiobjective and many-objective optimization. *IEEE Trans. Cybernet.* 52, 6684–6696. doi: 10.1109/TCYB.2020.3041212

Ma, L., Li, N., Guo, Y., Geng, X., Cheng, S., Wang, X., et al. (2023a). Pareto-wise ranking classifier for multi-objective evolutionary neural architecture search. *IEEE Trans. Evol. Comput.*:1. doi: 10.1109/TEVC.2023.3314766

Ma, L., Li, N., Guo, Y., Huang, M., Yang, S., Wang, X., et al. (2022b). Learning to optimize: reference vector reinforcement learning adaption to constrained many-objective optimization of industrial copper burdening system. *IEEE Trans. Cybernet.* 52, 12698–12711. doi: 10.1109/TCYB.2021.3086501

Ma, L., Liu, Y., Guo, Y., Wang, X., Mo, H., Wang, G., et al. (2023b). Decomposition-based multiobjective optimization for variable-length mixed-variable Pareto optimization and its application in cloud service allocation. *IEEE Trans. Syst. Man. Cybernet. Syst.* 53, 7138–7151. doi: 10.1109/TSMC.2023.3295371

Maka, A. O. M., Salem, S., and Mehmood, M. (2021). Solar photovoltaic (PV) applications in Libya: challenges, potential, opportunities and future perspectives. *Clean. Eng. Technol.* 5:100267. doi: 10.1016/j.clet.2021.100267

Manno, D., Cipriani, G., Ciulla, G., Di, D. V., Guarino, S., and Brano, V. L. (2021). Deep learning strategies for automatic fault diagnosis in photovoltaic systems by thermographic images. *Energy Convers. Manag.* 241:114315. doi: 10.1016/j.enconman.2021.114315

Marqusee, J., Becker, W., and Ericson, S. (2021). Resilience and economics of microgrids with PV, battery storage, and networked diesel generators. *Adv. Appl. Energy* 3:100049. doi: 10.1016/j.adapen.2021.100049

Mellit, A., and Kalogirou, S. (2021). Artificial intelligence and internet of things to improve efficacy of diagnosis and remote sensing of solar photovoltaic systems: challenges, recommendations and future directions. *Renew. Sust. Energ. Rev.* 143:110889. doi: 10.1016/j.rser.2021.110889

Navid, Q., Hassan, A., Fardoun, A. A., Ramzan, R., and Alraeesi, A. (2021). Fault diagnostic methodologies for utility-scale photovoltaic power plants: a state of the art review. *Sustain. For.* 13:1629. doi: 10.3390/su13041629

Patil, M., and Hinge, T. (2019). Improved fault detection and location scheme for PV system[C]//2019 innovations in power and advanced computing technologies (i-PACT). *IEEE* 1, 1–6. doi: 10.1109/i-PACT44901.2019.8960246

Qais, M. H., Hasanien, H. M., and Alghuwainem, S. (2020). Parameters extraction of three-diode photovoltaic model using computation and Harris hawks optimization. *Energy* 195:117040. doi: 10.1016/j.energy.2020.117040

Quiles, E., Roldán-Blay, C., Escrivá-Escrivá, G., and Roldán-Porta, C. (2020). Accurate sizing of residential stand-alone photovoltaic systems considering system reliability. *Sustain. For.* 12:1274. doi: 10.3390/su12031274

Ramadan, A., Kamel, S., Korashy, A., and Yu, J. (2020). Photovoltaic cells parameter estimation using an enhanced teaching–learning-based optimization algorithm. *Iran. J. Sci. Technol. Trans. Electr. Eng.* 44, 767–779. doi: 10.1007/s40998-019-00257-9

Ridha, H. M., Heidari, A. A., Wang, M., and Chen, H. (2020). Boosted mutation-based Harris hawks optimizer for parameters identification of single-diode solar cell models. *Energy Convers. Manag.* 209:112660. doi: 10.1016/j.enconman.2020.112660

Ridha, H. M., Hizam, H., Gomes, C., Chen, H., Ahmadipour, M., and Alghrairi, M. (2021). Parameters extraction of three diode photovoltaic models using boosted LSHADE algorithm and Newton Raphson method. *Energy* 224:120136. doi: 10.1016/j.energy.2021.120136

Romero-Fiances, I., Livera, A., Theristis, M., Stein, S., Nofuentes, G., and Georghiou, G. E. (2022). Impact of duration and missing data on the long-term photovoltaic degradation rate estimation. *Renew. Energy* 181, 738–748. doi: 10.1016/j.renene.2021.09.078

Stiubiener, U., Carneiro da Silva, T., Trigoso, F. B. M., Benedito, R. S., and Teixeira, J. C. (2020). PV power generation on hydro dam's reservoirs in Brazil: a way to improve operational flexibility. *Renew. Energy* 150, 765–776. doi: 10.1016/j.renene.2020.01.003

Tsanakas, J. A., van der Heide, A., Radavičius, T., Lemaire, E., Wang, K., and Voroshazi, E. (2020). Towards a circular supply chain for PV modules: review of today's challenges in PV recycling, refurbishment and re-certification. *Prog. Photovolt. Res. Appl.* 28, 454–464. doi: 10.1002/pip.3193

Veerasamy, V., Wahab, N. I. B., Othman, M. L., Sekar, K., Ramachandran, R., and Islam, M. Z. (2021). LSTM recurrent neural network classifier for high impedance fault detection in solar PV integrated power system. *IEEE Access* 9, 32672–32687. doi: 10.1109/ACCESS.2021.3060800

Wang, H., Liu, R., Ding, S. X., Hu, Q., Li, Z., and Zhou, H. (2024). Causal-trivial attention graph neural network for fault diagnosis of complex industrial processes. *IEEE Trans. Industr. Inform.* 20, 1987–1996. doi: 10.1109/TII.2023.3282979

Yang, Y., Ma, C., Lian, C., Zhang, Y., and Pang, X. (2021). Optimal power reallocation of large-scale grid-connected photovoltaic power station integrated with hydrogen production. *J. Clean. Prod.* 298, 126830–126835. doi: 10.1016/j.jclepro.2021.126830

Zghaibeh, M., Okonkwo, P. C., Belgacem, I. B., Beitelmal, W. H., and Mansir, I. B. (2022). Analytical model for a techno-economic assessment of green hydrogen production in photovoltaic power station case study Salalah city-Oman. *Int. J. Hydrog. Energy* 47, 14171–14179. doi: 10.1016/j.ijhydene.2022.02.180

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 22 May 2024
DOI 10.3389/fnbot.2024.1406604

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Chenfu Yi,
Guangdong Polytechnic Normal University, China
Vasilios N. Katsikis,
National and Kapodistrian University of Athens, Greece

*CORRESPONDENCE
Binbin Qiu
✉ qiubb6@mail.sysu.edu.cn

RECEIVED 25 March 2024
ACCEPTED 03 May 2024
PUBLISHED 22 May 2024

CITATION
Zhang Y, Han Y and Qiu B (2024) An adaptive discretized RNN algorithm for posture collaboration motion control of constrained dual-arm robots.
Front. Neurorobot. 18:1406604.
doi: 10.3389/fnbot.2024.1406604

An adaptive discretized RNN algorithm for posture collaboration motion control of constrained dual-arm robots

Yichen Zhang, Yu Han and Binbin Qiu*

School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China

Although there are many studies on repetitive motion control of robots, few schemes and algorithms involve posture collaboration motion control of constrained dual-arm robots in three-dimensional scenes, which can meet more complex work requirements. Therefore, this study establishes the minimum displacement repetitive motion control scheme for the left and right robotic arms separately. On the basis of this, the design mentality of the proposed dual-arm posture collaboration motion control (DAPCMC) scheme, which is combined with a new joint-limit conversion strategy, is described, and the scheme is transformed into a time-variant equation system (TVES) problem form subsequently. To address the TVES problem, a novel adaptive Taylor-type discretized recurrent neural network (ATT-DRNN) algorithm is devised, which fundamentally solves the problem of calculation accuracy which cannot be balanced well with the fast convergence speed. Then, stringent theoretical analysis confirms the dependability of the ATT-DRNN algorithm in terms of calculation precision and convergence rate. Finally, the effectiveness of the DAPCMC scheme and the excellent convergence competence of the ATT-DRNN algorithm is verified by a numerical simulation analysis and two control cases of dual-arm robots.

KEYWORDS

dual-arm robot, dual-arm posture collaboration motion control (DAPCMC), time-variant equation system (TVES), adaptive Taylor-type discretized recurrent neural network (ATT-DRNN), joint-limit conversion strategy

1 Introduction

With the continuous development of electronic information technology, robots, as a key carrier in the realm of artificial intelligence, have been assuming a progressively substantial role in manufacturing (Arents and Greitans, 2022), healthcare (Khan et al., 2020), service industries (McCartney and McCartney, 2020), and beyond (Cheng et al., 2023; Tanyıldızı, 2023; Yang et al., 2023; Liufu et al., 2024), bringing numerous conveniences to human life and work. Many scholars are focusing their attention on robotics research field.

A robotic arm is a mechanical device composed of multiple linked joints, typically equipped with various end-effectors based on the requirements of the work environment. By calculating and adjusting the rotational changes of each joint, the end-effector can be controlled to perform various movements in a predetermined manner, such as position and orientation, thereby accomplishing tasks. For instance, the MATLAB program and particle swarm optimization were utilized for the trajectory planning of the robotic arm (Ekrem and Aksoy, 2023); Chico et al. (2021) employed a hand gesture recognition system and the inertial measurement unit to control the position and orientation of a virtual robotic arm.

A target admittance model was designed in the joint space for hands-on procedures that can be applied in all commercially available general-purpose robotic arms with six or more DOF (Kastritsi and Doulgeri, 2021).

Due to the escalating complexity of task environments, single robotic arms frequently encounter challenges in effectively completing tasks, which highlights the advantages of dual robotic arms in collaborative and efficient task execution. For example, Jiang et al. (2022) presented an adaptive control method for a dual-arm robot to perform bimanual tasks under modeling uncertainties. Bombile and Billard (2022) designed a unified motion generation algorithm that enables a dual-arm robot to grab and release objects quickly. Wang et al. (2023) proposed a sliding mode controller with good robustness against the model uncertainties to capture and stabilize a spinning target in 3D space by a dual-arm space robot.

However, some of the methods mentioned above do not take into account the actual physical constraints of the robotic arms during initial modeling (e.g., Bombile and Billard, 2022; Jiang et al., 2022). This greatly limits the application scenarios of these algorithms and is inconsistent with the real working conditions of the robotic arms. Furthermore, the physical limitations of robotic arms typically pertain to constraints on joint angle and velocity. These constraints do not reside at the same constraint level, thus there are substantial computational challenges when attempting to address them collectively. An optimal approach entails a series of conversion strategies to harmonize these distinct hierarchical constraints to a congruous level (Zhang and Zhang, 2013) (e.g., velocity level). By implementing this approach, the constraints can be effectively unified and dealt without compromising their intended meaning. Some scholars (e.g., Li, 2020) have crafted novel approaches to these conversion strategies stemming from this foundation. Nevertheless, in the process, they have introduced too many supplementary parameters, rendering the strategies less straightforward for apprehension. Additionally, certain studies focus on the control of dual robotic arms based on 2D space, considerably limiting the operating range of robotic arms (Stolfi et al., 2017; Yang S. et al., 2020; Yang et al., 2021).

In recent years, with the rapid advancement of neural network research, many scholars have been committed to applying its formidable nonlinear modeling capability and efficient parallel computing ability to the domain of robotic arm motion control (Wang et al., 2021; Jin et al., 2024). This endeavor has given rise to a special kind of neural network known as the RNN (Xiao et al., 2021; Yan et al., 2022; Fu et al., 2023). For example, Xiao et al. (2021) proposed a noise-enduring and finite-time convergent design formula is suggested to establish a novel RNN. Fu et al. (2023) presented a gradient-feedback RNN to solve the unconstrained time-variant convex optimization problem.

To facilitate the calculation on computers and other digital hardware devices, some scholars focus on discretizing conventional CRNN models through time discretization techniques, leading to the development of DRNN algorithms (Liao et al., 2016; Liu et al., 2023a,b; Shi et al., 2023). The technique of second-order Taylor expansion was used to deal with the discrete time-variant nonlinear system, and a DRNN algorithm was proposed subsequently (Shi et al., 2023). Liao et al. (2016) proposed two Taylor-type DRNN algorithms on account of the Taylor-type formula to perform online dynamic equality-constrained quadratic programming. Liu et al. (2023a) designed a Taylor-type DRNN algorithm based on Taylor-type discrete scheme with smaller TE. It is worth noting that higher accuracy requirements often make the discretization formulas more complicated, inevitably leading to a large amount of computation and increasing the cost of actual production applications. After overall consideration, this study proposes an adaptive DRNN algorithm based on a three-step general Taylor-type discretization formula with an adaptive sampling period introduced, which is of high enough precision for practical applications.

Typically, due to the use of fixed sampling periods and fixed convergence factors in the conventional DRNN algorithms mentioned above, it is difficult for them to achieve a balance in computational precision and convergence rate, resulting in limited algorithmic dynamic and convergence performance. Therefore, some researchers have tried to introduce various adaptive mechanisms into model/algorithm design (Song et al., 2008; Yang M. et al., 2020; Dai et al., 2022; Cai and Yi, 2023). For example, Yang M. et al. (2020) proposed two discretized RNN algorithms with an adaptive Jacobian matrix. Cai and Yi (2023) developed an adaptive gradient-descent-based RNN model to solve time-variant problems based on the Lyapunov theory. Dai et al. (2022) proposed a hybrid RNN model by introducing a fuzzy adaptive control strategy to generate a fuzzy adaptive factor that can change its size adaptively according to the RE. Song et al. (2008) proposed a robust adaptive gradient-descent training algorithm based on an RNN hybrid training concept in discrete-time domain.

In light of the aforementioned circumstances, this study formulates a DAPCMC scheme in 3D space based on the dual-arm robot system and the new JLCS. Subsequently, a novel ATT-DRNN algorithm with adaptive sampling period and adaptive convergence factor is devised to effectively face the challenge of achieving a dynamic balance between great computational precision and rapid convergence rate. When compared with the CTT-DRNN algorithm and the CET-DRNN algorithm, the proposed ATT-DRNN algorithm demonstrates outstanding computational precision and rapid convergence rate. To demonstrate the features and strengths of the proposed ATT-DRNN algorithm, Table 1 shows the comparisons among distinct methods for the motion control of robots.

The remainder of this study consists of four parts. Section 2 formulates the DAPCMC scheme and designs the ATT-DRNN algorithm. Section 3 presents the theoretical analyses of the

Abbreviations: 3D, Three dimensional; 2D, Two dimensional; RNN, Recurrent neural network; TE, Truncation error; CRNN, Continuous recurrent neural network; DRNN, Discretized recurrent neural network; DAPCMC, Dual-arm posture collaboration motion control; JLCS, Joint-limit conversion strategy; ATT-DRNN, Adaptive Taylor-type discretized recurrent neural network; CTT-DRNN, Conventional Taylor-type discretized recurrent neural network; CET-DRNN, Conventional Euler-type discretized recurrent neural network; TVES, Time-variant equation system; MDRMC, Minimum displacement repetitive motion control; DOF, Degrees of freedom; LA, Left arm; RA, Right arm; TVQP, Time-variant quadratic programming; EE, Error equation; ACRNN, Adaptive continuous recurrent neural network; RE, Residual error; D-H, Denavit-Hartenberg; UAV, Unmanned aerial vehicle.

TABLE 1 Comparisons among distinct methods for motion control of robots.

Method	Posture control	Inequality constraint	Discretized handling	Adaptive mechanism	Applicable scene	Robotic arm number
Jiang et al. (2022)	No	No	No	Yes	2D	Dual
Yang et al. (2021)	Yes	Yes	Yes	No	2D	Dual
Yang S. et al. (2020)	No	Yes	No	No	2D	Dual
Fu et al. (2023)	No	No	No	No	3D	Single
Shi et al. (2023)	No	No	Yes	No	2D	Single
Liao et al. (2016)	No	No	Yes	No	2D	Single
Wu and Zhang (2023)	Yes	No	Yes	Yes	3D	Single
Yang M. et al. (2020)	No	No	Yes	Yes	3D	Single
ATT-DRNN	Yes	Yes	Yes	Yes	3D	Dual

proposed ATT-DRNN algorithm. Section 4 provides illustrative examples, and Section 5 concludes this study. Finally, the primary contributions/novelties of this paper can be summarized as follows.

1) Distinguishing from common dual-arm robot motion control schemes in 2D space, a novel construction methodology of the DAPCMC scheme in 3D space is provided, which can make a spatial dual-arm robot collaboratively execute repetitive tracking of a desired trajectory while adhering to a predetermined posture.
2) Distinguishing from existing strategies, an innovative JLCS is proposed, which has a ubiquitously differentiable and more succinct expression.
3) Distinguishing from conventional discretization methods, an innovative ATT-DRNN algorithm is engineered to address the DAPCMC scheme, which introduces a new adaptive convergence factor and sampling period to guarantee a notable convergence rate and exceptional convergence precision.
4) Distinguishing from the simple path-tracking task of single-arm robots, the posture collaboration motion control experiments of a UR5 dual-arm robot with the joint-angle and joint-velocity bound constraints considered substantiate the effectiveness of the proposed DAPCMC scheme and the outstanding convergence capability of the proposed ATT-DRNN algorithm.

2 Scheme formulation and algorithm design

This section describes how to construct a DAPCMC scheme that can be converted into a TVES problem and processed by the proposed ATT-DRNN algorithm.

2.1 Rudimentary knowledge

For the convenience of comprehension, let us construct a single robot arm motion control scheme with n DOF, which takes into account joint physical limits and can simultaneously ensure position control and orientation control during the MDRMC. Specifically, such a scheme can be described as below:

$$\min_{\dot{\mathfrak{z}}(t)} \quad \frac{1}{2}\dot{\mathfrak{z}}^{\mathrm{T}}(t)\boldsymbol{U}(t)\dot{\mathfrak{z}}(t)+\boldsymbol{\varphi}^{\mathrm{T}}(t)\dot{\mathfrak{z}}(t), \tag{1}$$

$$\text{s.t.} \quad \boldsymbol{J}_1(\mathfrak{z}(t))\dot{\mathfrak{z}}(t)=\dot{\boldsymbol{\Upsilon}}_{\mathrm{I}}(t)-\alpha\left[\boldsymbol{\Upsilon}_{\mathrm{R}}(t)-\boldsymbol{\Upsilon}_{\mathrm{I}}(t)\right], \tag{2}$$

$$\boldsymbol{J}_2(\mathfrak{z}(t))\dot{\mathfrak{z}}(t)=\dot{\boldsymbol{o}}_{\mathrm{I}}(t)-\beta\left[\boldsymbol{o}_{\mathrm{R}}(t)-\boldsymbol{o}_{\mathrm{I}}(t)\right], \tag{3}$$

$$\mathfrak{z}^{-}\le\mathfrak{z}(t)\le\mathfrak{z}^{+}, \tag{4}$$

$$\dot{\mathfrak{z}}^{-}\le\dot{\mathfrak{z}}(t)\le\dot{\mathfrak{z}}^{+}, \tag{5}$$

where superscript $^{\mathrm{T}}$ represents the transpose operator; $\mathfrak{z}(t)=\left[\mathfrak{z}_1(t),\mathfrak{z}_2(t),...,\mathfrak{z}_n(t)\right]^{\mathrm{T}}\in\mathbb{R}^n$ indicates the angle values of the robotic joints, and $\dot{\mathfrak{z}}(t)\in\mathbb{R}^n$ means the angular velocities of the robotic joints; matrix $\boldsymbol{U}(t)=\boldsymbol{I}_{n\times n}\in\mathbb{R}^{n\times n}$ is an identity matrix; vector $\boldsymbol{\varphi}(t)=\xi\left[\mathfrak{z}(t)-\mathfrak{z}(0)\right]\in\mathbb{R}^n$ with design parameter $\xi>0$ and $\mathfrak{z}(0)$ means the initial joint-angle vector; $\boldsymbol{J}_1(\mathfrak{z}(t))\in\mathbb{R}^{3\times n}$ and $\boldsymbol{J}_2(\mathfrak{z}(t))\in\mathbb{R}^{3\times n}$ represent the position Jacobian matrix and the orientation Jacobian matrix, respectively; $\boldsymbol{\Upsilon}_{\mathrm{I}}(t)\in\mathbb{R}^3$ and $\boldsymbol{\Upsilon}_{\mathrm{R}}(t)\in\mathbb{R}^3$ represent the ideal path and the real position of the end-executor, separately; $\boldsymbol{o}_{\mathrm{I}}(t)\in\mathbb{R}^3$ and $\boldsymbol{o}_{\mathrm{R}}(t)\in\mathbb{R}^3$ represent the ideal orientation and the real orientation of the end-executor, respectively; $\alpha>0$ and $\beta>0$ are both the error-feedback gains; $\mathfrak{z}^{\pm}$ and $\dot{\mathfrak{z}}^{\pm}$ denote the upper and lower limits of $\mathfrak{z}(t)$ and $\dot{\mathfrak{z}}(t)$, separately.

Remark 2.1: In accordance with previous experience (Zhang and Zhang, 2013), when $t\to\infty$, the objective function (1) at the joint-velocity level is equivalent to $\left\|\mathfrak{z}(t)-\mathfrak{z}(0)\right\|_2^2/2$ at the joint-angle level, where the design parameter $\xi>0$ ought to be adjusted as large as allowed by the manipulator conditions. Note that the robot arm's repetitive motion planning scheme under minimal displacement can be regarded as an optimization objective that can be resolved at the joint-velocity level.

Remark 2.2: Referring to the contributions of previous scholars (Yang et al., 2021), the equality constraint (2) at the joint-velocity level is equivalent to $\boldsymbol{f}(\mathfrak{z}(t))=\boldsymbol{\Upsilon}_{\mathrm{I}}(t)$ at the joint-angle level, when $t\to\infty$ and the error-feedback gain $\alpha>0$ is at an appropriate value, where $\boldsymbol{f}(\cdot):\mathbb{R}^n\to\mathbb{R}^3$ represents the forward kinematics mapping function of a robotic arm.

Remark 2.3: Similarly, the equality constraint (3) at the joint-velocity level is equivalent to $\boldsymbol{g}(\mathfrak{z}(t))=\boldsymbol{o}_{\mathrm{I}}(t)$ at the joint-angle

level, when $t \to \infty$ and the error-feedback gain $\beta > 0$ is at an appropriate value, where nonlinear function $\boldsymbol{g}(\boldsymbol{z}(t)) = \boldsymbol{o}_{\mathrm{R}}(t) = \left[o_{\mathrm{Rx}}(t), o_{\mathrm{Ry}}(t), o_{\mathrm{Rz}}(t)\right]^{\mathrm{T}} \in \mathbb{R}^3$ and the 2-norm of the real orientation vector $\boldsymbol{o}_{\mathrm{R}}(t)$ satisfies $\left\|\boldsymbol{o}_{\mathrm{R}}(t)\right\|_2 = 1$.

Note that the inequality constraint (4) is at the joint-angle level of the system. In order to integrate inequality constraints (4) and (5) of distinct constraint levels into a unified formulation at the joint-velocity level as below:

$$\mho^{-}(t) \le \dot{\boldsymbol{z}}(t) \le \mho^{+}(t), \tag{6}$$

previous studies (Zhang and Zhang, 2013; Zhang et al., 2018; Li, 2020; Li et al., 2023; Qiu et al., 2023) supply a large number of JLCSs.

Nevertheless, the JLCS in Zhang and Zhang (2013) is unable to guarantee $\mho^{-}(t)$ or $\mho^{+}(t)$ to be differentiable anywhere. Meanwhile, as regard to the JLCS in Li (2020), $\mho^{-}(t)$ and $\mho^{+}(t)$ are designed as piecewise functions, respectively, and complex compound functions are embedded in them. In addition, the JLCS in Li et al. (2023); Qiu et al. (2023) adopt numerous design parameters and construct pretty complex expressions.

Therefore, as one of the contributions of this study, we provide a new JLCS. The ith ($i = 1, 2, ..., n$) elements of $\mho^{-}(t)$ and $\mho^{+}(t)$ in (6) are designed as follows:

$$\begin{cases} \mho_i^{-}(t) = \dot{\boldsymbol{z}}_i^{-} \exp\left[\dfrac{\gamma \dot{\boldsymbol{z}}_i(t)}{\dot{\boldsymbol{z}}_i(t) - \dot{\boldsymbol{z}}_i^{-} + \varepsilon_1}\right], \varepsilon_1 \to 0_{+}, & (7) \\ \mho_i^{+}(t) = \dot{\boldsymbol{z}}_i^{+} \exp\left[\dfrac{\gamma \dot{\boldsymbol{z}}_i(t)}{\dot{\boldsymbol{z}}_i(t) - \dot{\boldsymbol{z}}_i^{+} + \varepsilon_2}\right], \varepsilon_2 \to 0_{-}, & (8) \end{cases}$$

where $\dot{\boldsymbol{z}}_i(t), \dot{\boldsymbol{z}}_i^{-}, \dot{\boldsymbol{z}}_i^{+}, \dot{\boldsymbol{z}}_i^{-}, \dot{\boldsymbol{z}}_i^{+}$ denote the ith element of $\dot{\boldsymbol{z}}(t), \dot{\boldsymbol{z}}^{-}, \dot{\boldsymbol{z}}^{+}, \dot{\boldsymbol{z}}^{-}, \dot{\boldsymbol{z}}^{+}$ in (4) and (5), separately; ε_1 and ε_2 are both non-zero minimum terms to ensure that the above equations are able to differentiable everywhere; design parameter $\gamma \in (0, 1)$ should be as small as possible.

Remark 2.4: To present the proposed JLCS (7)-(8) more specifically, Figure 1 exhibits the relationship between the ith joint angle $\dot{\boldsymbol{z}}_i(t)$ and the ith joint velocity $\dot{\boldsymbol{z}}_i(t)$. It is worth noting that, when the joint approaches its lower or upper limit, the value of γ has a crucial effect on the changing rate of the joint-velocity boundary.

2.2 DAPCMC scheme

Finally, upon the previous section, we construct a dual-arm collaborative control system consisting of the LA and RA.

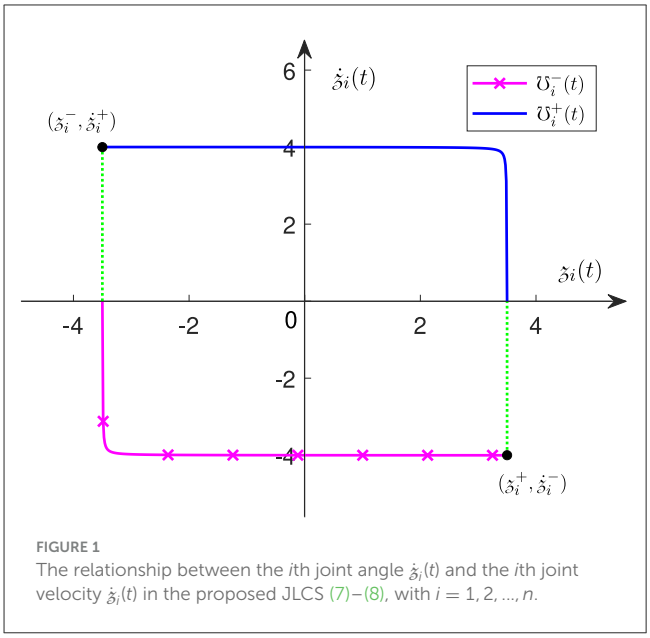


FIGURE 1
The relationship between the ith joint angle $\dot{\boldsymbol{z}}_i(t)$ and the ith joint velocity $\dot{\boldsymbol{z}}_i(t)$ in the proposed JLCS (7)–(8), with $i = 1, 2, ..., n$.

2.2.1 LA collaborative control subsystem

According to (1)–(5), we construct the MDRMC scheme for the n-DOF LA as follows:

$$\min_{\dot{\boldsymbol{z}}_{\mathcal{L}}(t)} \frac{1}{2}\dot{\boldsymbol{z}}_{\mathcal{L}}^{\mathrm{T}}(t)\boldsymbol{U}_{\mathcal{L}}(t)\dot{\boldsymbol{z}}_{\mathcal{L}}(t) + \boldsymbol{\varphi}_{\mathcal{L}}^{\mathrm{T}}(t)\dot{\boldsymbol{z}}_{\mathcal{L}}(t), \tag{9}$$

$$\text{s.t.} \boldsymbol{J}_{1\mathcal{L}}(\boldsymbol{z}_{\mathcal{L}}(t))\dot{\boldsymbol{z}}_{\mathcal{L}}(t) = \dot{\boldsymbol{\Upsilon}}_{1\mathcal{L}}(t) - \alpha_{\mathcal{L}}\left[\boldsymbol{\Upsilon}_{\mathrm{R}\mathcal{L}}(t) - \boldsymbol{\Upsilon}_{1\mathcal{L}}(t)\right], \tag{10}$$

$$\boldsymbol{J}_{2\mathcal{L}}(\boldsymbol{z}_{\mathcal{L}}(t))\dot{\boldsymbol{z}}_{\mathcal{L}}(t) = \dot{\boldsymbol{o}}_{1\mathcal{L}}(t) - \beta_{\mathcal{L}}\left[\boldsymbol{o}_{\mathrm{R}\mathcal{L}}(t) - \boldsymbol{o}_{1\mathcal{L}}(t)\right], \tag{11}$$

$$\boldsymbol{z}_{\mathcal{L}}^{-} \le \boldsymbol{z}_{\mathcal{L}}(t) \le \boldsymbol{z}_{\mathcal{L}}^{+}, \tag{12}$$

$$\dot{\boldsymbol{z}}_{\mathcal{L}}^{-} \le \dot{\boldsymbol{z}}_{\mathcal{L}}(t) \le \dot{\boldsymbol{z}}_{\mathcal{L}}^{+}, \tag{13}$$

where the subscript $\mathcal{L}$ denotes the LA; vector $\boldsymbol{z}_{\mathcal{L}}(t) = \left[\dot{\boldsymbol{z}}_{\mathcal{L}1}(t), \dot{\boldsymbol{z}}_{\mathcal{L}2}(t), ..., \dot{\boldsymbol{z}}_{\mathcal{L}n}(t)\right]^{\mathrm{T}} \in \mathbb{R}^n$; vector $\boldsymbol{\varphi}_{\mathcal{L}}(t) = \xi_{\mathcal{L}}\left[\boldsymbol{z}_{\mathcal{L}}(t) - \boldsymbol{z}_{\mathcal{L}}(0)\right] \in \mathbb{R}^n$ with the design parameter $\xi_{\mathcal{L}} > 0$ and $\boldsymbol{z}_{\mathcal{L}}(0)$ means the LA's initial joint-angle vector. Additionally, the meanings represented by the other symbols are similar to those in the MDRMC scheme (1)–(5).

Moreover, according to the JLCS (7)–(8), (12) and (13) in the LA's MDRMC scheme can be converted into the following form:

$$\mho_{\mathcal{L}}^{-}(t) \le \mho_{\mathcal{L}}(t) \le \mho_{\mathcal{L}}^{+}(t), \tag{14}$$

where $\mho_{\mathcal{L}}(t) = \dot{\boldsymbol{z}}_{\mathcal{L}}(t) = \left[\dot{\boldsymbol{z}}_{\mathcal{L}1}(t), \dot{\boldsymbol{z}}_{\mathcal{L}2}(t), , ..., \dot{\boldsymbol{z}}_{\mathcal{L}n}(t)\right]^{\mathrm{T}} \in \mathbb{R}^n$; the upper and lower limit values of $\mho_{\mathcal{L}}^{-}(t)$ and $\mho_{\mathcal{L}}^{+}(t)$ are correspondence to those of $\mho^{-}(t)$ and $\mho^{+}(t)$ in the JLCS (7)-(8).

Furthermore, by reorganizing the LA's MDRMC scheme (9)-(13), we obtain the LA collaborative control subsystem scheme, which has a briefer representation:

$$\min_{\mho_{\mathcal{L}}(t)} \frac{1}{2}\mho_{\mathcal{L}}^{\mathrm{T}}(t)\boldsymbol{U}_{\mathcal{L}}(t)\mho_{\mathcal{L}}(t) + \boldsymbol{\varphi}_{\mathcal{L}}^{\mathrm{T}}(t)\mho_{\mathcal{L}}(t), \tag{15}$$

$$\text{s.t.} \quad \boldsymbol{A}_{\mathcal{L}}(t)\mho_{\mathcal{L}}(t) = \boldsymbol{c}_{\mathcal{L}}(t), \tag{16}$$

$$\boldsymbol{B}_{\mathcal{L}}(t)\mho_{\mathcal{L}}(t) \le \boldsymbol{d}_{\mathcal{L}}(t), \tag{17}$$

where matrices $\boldsymbol{A}_{\mathcal{L}}(t) \in \mathbb{R}^{6\times n}$ and $\boldsymbol{B}_{\mathcal{L}}(t) \in \mathbb{R}^{2n\times n}$ and vectors $\boldsymbol{c}_{\mathcal{L}}(t) \in \mathbb{R}^{6}$ and $\boldsymbol{d}_{\mathcal{L}}(t) \in \mathbb{R}^{2n}$ are expressed as below:

$$\boldsymbol{A}_{\mathcal{L}}(t) = \begin{bmatrix} \boldsymbol{J}_{1\mathcal{L}}(\boldsymbol{\jmath}_{\mathcal{L}}(t)) \\ \boldsymbol{J}_{2\mathcal{L}}(\boldsymbol{\jmath}_{\mathcal{L}}(t)) \end{bmatrix},$$

$$\boldsymbol{c}_{\mathcal{L}}(t) = \begin{bmatrix} \dot{\boldsymbol{\Upsilon}}_{\mathrm{I}\mathcal{L}}(t) - \alpha_{\mathcal{L}}\left[\boldsymbol{\Upsilon}_{\mathrm{R}\mathcal{L}}(t) - \boldsymbol{\Upsilon}_{\mathrm{I}\mathcal{L}}(t)\right] \\ \dot{\boldsymbol{o}}_{\mathrm{I}\mathcal{L}}(t) - \beta_{\mathcal{L}}\left[\boldsymbol{o}_{\mathrm{R}\mathcal{L}}(t) - \boldsymbol{o}_{\mathrm{I}\mathcal{L}}(t)\right] \end{bmatrix}$$

$$\boldsymbol{B}_{\mathcal{L}}(t) = \begin{bmatrix} \boldsymbol{I}_{n\times n} \\ -\boldsymbol{I}_{n\times n} \end{bmatrix}, \qquad \boldsymbol{d}_{\mathcal{L}}(t) = \begin{bmatrix} \upsilon_{\mathcal{L}}^{+}(t) \\ -\upsilon_{\mathcal{L}}^{-}(t) \end{bmatrix}.$$

2.2.2 RA collaborative control subsystem

Similar to the (2.2.1), the MDRMC scheme for the n-DOF RA is follows:

$$\min_{\dot{\jmath}_{\mathcal{R}}(t)} \frac{1}{2}\dot{\jmath}_{\mathcal{R}}^{\mathrm{T}}(t)\boldsymbol{U}_{\mathcal{R}}(t)\dot{\jmath}_{\mathcal{R}}(t) + \boldsymbol{\varphi}_{\mathcal{R}}^{\mathrm{T}}(t)\dot{\jmath}_{\mathcal{R}}(t), \tag{18}$$

$$\text{s.t.} \boldsymbol{J}_{1\mathcal{R}}(\jmath_{\mathcal{R}}(t))\dot{\jmath}_{\mathcal{R}}(t) = \dot{\boldsymbol{\Upsilon}}_{\mathrm{I}\mathcal{R}}(t) - \alpha_{\mathcal{R}}\left[\boldsymbol{\Upsilon}_{\mathrm{R}\mathcal{R}}(t) - \boldsymbol{\Upsilon}_{\mathrm{I}\mathcal{R}}(t)\right], \tag{19}$$

$$\boldsymbol{J}_{2\mathcal{R}}(\jmath_{\mathcal{R}}(t))\dot{\jmath}_{\mathcal{R}}(t) = \dot{\boldsymbol{o}}_{\mathrm{I}\mathcal{R}}(t) - \beta_{\mathcal{R}}\left[\boldsymbol{o}_{\mathrm{R}\mathcal{R}}(t) - \boldsymbol{o}_{\mathrm{I}\mathcal{R}}(t)\right], \tag{20}$$

$$\jmath_{\mathcal{R}}^{-} \le \jmath_{\mathcal{R}}(t) \le \jmath_{\mathcal{R}}^{+}, \tag{21}$$

$$\dot{\jmath}_{\mathcal{R}}^{-} \le \dot{\jmath}_{\mathcal{R}}(t) \le \dot{\jmath}_{\mathcal{R}}^{+}, \tag{22}$$

where the subscript $\mathcal{R}$ denotes the RA; vector $\jmath_{\mathcal{R}}(t) = \left[\dot{\jmath}_{\mathcal{R}1}(t), \dot{\jmath}_{\mathcal{R}2}(t), ..., \dot{\jmath}_{\mathcal{R}n}(t)\right]^{\mathrm{T}} \in \mathbb{R}^{n}$; vector $\boldsymbol{\varphi}_{\mathcal{R}}(t) = \xi_{\mathcal{R}}\left[\jmath_{\mathcal{R}}(t) - \jmath_{\mathcal{R}}(0)\right] \in \mathbb{R}^{n}$ with design parameter $\xi_{\mathcal{R}} > 0$ and $\jmath_{\mathcal{R}}(0)$ means the RA's initial joint-angle vector. Additionally, the meanings represented by the other symbols are similar to those in the MDRMC scheme for the LA (9)–(13).

Similarly, (21) and (22) in the RA's MDRMC scheme can be transformed into the following form:

$$\upsilon_{\mathcal{R}}^{-}(t) \le \upsilon_{\mathcal{R}}(t) \le \upsilon_{\mathcal{R}}^{+}(t), \tag{23}$$

where $\upsilon_{\mathcal{R}}(t) = \dot{\jmath}_{\mathcal{R}}(t) = \left[\dot{\jmath}_{\mathcal{R}1}(t), \dot{\jmath}_{\mathcal{R}2}(t), , ..., \dot{\jmath}_{\mathcal{R}n}(t)\right]^{\mathrm{T}} \in \mathbb{R}^{n}$; the upper and lower limit values of $\upsilon_{\mathcal{R}}^{-}(t)$ and $\upsilon_{\mathcal{R}}^{+}(t)$ are parallelism to those of $\upsilon_{\mathcal{L}}^{-}(t)$ and $\upsilon_{\mathcal{L}}^{+}(t)$ in the LA's JLCS (14).

Then, by reorganizing the RA's MDRMC scheme (18)–(22), we obtain the RA collaborative control subsystem scheme:

$$\min_{\upsilon_{\mathcal{R}}(t)} \frac{1}{2}\upsilon_{\mathcal{R}}^{\mathrm{T}}(t)\boldsymbol{U}_{\mathcal{R}}(t)\upsilon_{\mathcal{R}}(t) + \boldsymbol{\varphi}_{\mathcal{R}}^{\mathrm{T}}(t)\upsilon_{\mathcal{R}}(t), \tag{24}$$

$$\text{s.t.} \quad \boldsymbol{A}_{\mathcal{R}}(t)\upsilon_{\mathcal{R}}(t) = \boldsymbol{c}_{\mathcal{R}}(t), \tag{25}$$

$$\boldsymbol{B}_{\mathcal{R}}(t)\upsilon_{\mathcal{R}}(t) \le \boldsymbol{d}_{\mathcal{R}}(t), \tag{26}$$

where matrices $\boldsymbol{A}_{\mathcal{R}}(t) \in \mathbb{R}^{6\times n}$ and $\boldsymbol{B}_{\mathcal{R}}(t) \in \mathbb{R}^{2n\times n}$ and vectors $\boldsymbol{c}_{\mathcal{R}}(t) \in \mathbb{R}^{6}$ and $\boldsymbol{d}_{\mathcal{R}}(t) \in \mathbb{R}^{2n}$ are expressed as follows:

$$\boldsymbol{A}_{\mathcal{R}}(t) = \begin{bmatrix} \boldsymbol{J}_{1\mathcal{R}}(\jmath_{\mathcal{R}}(t)) \\ \boldsymbol{J}_{2\mathcal{R}}(\jmath_{\mathcal{R}}(t)) \end{bmatrix},$$

$$\boldsymbol{c}_{\mathcal{R}}(t) = \begin{bmatrix} \dot{\boldsymbol{\Upsilon}}_{\mathrm{I}\mathcal{R}}(t) - \alpha_{\mathcal{R}}\left[\boldsymbol{\Upsilon}_{\mathrm{R}\mathcal{R}}(t) - \boldsymbol{\Upsilon}_{\mathrm{I}\mathcal{R}}(t)\right] \\ \dot{\boldsymbol{o}}_{\mathrm{I}\mathcal{R}}(t) - \beta_{\mathcal{R}}\left[\boldsymbol{o}_{\mathrm{R}\mathcal{R}}(t) - \boldsymbol{o}_{\mathrm{I}\mathcal{R}}(t)\right] \end{bmatrix}$$

$$\boldsymbol{B}_{\mathcal{R}}(t) = \begin{bmatrix} \boldsymbol{I}_{n\times n} \\ -\boldsymbol{I}_{n\times n} \end{bmatrix}, \qquad \boldsymbol{d}_{\mathcal{R}}(t) = \begin{bmatrix} \upsilon_{\mathcal{R}}^{+}(t) \\ -\upsilon_{\mathcal{R}}^{-}(t) \end{bmatrix}.$$

Furthermore we combine the LA collaborative control subsystem scheme (15)–(17) with the RA collaborative control subsystem scheme (24)–(26) to obtain a complete DAPCMC scheme, which is also a TVQP problem:

$$\min_{\upsilon(t)} \frac{1}{2}\upsilon^{\mathrm{T}}(t)\boldsymbol{U}(t)\upsilon(t) + \boldsymbol{\varphi}^{\mathrm{T}}(t)\upsilon(t), \tag{27}$$

$$\text{s.t.} \quad \boldsymbol{A}(t)\upsilon(t) = \boldsymbol{c}(t), \tag{28}$$

$$\boldsymbol{B}(t)\upsilon(t) \le \boldsymbol{d}(t), \tag{29}$$

where

$$\boldsymbol{A}(t) = \begin{bmatrix} \boldsymbol{A}_{\mathcal{L}}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{12\times 2n}, \ \upsilon(t) = \begin{bmatrix} \upsilon_{\mathcal{L}}(t) \\ \upsilon_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{2n},$$

$$\boldsymbol{c}(t) = \begin{bmatrix} \boldsymbol{c}_{\mathcal{L}}(t) \\ \boldsymbol{c}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{12}$$

$$\boldsymbol{B}(t) = \begin{bmatrix} \boldsymbol{B}_{\mathcal{L}}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{4n\times 2n}, \ \boldsymbol{d}(t) = \begin{bmatrix} \boldsymbol{d}_{\mathcal{L}}(t) \\ \boldsymbol{d}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{4n},$$

$$\boldsymbol{\varphi}(t) = \begin{bmatrix} \boldsymbol{\varphi}_{\mathcal{L}}(t) \\ \boldsymbol{\varphi}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{2n}$$

$$\boldsymbol{U}(t) = \begin{bmatrix} \boldsymbol{U}_{\mathcal{L}}(t) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U}_{\mathcal{R}}(t) \end{bmatrix} \in \mathbb{R}^{2n\times 2n}.$$

In order to resolve the proposed DAPCMC scheme (27)–(29), that is, to seek the optimal solution to the TVQP problem (27)–(29), it is necessary for us to concentrate on how to translate such a TVQP problem (27)–(29) into a more computationally convenient TVES problem. After that, solving the TVES problem is tantamount to finding the optimal solution to the TVQP problem (27)–(29).

With reference to Wei et al. (2022), the optimal solution to the TVQP problem (27)–(29) can be obtained by dealing with the following TVES problem:

$$\mathcal{H}(t)\boldsymbol{\chi}(t) + \boldsymbol{g}(t) = \boldsymbol{0}, \tag{30}$$

where the coefficient matrix $\mathcal{H}(t) \in \mathbb{R}^{\varpi\times\varpi}$ and the vectors $\boldsymbol{\chi}(t) \in \mathbb{R}^{\varpi}$ and $\boldsymbol{g}(t) \in \mathbb{R}^{\varpi}$ can be described as follows:

$$\mathcal{H}(t) = \begin{bmatrix} \boldsymbol{U}(t) & \boldsymbol{A}^{\mathrm{T}}(t) & \boldsymbol{B}^{\mathrm{T}}(t) \\ \boldsymbol{A}(t) & \boldsymbol{0}_{12\times 12} & \boldsymbol{0}_{12\times 4n} \\ -\boldsymbol{B}(t) & \boldsymbol{0}_{4n\times 12} & \boldsymbol{I}_{4n\times 4n} \end{bmatrix}$$

$$\boldsymbol{\chi}(t) = \begin{bmatrix} \upsilon(t) \\ \boldsymbol{\lambda}(t) \\ \boldsymbol{\mu}(t) \end{bmatrix}, \qquad \boldsymbol{g}(t) = \begin{bmatrix} \boldsymbol{\varphi}(t) \\ -\boldsymbol{c}(t) \\ \boldsymbol{r}(t) \end{bmatrix}$$

where the Lagrange multiplier $\boldsymbol{\lambda}(t) \in \mathbb{R}^{12}$ is connected with the equality constraint (28) and the Lagrange multiplier $\boldsymbol{\mu}(t) \in \mathbb{R}^{4n}$ connected with the inequality constraint (29); $\boldsymbol{r}(t) = \boldsymbol{d}(t) - \sqrt{\boldsymbol{v}(t) \circ \boldsymbol{v}(t) + \boldsymbol{\mu}(t) \circ \boldsymbol{\mu}(t) + \boldsymbol{\varepsilon}_3}$ and $\boldsymbol{v}(t) = \boldsymbol{d}(t) - \boldsymbol{B}(t)\upsilon(t)$; $\circ$ bespeaks the Hadamard product operator; $\boldsymbol{\varepsilon}_3 \to 0_+$ and $\varpi = 6n + 12$.

In other words, as long as we can explore the solution $\boldsymbol{\chi}(t)$ suitable for the TVES problem (30), it means that we have found the optimal solution to the TVQP problem (27)–(29); Next, we will explain the derivation of the ATT-DRNN algorithm and employ it to work out the TVES problem (30), the TVQP problem (27)–(29), and the proposed DAPCMC scheme (27)–(29).

2.3 Algorithm design

First, we set up the following vector-valued EE in the light of the TVES problem (30):

$$\boldsymbol{e}(t) = \mathcal{H}(t)\boldsymbol{\chi}(t) + \boldsymbol{g}(t). \tag{31}$$

Finally, we utilize the RNN evolution rule (Shi and Zhang, 2018) as below:

$$\dot{\boldsymbol{e}}(t) = \frac{\mathrm{d}\boldsymbol{e}(t)}{\mathrm{d}t} = -\zeta \boldsymbol{e}(t), \tag{32}$$

where the fixed convergence factor $\zeta > 0$ has an important impact on the global exponential convergence rate. The larger the ζ chooses, the faster the convergence rate one acquires.

Then, the RNN evolution rule (32) can be further expanded as the following equation on account of the EE :

$$\dot{\mathcal{H}}(t)\boldsymbol{\chi}(t) + \mathcal{H}(t)\dot{\boldsymbol{\chi}}(t) + \dot{\boldsymbol{g}}(t) = -\zeta\left[\mathcal{H}(t)\boldsymbol{\chi}(t) + \boldsymbol{g}(t)\right]. \tag{33}$$

For the handiness of figuring out the optimal solution to the TVQP problem (27)–(28), we reformulate (33) as

$$\boldsymbol{D}(t)\dot{\boldsymbol{\chi}}(t) = -\boldsymbol{V}(t)\boldsymbol{\chi}(t) - \boldsymbol{\varrho}(t) - \zeta\left[\mathcal{H}(t)\boldsymbol{\chi}(t) + \boldsymbol{g}(t)\right], \tag{34}$$

where

$$\boldsymbol{D}(t) = \begin{bmatrix} \boldsymbol{U}(t) & \boldsymbol{A}^{\mathrm{T}}(t) & \boldsymbol{B}^{\mathrm{T}}(t) \\ \boldsymbol{A}(t) & \boldsymbol{0}_{12\times 12} & \boldsymbol{0}_{12\times 4n} \\ \left[\boldsymbol{\ell}_1(t) - \boldsymbol{I}_{4n\times 4n}\right]\boldsymbol{B}(t) & \boldsymbol{0}_{4n\times 12} & \boldsymbol{I}_{4n\times 4n} - \boldsymbol{\ell}_2(t) \end{bmatrix} \in \mathbb{R}^{\varpi\times\varpi}$$

$$\boldsymbol{V}(t) = \begin{bmatrix} \dot{\boldsymbol{U}}(t) & \dot{\boldsymbol{A}}^{\mathrm{T}}(t) & \dot{\boldsymbol{B}}^{\mathrm{T}}(t) \\ \dot{\boldsymbol{A}}(t) & \boldsymbol{0}_{12\times 12} & \boldsymbol{0}_{12\times 4n} \\ \left[\boldsymbol{\ell}_1(t) - \boldsymbol{I}_{4n\times 4n}\right]\dot{\boldsymbol{B}}(t) & \boldsymbol{0}_{4n\times 12} & \boldsymbol{0}_{4n\times 4n} \end{bmatrix} \in \mathbb{R}^{\varpi\times\varpi}$$

$$\boldsymbol{\varrho}(t) = \begin{bmatrix} \dot{\boldsymbol{\varphi}}(t) \\ -\dot{\boldsymbol{c}}(t) \\ \left[\boldsymbol{I}_{4n\times 4n} - \boldsymbol{\ell}_1(t)\right]\dot{\boldsymbol{d}}(t) \end{bmatrix} \in \mathbb{R}^{\varpi},$$

with

$$\begin{cases} \boldsymbol{\ell}_1(t) = \wedge\left[\boldsymbol{\Omega}(t) \circ \boldsymbol{\nu}(t)\right] \\ \boldsymbol{\ell}_2(t) = \wedge\left[\boldsymbol{\Omega}(t) \circ \boldsymbol{\mu}(t)\right] \\ \boldsymbol{\Omega}(t) = \left[\boldsymbol{\nu}(t) \circ \boldsymbol{\nu}(t) + \boldsymbol{\mu}(t) \circ \boldsymbol{\mu}(t) + \boldsymbol{\varepsilon}_3\right]^{-\frac{1}{2}}, \boldsymbol{\varepsilon}_3 \to \boldsymbol{0}_+. \end{cases}$$

We treat (34) as a CRNN model. In order to facilitate its realization in computer system and digital hardware, the CTT-DRNN algorithm and the ATT-DRNN algorithm are introduced in the following subsections.

2.3.1 CTT-DRNN algorithm

In this subsection, a conventional Taylor-type discretization formula is given, and the CTT-DRNN algorithm is obtained by combining it with the CRNN model (34).

Based on Hu et al. (2018), the three-step general Taylor-type discretization formula is formulated as follows:

$$\dot{x}_k = \frac{(-2a+1)x_{k+1} + 6ax_k - (6a+1)x_{k-1} + 2ax_{k-2}}{2\sigma} + O(\sigma^2), \quad k = 2, 3, 4, \ldots, \tag{35}$$

where the argument $a < 0$; k is the updating index; $\sigma > 0$ is the fixed sampling period; $x_k = x(t_k)$ denotes the samping value of function $x(t)$ at time instant $t_k = k\sigma$; $O(\sigma^2)$ is the TE.

By applying the three-step general Taylor-type discretization formula (35) to discretize the CRNN model (34), we can acquire CTT-DRNN algorithm as below:

$$\boldsymbol{\chi}_{k+1} \doteq \frac{6a\boldsymbol{\chi}_k - (6a+1)\boldsymbol{\chi}_{k-1} + 2a\boldsymbol{\chi}_{k-2} - 2\boldsymbol{M}_k\left[\sigma\left(-\boldsymbol{V}_k\boldsymbol{\chi}_k - \boldsymbol{\varrho}_k\right) - h\left(\mathcal{H}_k\boldsymbol{\chi}_k + \boldsymbol{g}_k\right)\right]}{2a-1} \tag{36}$$

where symbol $\doteq$ denotes the computational assignment operation; $\boldsymbol{M}_k, \boldsymbol{V}_k, \mathcal{H}_k, \boldsymbol{\varrho}_k, \boldsymbol{g}_k$, and $\boldsymbol{\chi}_k$ mean the instantaneous values of $\boldsymbol{M}(t), \boldsymbol{V}(t), \mathcal{H}(t), \boldsymbol{\varrho}(t), \boldsymbol{g}(t)$, and $\boldsymbol{\chi}(t)$ sampling at time instant t_k with $\boldsymbol{M}(t)$ denoting the pseudoinverse of $\boldsymbol{D}(t)$; parameter $h = \sigma\zeta$ represents the solution step size generally set at the range of $(0, 1)$.

2.3.2 ATT-DRNN algorithm

According to the analysis of Subsection (2.3.1), on the one hand, the larger the fixed convergence factor ζ, the faster the global convergence rate of the system, thus we should naturally set ζ as large as possible at the beginning to ensure a sublime exponential converging capability of the CRNN model (34). On the other hand, it is recognized that the fixed argument σ as the sampling period is a significant factor affecting the convergence precision of the CTT-DRNN algorithm (36). Generally, the more remarkable convergence precision is guaranteed by a smaller value of σ taken at the initial stage. However, blindly setting a small value of σ may directly lead to an exiguous solution step size h, making it knotty for the solution process to converge rapidly or even proceed normally. Similarly, an excessively huge ζ also makes it hard to ensure a brilliant exactness of the algorithm due to incurring a gigantic solution step size h. It can be seen that the above situations are contradictory to each other. Moreover, according to the changes in system conditions, fixed parameters cannot meet the needs of different states. In view of this, to autonomously adjust the convergence factor ζ and the sampling period σ according to the actual convergence situation, and assure that the global state both has a remarkable convergence rate and outstanding convergence precision, a novel ATT-DRNN algorithm is designed as described in the following text.

First, according to the actual solution status, the adaptive sampling period $\sigma_k = \sigma(t_k)$ is designed as follows:

$$\sigma_k = \frac{q}{\left(p + \|\boldsymbol{e}_k\|_2\right)^{\delta}}, \tag{37}$$

where fixed arguments $p, q > 0$ are applied to adjust the solution accuracy of the algorithm; variable argument δ is utilized to ensure the algorithm precision while adjusting the sampling period change rate; error $\boldsymbol{e}_k = \mathcal{H}_k\boldsymbol{\chi}_k + \boldsymbol{g}_k$ and symbol $\|\cdot\|_2$ represents the 2-norm of a vector.

Accordingly, the adaptive convergence factor $\zeta_k = \zeta(t_k)$ is designed as follows:

$$\zeta_k = \frac{h\left(p + \|\boldsymbol{e}_k\|_2\right)^{\delta}}{q}, \tag{38}$$

where variable argument δ is utilized to ensure the algorithm accuracy while adjusting the global convergence rate of the algorithm; $h = \sigma_k \zeta_k$ is the same as that in (36).

Meanwhile, the corresponding continuous adaptive convergence factor $\zeta(t)$ can be written as follows:

$$\zeta(t) = \frac{h\left(p + \left\|\boldsymbol{e}(t)\right\|_2\right)^{\delta}}{q}. \tag{39}$$

With the help of the continuous adaptive convergence factor $\zeta(t)$ (39), a novel RNN evolution rule can be written as follows:

$$\dot{\boldsymbol{e}}(t) = \frac{\mathrm{d}\boldsymbol{e}(t)}{\mathrm{d}t} = -\zeta(t)\boldsymbol{e}(t). \tag{40}$$

Thus, on the account of the EE (31), the novel RNN evolution rule (40) can be further expanded and reformulated as the ACRNN model:

$$\dot{\boldsymbol{\chi}}(t) = \boldsymbol{M}(t)\left\{-\boldsymbol{V}(t)\boldsymbol{\chi}(t) - \boldsymbol{\varrho}(t) - \zeta(t)\left[\mathcal{H}(t)\boldsymbol{\chi}(t) + \boldsymbol{g}(t)\right]\right\}, \tag{41}$$

where the corresponding parameters are all the same as in the previous section.

Besides, by taking into account the adaptive sampling period σ_k (37), the adaptive three-step general Taylor-type discretization formula can be expressed as follows:

$$\dot{x}_k = \frac{(-2a+1)x_{k+1} + 6ax_k - (6a+1)x_{k-1} + 2ax_{k-2}}{2\sigma_k} + O(\sigma_k^2), \quad k = 2, 3, 4, \ldots, \tag{42}$$

Then, we can acquire the ATT-DRNN algorithm by using the adaptive three-step general Taylor-type discretization formula (42) to discretize the ACRNN model (41), which can be written as follows:

$$\begin{aligned}\boldsymbol{\chi}_{k+1} \doteq\ & \tfrac{6a}{2a-1}\boldsymbol{\chi}_k - \tfrac{6a+1}{2a-1}\boldsymbol{\chi}_{k-1} + \tfrac{2a}{2a-1}\boldsymbol{\chi}_{k-2} \\ & -\tfrac{2}{2a-1}\boldsymbol{M}_k\left[\sigma_k\left(-\boldsymbol{V}_k\boldsymbol{\chi}_k - \boldsymbol{\varrho}_k\right) - h\left(\mathcal{H}_k\boldsymbol{\chi}_k + \boldsymbol{g}_k\right)\right],\end{aligned} \tag{43}$$

where the solution step size $h = \sigma_k\zeta_k$ is generally set at the range of $(0, 1)$. Moreover, three initial state vectors $\boldsymbol{\chi}_k$ with $k = 0, 1, 2$ are necessary to start up the proposed ATT-DRNN algorithm (43). The first one $\boldsymbol{\chi}_0$ consists of υ_0, $\boldsymbol{\lambda}_0$, and $\boldsymbol{\mu}_0$, where υ_0 is determined by the initial joint-velocity vectors of the LA and RA, while $\boldsymbol{\lambda}_0$ and $\boldsymbol{\mu}_0$ are relatively arbitrarily set. The remaining initial state vectors can be generated by utilizing an adaptive Euler-type DRNN algorithm, which can be obtained by applying adaptive Euler forward formula to discretize the ACRNN model (41), i.e., $\boldsymbol{\chi}_{k+1} \doteq \boldsymbol{\chi}_k + \sigma_k\dot{\boldsymbol{\chi}}_k$ with $\dot{\boldsymbol{\chi}}_k = \boldsymbol{M}_k\left[-\boldsymbol{V}_k\boldsymbol{\chi}_k - \boldsymbol{\varrho}_k - \zeta_k\left(\mathcal{H}_k\boldsymbol{\chi}_k + \boldsymbol{g}_k\right)\right]$.

Remark 2.5: By observing (37), it is evident that the adaptive sampling period σ_k continuously adjusts according to the changes in the RE $\|\boldsymbol{e}_k\|_2$, with an increase in the RE $\|\boldsymbol{e}_k\|_2$ and a decrease in the sampling period σ_k, and vice versa.

Remark 2.6: By observing (38), it is evident that the adaptive convergence factor ζ_k continuously adjusts according to the changes in the RE $\|\boldsymbol{e}_k\|_2$, when the RE $\|\boldsymbol{e}_k\|_2$ increases, the adaptive convergence factor ζ_k grows, leading to a higher convergence rate, and vice versa.

Remark 2.7: The solution step size h procured through multiplying σ_k and ζ_k is always a positive constant. By observing Equations (37) and (38) simultaneously, it can be easily found that σ_k and ζ_k exhibit the reciprocal states to each other. That is to say, when the RE $\|\boldsymbol{e}_k\|_2$ is large, the algorithm will adjust and yield a smaller sampling period σ_k and a larger convergence factor ζ_k to guarantee a rapid convergence of the algorithm in an extremely short sampling time; on the contrary, when the RE $\|\boldsymbol{e}_k\|_2$ reduces, the algorithm will adaptively increase the sampling period σ_k and simultaneously decrease the convergence factor ζ_k. By decreasing the sampling period and increasing the convergence rate, the algorithm can promptly complete the calculation and improve its computational efficiency. Therefore, the ATT-DRNN algorithm (43) can consider both computational accuracy and convergence efficiency during the calculation process.

3 Theoretical analyses and results

This section theoretically analyzes the convergence property of the ACRNN model (41) and the computational precision of the ATT-DRNN algorithm (43) for solving the TVQP problem (27)–(29).

Theorem 1: With the parameters $h, p, q, \delta > 0$ of the continuous adaptive convergence factor $\zeta(t)$, the RE $\left\|\boldsymbol{e}(t)\right\|_2$ generated by the ACRNN model (41) exponentially converges to zero in a large-scale manner with the exponential convergence rate at least being hp^{δ}/q.

Proof: To begin with, by exploiting the EE (31), a Lyapunov function can be chosen as follows:

$$\mathbb{L}(t) = \frac{\left\|\boldsymbol{e}(t)\right\|_2^2}{2}. \tag{44}$$

Then, the time derivative of the function $\mathbb{L}(t)$ is obtained by referring to (40):

$$\dot{\mathbb{L}}(t) = \boldsymbol{e}^{\mathrm{T}}(t)\dot{\boldsymbol{e}}(t) = -\zeta(t)\left\|\boldsymbol{e}(t)\right\|_2^2. \tag{45}$$

Observing (44) and (45), one can draw the following conclusions.

(1) If and only if $\boldsymbol{e}(t) = \boldsymbol{0}$, $\mathbb{L}(t) = 0$; otherwise, $\mathbb{L}(t) > 0$.

(2) If and only if $\boldsymbol{e}(t) = \boldsymbol{0}$, $\dot{\mathbb{L}}(t) = 0$; otherwise, $\dot{\mathbb{L}}(t) < 0$.

In other words, the function $\mathbb{L}(t)$ is positive definite and its derivative $\dot{\mathbb{L}}(t)$ is negative definite, which satisfies the Lyapunov stability theory conditions (Isidori, 1989). Thus, it can be concluded that the RE $\left\|\boldsymbol{e}(t)\right\|_2$ converges to zero in a large-scale manner.

Second, by reconstructing and expanding (45), we acquire

$$\dot{\mathbb{L}}(t) = -2\frac{h\left(p + \left\|\boldsymbol{e}(t)\right\|_2\right)^{\delta}}{q}\frac{\left\|\boldsymbol{e}(t)\right\|_2^2}{2} = -2\frac{h\left(p + \left\|\boldsymbol{e}(t)\right\|_2\right)^{\delta}}{q}\mathbb{L}(t). \tag{46}$$

Furthermore, based on (46), the following inequality can be further formulated as follows:

$$\dot{\mathbb{L}}(t) = -2\frac{h\left(p + \left\|\boldsymbol{e}(t)\right\|_2\right)^{\delta}}{q}\mathbb{L}(t) \leq -2\frac{hp^{\delta}}{q}\mathbb{L}(t). \tag{47}$$

Attempting to figure out the inequality (47), we get

$$\frac{\left\|\boldsymbol{e}(t)\right\|_2^2}{2} \leq \frac{\left\|\boldsymbol{e}(0)\right\|_2^2}{2}\exp\left(-2\frac{hp^{\delta}}{q}t\right), \tag{48}$$

and the inequality (48) can be further formulated as

$$\left\|\boldsymbol{e}(t)\right\|_2 \leq \left\|\boldsymbol{e}(0)\right\|_2 \exp\left(-\frac{hp^{\delta}}{q}t\right). \tag{49}$$

Until now, in accordance with the inequality (49), we can conclude that the RE $\left\|\boldsymbol{e}(t)\right\|_2$ of the ACRNN model (41) exponentially converges to zero in a large-scale manner with the exponential convergence rate at least being hp^{δ}/q, which completes the proof. ■

Theorem 2: With the parameters $h, p, q, \delta > 0$ of the adaptive sampling period σ_k, the ATT-DRNN algorithm (43) is zero-stable and convergent with the TE of order $\boldsymbol{O}((q/p^{\delta})^3)$. In addition, the theoretical solution of TVES problem (30) converged by the ATT-DRNN algorithm (43) with a maximal steady-status RE $\lim_{k\to\infty}\sup\left\|\boldsymbol{e}_{k+1}\right\|_2$ bing of order $O((q/p^{\delta})^3)$.

Proof: First, drawing on the experience of Hu et al. (2018), it testified that the adaptive three-step general Taylor-type discretization formula (42) meets the conditions of zero-stable and convergent, which has a TE term $\boldsymbol{O}(\sigma^2)$.

By using the adaptive three-step general Taylor-type discretization formula (42) to discretize the ACRNN model (41), the new ATT-DRNN algorithm (43) can be rewritten as follows:

$$\begin{aligned}\boldsymbol{\chi}_{k+1} &= \tfrac{6a}{2a-1}\boldsymbol{\chi}_k - \tfrac{6a+1}{2a-1}\boldsymbol{\chi}_{k-1} + \tfrac{2a}{2a-1}\boldsymbol{\chi}_{k-2}\\ &-\tfrac{2}{2a-1}\boldsymbol{M}_k\left[\sigma_k\left(-\boldsymbol{V}_k\boldsymbol{\chi}_k - \boldsymbol{\varrho}_k\right) - h\left(\mathcal{H}_k\boldsymbol{\chi}_k + \boldsymbol{g}_k\right)\right] + \boldsymbol{O}(\sigma_k^3),\end{aligned} \tag{50}$$

where $\boldsymbol{O}(\sigma_k^3)$ is the TE term.

Due to the development process recommended above, it is distinct that the ATT-DRNN algorithm (43) originating from (50) is also zero-stable and similarly convergent with the TE of order $\boldsymbol{O}(\sigma_k^3)$. Therefore, we get

$$\lim_{k\to\infty}\sigma_k = \lim_{k\to\infty}\frac{q}{\left(p+\|\boldsymbol{e}_k\|_2\right)^{\delta}} = \frac{q}{p^{\delta}}, \tag{51}$$

which means that the TE term for the ATT-DRNN algorithm (43) is $\boldsymbol{O}((q/p^{\delta})^3)$.

Then, based on the ATT-DRNN algorithm (43), the theoretical solution $\boldsymbol{\chi}_{k+1}^*$ of TVES problem (30) can be expressed as follows:

$$\boldsymbol{\chi}_{k+1}^* = \boldsymbol{\chi}_{k+1} + \boldsymbol{O}((q/p^{\delta})^3). \tag{52}$$

In addition, it is known that the theoretical solution $\boldsymbol{\chi}_{k+1}^*$ of the TVES problem (30) satisfies $\mathcal{H}_{k+1}\boldsymbol{\chi}_{k+1}^* + \boldsymbol{g}_{k+1} = \boldsymbol{0}$. Thus, by combining (51) with (52), we can draw the following conclusion:

$$\begin{aligned}&\lim_{k\to\infty}\sup\left\|\boldsymbol{e}_{k+1}\right\|_2\\ &= \lim_{k\to\infty}\sup\left\|\mathcal{H}_{k+1}\boldsymbol{\chi}_{k+1} + \boldsymbol{g}_{k+1}\right\|_2\\ &= \lim_{k\to\infty}\sup\left\|\mathcal{H}_{k+1}\boldsymbol{\chi}_{k+1}^* + \boldsymbol{g}_{k+1} - \mathcal{H}_{k+1}\boldsymbol{O}(\sigma_k^3)\right\|_2\\ &= \lim_{k\to\infty}\sup\left\|\mathcal{H}_{k+1}\boldsymbol{O}(\sigma_k^3)\right\|_2\\ &\leq \lim_{k\to\infty}\sup(\left\|\mathcal{H}_{k+1}\right\|_{\mathrm{F}}\left\|\boldsymbol{O}(\sigma_k^3)\right\|_2) = O((q/p^{\delta})^3).\end{aligned} \tag{53}$$

Based on (53), it can be concluded that the maximal steady-status RE $\lim_{k\to\infty}\sup\left\|\boldsymbol{e}_{k+1}\right\|_2$ generated by the ATT-DRNN algorithm (43) is $O((q/p^{\delta})^3)$. Thus, we complete the proof. ■

4 Illustrative examples

In this section, a numerical simulation example is provided first and explored to state explicitly the remarkable competence of the devised ATT-DRNN algorithm (43) when tackling the TVQP problem (27)–(29). Then, two examples of dual-arm robot control are provided to demonstrate the effectiveness of the devised ATT-DRNN algorithm (43) in addressing the proposed DAPCMC scheme (27)–(29). Meanwhile, we utilize the CTT-DRNN algorithm (36) and the CET-DRNN algorithm in Wu and Zhang (2023) for comparisons to show the superior performance of the devised ATT-DRNN algorithm (43). To help readers understand the algorithm implementation process, the pseudo-code of the proposed ATT-DRNN algorithm (43) for addressing the DAPCMC scheme (27)–(29) is presented in Algorithm 1.

Set: α, β, ξ, a, h, p, q, δ, T_{e}, γ, $\dot{\mathfrak{s}}_{\mathcal{L}}^{\pm}$, $\dot{\mathfrak{s}}_{\mathcal{R}}^{\pm}$, $\dot{\mathfrak{s}}_{\mathcal{L}}^{\pm}$, $\dot{\mathfrak{s}}_{\mathcal{R}}^{\pm}$;
Initialize: $\mathfrak{s}_{\mathcal{L}}(0)$, $\mathfrak{s}_{\mathcal{R}}(0)$, $\mho_{\mathcal{L}}(0)$, $\mho_{\mathcal{R}}(0)$, $\lambda(0)$, $\boldsymbol{\mu}(0)$, $t(0)$;
while $t_k \leq T_{\mathrm{e}}$ **do**
  Generate $\mho_{\mathcal{L}}^{\pm}(t_k)$, $\mho_{\mathcal{R}}^{\pm}(t_k)$ from the JCLS;
  Compute $\mathcal{H}(t_k)$, $\boldsymbol{\chi}(t_k)$, $\boldsymbol{g}(t_k)$, $\boldsymbol{M}(t_k)$, $\boldsymbol{V}(t_k)$, $\boldsymbol{\varrho}(t_k)$, $\boldsymbol{V}(t_k)$, $\sigma(t_k)$, $\zeta(t_k)$;
  if $k = 0, 1$ **then**
    Compute $\boldsymbol{\chi}(t_{k+1})$ via the adaptive Euler-type DRNN algorithm:

$$\boldsymbol{\chi}(t_{k+1}) \doteq \boldsymbol{\chi}(t_k) + \sigma(t_k)\dot{\boldsymbol{\chi}}(t_k);$$

  else
  Compute $\boldsymbol{\chi}(t_{k+1})$ via the ATT-DRNN algorithm:

$$\begin{aligned}\boldsymbol{\chi}(t_{k+1}) \doteq{}& \frac{6a}{2a-1}\boldsymbol{\chi}(t_k) - \frac{6a+1}{2a-1}\boldsymbol{\chi}(t_{k-1}) + \frac{2a}{2a-1}\boldsymbol{\chi}(t_{k-2})\\ &-\frac{2}{2a-1}\sigma(t_k)\dot{\boldsymbol{\chi}}(t_k);\end{aligned}$$

  Obtain $\mho_{\mathcal{L}}(t_{k+1})$ from the first n elements of $\boldsymbol{\chi}(t_{k+1})$ and $\mho_{\mathcal{R}}(t_{k+1})$ from the $n+1$ to $2n$ elements of $\boldsymbol{\chi}(t_{k+1})$;
  if $k = 0$ **then**
    Compute $\mathfrak{s}_{\mathcal{L}}(t_{k+1}) = \mathfrak{s}_{\mathcal{L}}(t_k) + \sigma(t_k)\mho_{\mathcal{L}}(t_{k+1})$, $\mathfrak{s}_{\mathcal{R}}(t_{k+1}) = \mathfrak{s}_{\mathcal{R}}(t_k) + \sigma(t_k)\mho_{\mathcal{R}}(t_{k+1})$;
  else
    Compute $\mathfrak{s}_{\mathcal{L}}(t_{k+1}) = \frac{4}{3}\mathfrak{s}_{\mathcal{L}}(t_k) - \frac{1}{3}\mathfrak{s}_{\mathcal{L}}(t_{k-1}) + \frac{2}{3}\sigma(t_k)\mho_{\mathcal{L}}(t_{k+1})$, $\mathfrak{s}_{\mathcal{R}}(t_{k+1}) = \frac{4}{3}\mathfrak{s}_{\mathcal{R}}(t_k) - \frac{1}{3}\mathfrak{s}_{\mathcal{R}}(t_{k-1}) + \frac{2}{3}\sigma(t_k)\mho_{\mathcal{R}}(t_{k+1})$;
    Update $t_{k+1} = t_k + \sigma(t_k)$;
  end

Algorithm 1. Pseudo-code of the proposed ATT-DRNN algorithm (43) for addressing the DAPCMC scheme (27)–(29).

4.1 Numerical simulation verification

A specific TVQP problem with equality and inequality constraints is provided, the details of which are

outlined below:

$$\begin{aligned}\min_{\mho(t)}&[\cos(3t)/6+1]\mho_1^2(t)+\sin(t)\mho_1(t)\mho_2(t)\\&+\cos(3t)\mho_1(t)\mho_3(t)+2\sin(t)\mho_1(t)\\&+[\sin(3t)/6+1]\mho_2^2(t)+[\sin(2t)+1]\mho_2(t)\mho_4(t)\\&+2\cos(t)\mho_2(t)-[\cos(t)/2+1/2]\mho_3^2(t)\\&+[\cos(2t)\mho_3(t)\mho_4(t)]/2+\sin(2t)\mho_3(t)\\&-[\sin(t)/2+1/2]\mho_4^2(t)+\cos(2t)\mho_4(t)\\\text{s.t.}\ &-t\sin(t/2)\mho_1(t)+\{[4\cos(t/2)]/5+1/5\}\mho_2(t)=t\sin(2t+1)\\&-[3\cos(9t/10)]/2\mho_3(t)+\sin(9t/10)\mho_4(t)=-\cos(3t/2)/2-3/10\\&-0.2\sin(3t)-1.2\le\mho_1(t),\mho_2(t),\mho_3(t),\mho_4(t)\le 0.2\sin(3t)+1.2\end{aligned}\tag{54}$$

where $\mho(t)=[\mho_1(t),\mho_2(t),\mho_3(t),\mho_4(t)]^{\mathrm{T}}$. By referring to the standard form of the TVQP problem (27)–(29), the corresponding coefficients are as follows:

$$\boldsymbol{U}(t)=\begin{bmatrix}\cos(3t)/3+2 & \sin(t) & \cos(3t) & 0\\ \sin(t) & \sin(3t)/3+2 & 0 & \sin(2t)+1\\ \cos(3t) & 0 & -\cos(t)-1 & \cos(2t)/2\\ 0 & \sin(2t)+1 & \cos(2t)/2 & -\sin(t)-1\end{bmatrix}\in\mathbb{R}^{4\times4}$$

$$\boldsymbol{\varphi}(t)=\begin{bmatrix}\sin(t) & \cos(t) & \sin(2t) & \cos(2t)\end{bmatrix}\in\mathbb{R}^{4}$$

$$\boldsymbol{A}(t)=\begin{bmatrix}-t\sin(t/2) & [4\cos(t/2)]/5+1/5 & 0 & 0\\ 0 & 0 & -[3\cos(9t/10)]/2 & \sin(9t/10)\end{bmatrix}\in\mathbb{R}^{2\times4}$$

$$\boldsymbol{B}(t)=\begin{bmatrix}\boldsymbol{I}_{4\times4}\\ -\boldsymbol{I}_{4\times4}\end{bmatrix}\in\mathbb{R}^{8\times4}$$

$$\boldsymbol{c}(t)=\begin{bmatrix}t\sin(2t+1)\\ -\cos(3t/2)/2-3/10\end{bmatrix}\in\mathbb{R}^{2}$$

$$\boldsymbol{d}(t)=\begin{bmatrix}0.2\sin(3t)+1.2\end{bmatrix}_{8\times1}\in\mathbb{R}^{8}.$$

To successfully address the above TVQP problem (54) using the ATT-DRNN algorithm (42), we set parameters $h=0.1$, $p=5$, $q=0.05$, and $\gamma=0.001$; the initial values of $\mho(0),\boldsymbol{\lambda}(0)$, and $\boldsymbol{\mu}(0)$ are set to random values at the range of $(0,0.001)$. Then, the entire simulation calculation time in the program is uniformly set to $T_{\mathrm{e}}=4$ s. Besides, we set the fixed sampling period $\sigma=0.01$ s for the CTT-DRNN algorithm (36) and the CET-DRNN algorithm in Wu and Zhang (2023).

Figure 2A shows the element trajectories of the status vector $\mho(t)$ generated by the ATT-DRNN algorithm (43) with $a=-0.3$ and $\delta=2$, which are strictly confined to the ranges of inequality constraints. Meanwhile, it can be seen from Figure 2B that the equality constraint $\boldsymbol{A}(t)\mho(t)$ of the TVQP problem (54) can be promptly satisfied and can consistently maintain this state. To save space, the solving states of the proposed algorithm (43) with different a and δ, as well as similar figures of other algorithms, are omitted.

In order to research the impact of δ on the solving results of the ATT-DRNN algorithm (43), the variation trajectory of the RE $\|\boldsymbol{e}(t)\|_2$ when taking different δ with $a=-0.3$ is exhibited in Figure 2C. As we can see that, when entering the steady state, the RE $\|\boldsymbol{e}(t)\|_2$ maintains at around 10^{-4} with $\delta=1$, 10^{-6} with $\delta=2$, and 10^{-8} with $\delta=3$. In other words, as the setting value of δ increases, the convergence speed of the ATT-DRNN algorithm (43) is accelerated, and the solution precision is higher.

To demonstrate the excellent performance of the proposed ATT-DRNN algorithm (43) compared with other conventional algorithms, we further investigate the REs $\|\boldsymbol{e}(t)\|_2$ generated by the algorithms of ATT-DRNN (43) with $\delta=2$, CTT-DRNN (36) and CET-DRNN in Wu and Zhang (2023) by figuring out the TVQP problem (54), respectively. The REs $\|\boldsymbol{e}(t)\|_2$ synthesized by these three algorithms with different a are displayed in Figure 2D. It can be seen that the RE $\|\boldsymbol{e}(t)\|_2$ generated by the ATT-DRNN algorithm (43) with $\delta=2$ and different a values can converge as small as 10^{-6} in approximately 0.3 s. The REs $\|\boldsymbol{e}(t)\|_2$ generated by the CTT-DRNN algorithm (36) with different a values can converge to roughly 10^{-4} in 1 s. The RE $\|\boldsymbol{e}(t)\|_2$ generated by the CET-DRNN algorithm in Wu and Zhang (2023) merely converges to around 10^{-2} in 0.5 s. Overall, the solution accuracy and convergence rate of the ATT-DRNN algorithm (43) are superior to the other two conventional algorithms. Besides, it can be concluded that the computing precision of the ATT-DRNN algorithm (43) is higher with the smaller absolute value of a chosen. In addition, the variation curves of the adaptive sampling period $\sigma(t)$ and the adaptive convergence factor $\zeta(t)$ with different δ values are portrayed in Figures 2E, F, separately, which indicate that $\sigma(t)$ and $\zeta(t)$ can converge and stabilize to their corresponding values in an extremely short time. The greater the δ chosen, the smaller the final stable value of $\sigma(t)$ and the larger the final stable value of $\zeta(t)$. Furthermore, during the solving process, as the RE $\|\boldsymbol{e}(t)\|_2$ rapidly converges and decreases at the beginning stage, the change of $\sigma(t)$ is inversely proportional to it, and the change of $\zeta(t)$ is directly proportional to it, which is consistent with our previous analysis conclusions from *Remark 2.5* to *Remark 2.7*.

In summary, the several situations above confirm that the ATT-DRNN algorithm (43) has an excellent ability to solve the TVQP problem. Compared with other conventional algorithms, the proposed algorithm has faster convergence speed and higher precision.

4.2 Control case I of dual-arm robot

For this fraction, we establish a DAPCMC scheme consisting of two UR5 robotic arms placed on the contralateral side, controlled by the ATT-DRNN algorithm (43) for dual heart-shaped trajectory tracking. In addition, the UR5 robotic arm is a sensitive lightweight 6-DOF robot, which has a small footprint and can be directly installed in a narrow workspace to complete tasks with high sensitivity requirements (Vivas and Sabater, 2021).

According to the design form of the DAPCMC scheme (27), (28), we establish a particular TVQP problem for a UR5 dual-arm robot consisting of two 6-DOF (with $n=6$) UR5 robotic arms. In this scheme, the LA and RA's initial joint-angle vectors are set

10.3389/fnbot.2024.1406604

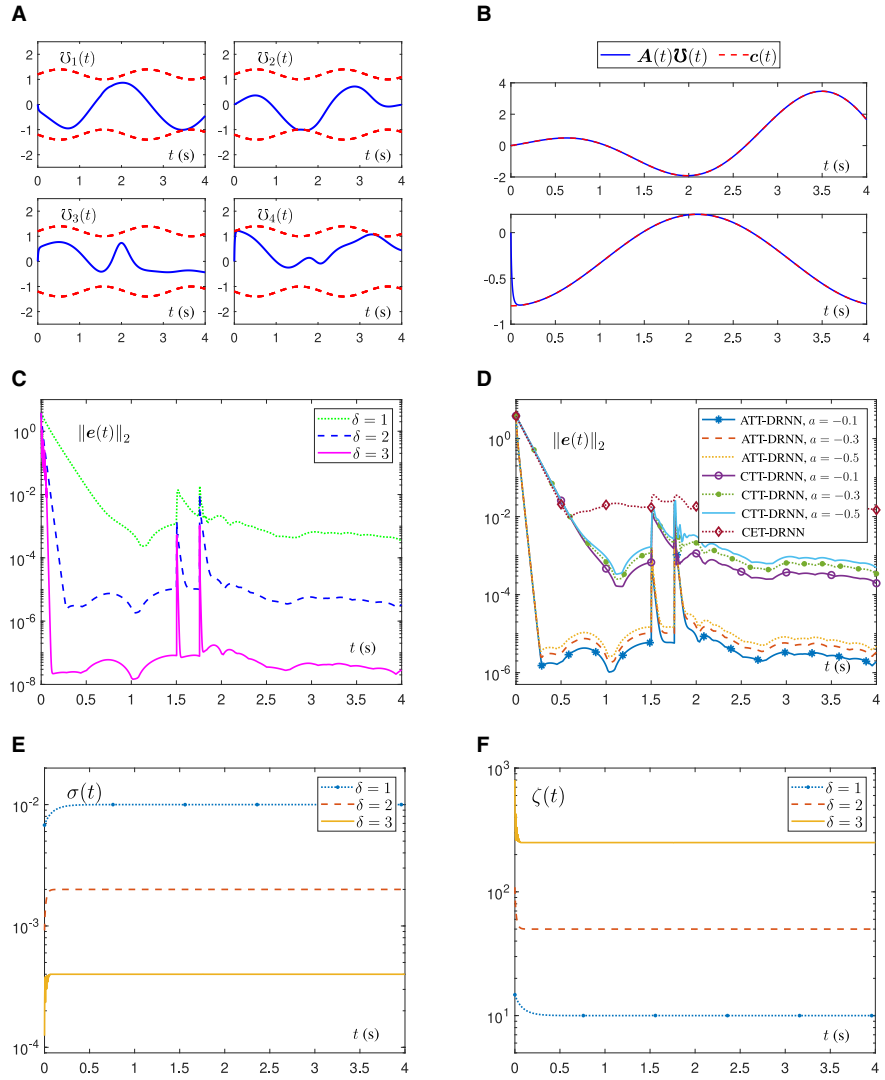


FIGURE 2
Numerical simulation results of the ATT-DRNN (43), CTT-DRNN (36), and CET-DRNN in Wu and Zhang (2023) algorithms for addressing the TVQP problem (53), separately. **(A)** Status vector $\mathfrak{v}(t)$ under inequality constraint generated by the ATT-DRNN algorithm (43) with $a = -0.3$ and $\delta = 2$. **(B)** $A(t)\mathfrak{v}(t)$ under equality constraint $c(t)$ obtained by the ATT-DRNN algorithm (43) with $a = -0.3$ and $\delta = 2$. **(C)** RE $\|e(t)\|_2$ generated by the ATT-DRNN algorithm (43) with $a = -0.3$ and different δ. **(D)** RE $\|e(t)\|_2$ by three algorithms with different a and $\delta = 2$. **(E)** Adaptive sampling period $\sigma(t)$ with different δ. **(F)** Adaptive convergence factor $\zeta(t)$ with different δ.

TABLE 2 The D-H parameters of UR5 robotic arm and its joint-angle and joint-velocity physical limits in the DAPCMC scheme (27)–(29).

Joint	$\widetilde{\alpha}$ (rad)	$\widetilde{a}$ (m)	$\widetilde{d}$ (m)	$\mathfrak{z}$ (rad)	$\mathfrak{z}^+$ (rad)	$\mathfrak{z}^-$ (rad)	$\dot{\mathfrak{z}}^+$ (rad/s)	$\dot{\mathfrak{z}}^-$ (rad/s)
1	1.5708	0	0.0892	$\mathfrak{z}_1$	1.5708	−1.5708	0.285	−0.285
2	0	−0.4250	0	$\mathfrak{z}_2$	0	−3.1416	0.285	−0.285
3	0	−0.3923	0	$\mathfrak{z}_3$	0	−3.1416	0.285	−0.285
4	1.5708	0	0.1092	$\mathfrak{z}_4$	1.5708	−1.5708	0.285	−0.285
5	−1.5708	0	0.0948	$\mathfrak{z}_5$	3.1416	0	0.285	−0.285
6	0	0	0.0825	$\mathfrak{z}_6$	1.5708	−1.5708	0.285	−0.285

as $\mathfrak{z}_{\mathcal{L}}(0) = [0, \pi/12, -4\pi/9, 13\pi/36, \pi/2, 0]^{\mathrm{T}}$ rad and $\mathfrak{z}_{\mathcal{R}}(0) = [0, -\pi/12, 4\pi/9, -13\pi/36, -\pi/2, 0]^{\mathrm{T}}$ rad, respectively; the LA and RA's initial joint-velocity vectors are set as $\dot{\mathfrak{z}}_{\mathcal{L}}(0) = \dot{\mathfrak{z}}_{\mathcal{R}}(0) = [\mathbf{0}]_{6\times 1}$ rad/s. We set the design parameters as $h = 0.2$, $p = 10$, $q = 0.2$, and $\delta = 2$. Then, the other correlative parameters are taken as $a = -0.1$, $\xi = 5$, and $\gamma = 0.001$; λ and μ are set to random values at the range of $(0, 0.001)$; α and β for two robotic arms are uniformly set as 0.8 and 0.1. Besides, we set the fixed

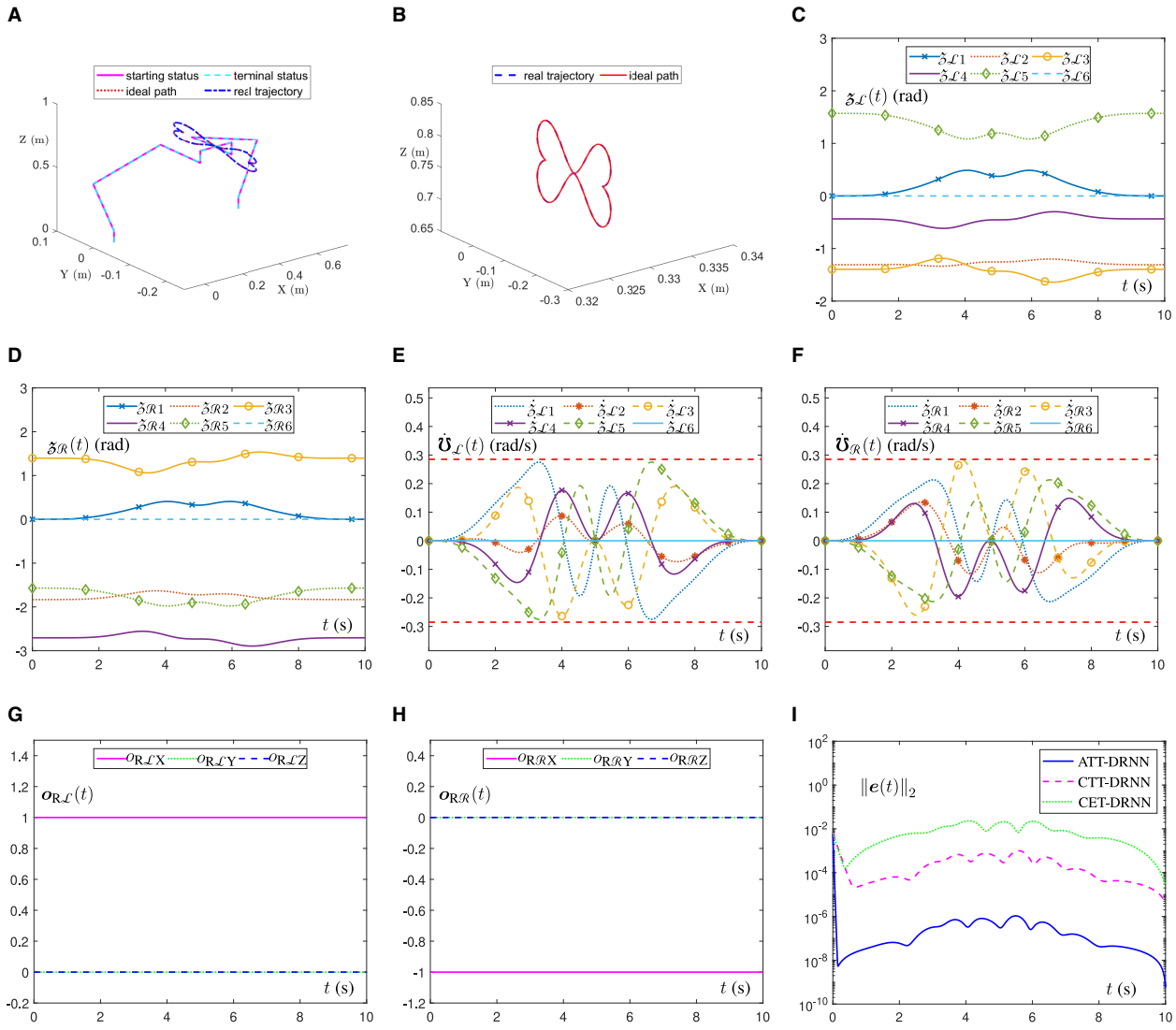


FIGURE 3
The ATT-DRNN algorithm (43) controls the posture collaboration motion of the UR5 dual-arm robot with two arms placed on the contralateral side for the dual heart-shaped trajectory tracking. **(A)** Starting and terminal statuses of the dual-arm robot and the end-executor real and ideal trajectories in 3D space. **(B)** Outlines of real trajectory and ideal path in a 3D space. **(C)** Variations of the LA joint angles $\mathfrak{z}_{\mathcal{L}}(t)$. **(D)** Variations of the RA joint angles $\mathfrak{z}_{\mathcal{R}}(t)$. **(E)** Variations of the LA joint velocities $\dot{\mathfrak{z}}_{\mathcal{L}}(t)$. **(F)** Variations of the RA joint velocities $\dot{\mathfrak{z}}_{\mathcal{R}}(t)$. **(G)** Variations of the LA end-executor orientation $o_{\mathrm{R}\mathcal{L}}(t)$. **(H)** Variations of the RA end-executor orientation $o_{\mathrm{R}\mathcal{R}}(t)$. **(I)** The RE $\|e(t)\|_2$ generated by three different algorithms.

sampling period $\sigma = 0.01$ s for the CTT-DRNN algorithm (36) and the CET-DRNN algorithm in Wu and Zhang (2023). Moreover, the D-H parameters of the UR5 robotic arm and its joint-angle and joint-velocity physical limits in the DAPCMC scheme (27)–(29) are exhibited in Table 2.

Figure 3A illustrates the movement trajectory outlines of the dual-arm robot in a 3D space. It can be observed that the end-executor real trajectory is unanimous with the ideal path and that the joint-angle terminal status also perfectly overlaps with the starting status for each side of the dual robotic arms, which can be further confirmed by Figures 3C, D. Similarly, in Figure 3B, the ATT-DRNN algorithm (43) controls the dual-arm robot to achieve the posture collaboration motion and accomplish the dual heart-shaped path tracking. Finally, Figures 3E, F outline the joint-velocity variation curves for the joints of the left and right robotic arms. Clearly, all the joint-velocity values are not beyond the joint-velocity physical limits set at the beginning. Besides, the end-executor orientation variation curves are shown in Figures 3G, H, which remain constant during the task execution. Furthermore, in Figure 3I, the RE $\|e(t)\|_2$ generated by the ATT-DRNN algorithm (43) for addressing the DAPCMC scheme (27)-(29) maintains at around 10^{-7}. By contrast, the RE $\|e(t)\|_2$ generated by the CTT-DRNN algorithm (36) keeps at roughly 10^{-4}, and that generated by the CET-DRNN algorithm in Wu and Zhang (2023) can merely remain at about 10^{-2}.

In addition, Figure 4 shows the position error variation curves of the end-executor when the UR5 dual-arm robot tracks the dual heart-shaped trajectory under the control of different algorithms. In Figure 4A, the dual-arm robot controlled by the ATT-DRNN algorithm (43) can accomplish the trajectory following task

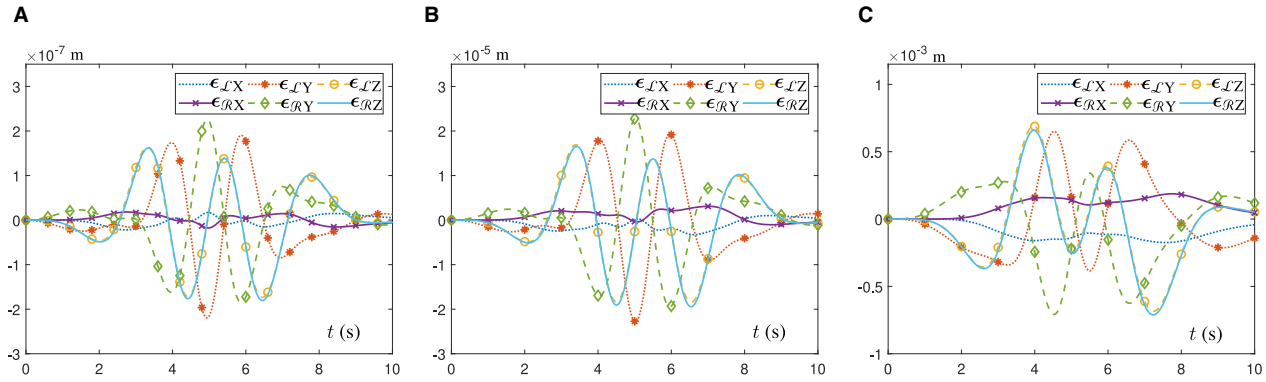


FIGURE 4
Variations of the end-executor position error when the UR5 dual-arm robot with two arms placed on the contralateral side to achieve the posture collaboration motion for the dual heart-shaped trajectory tracking. **(A)** Generated by the ATT-DRNN algorithm (43). **(B)** Generated by the CTT-DRNN algorithm (36). **(C)** Generated by the CET-DRNN algorithm in Wu and Zhang (2023).

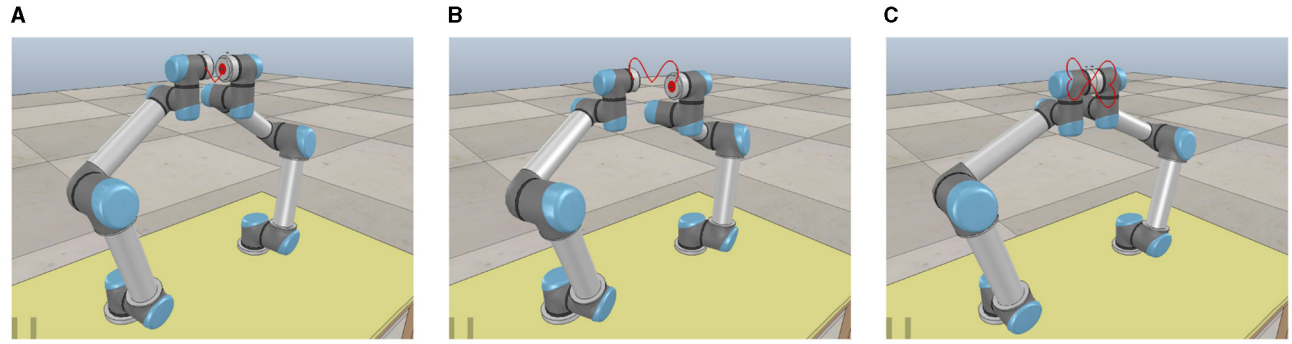


FIGURE 5
Snapshots of the UR5 dual-arm robot with two arms placed on the contralateral side during the trajectory tracking task of the dual heart-shaped path via the ATT-DRNN algorithm (43) solving the DAPCMC scheme (27)–(29). **(A)** Capturing at the starting moment. **(B)** Capturing at the intermediate moment. **(C)** Capturing at the terminal moment.

accurately with the maximal position error of the end-executor being less than 2.3×10^{-7} m. In Figure 4B, the dual-arm robot controlled by the CTT-DRNN algorithm (36) can realize the maximal tracking error of the end-executor no more than 2.3×10^{-5} m. In Figure 4C, the dual-arm robot controlled by the CET-DRNN algorithm in Wu and Zhang (2023) can merely ensure that the position error of the end-executor is within 7×10^{-4} m.

To further simulate the movement status of the dual-arm robot vividly and intuitively in the physical scene, we utilize a virtual robot experiment platform (i.e., CoppeliaSim 2020) to show the real-time status of the dual-arm robot following the ideal paths with the help of the ATT-DRNN algorithm (43) in solving the DAPCMC scheme (27)–(29). Snapshots describing the movement process (i.e., starting moment, intermediate moment, and terminal moment) are shown in Figure 5.

4.3 Control case II of dual–arm robot

For this fraction, we establish a DAPCMC scheme consisting of two UR5 robotic arms placed on the identical side, controlled by the ATT-DRNN algorithm (43) for the heart-shaped and auspicious cloud trajectory tracking.

According to the design form of the DAPCMC scheme (27), (28), we establish another particular TVQP problem for a UR5 dual-arm robot consisting of two 6-DOF UR5 robotic arms. In this scheme, the LA and RA's initial joint-angle vectors are set as $\mathfrak{z}_{\mathcal{L}}(0) = \mathfrak{z}_{\mathcal{R}}(0) = [0, -\pi/12, -2\pi/3, \pi/4, -\pi/2, 0]^{\mathrm{T}}$ rad; the LA and RA's initial joint-velocity vectors are set as $\dot{\mathfrak{z}}_{\mathcal{L}}(0) = \dot{\mathfrak{z}}_{\mathcal{R}}(0) = [\mathbf{0}]_{6\times 1}$ rad/s. We set the design parameters as $h = 0.2$, $p = 5$, $q = 0.05$, and $\delta = 2$. Then, the other correlative parameters are taken as $a = -0.5$, $\xi = 5$, and $\gamma = 0.001$; λ and μ are set to random values at the range of $(0, 0.001)$; α and β for two robotic arms are uniformly set as 0.8 and 0.1. Additionally, the joint-angle and joint-velocity physical limits in the DAPCMC scheme (27)-(28) are separately set as follows: $\mathfrak{z}_{\mathcal{L}}^{+} = 0.3$ rad/s, $\mathfrak{z}_{\mathcal{L}}^{-} = -0.3$ rad/s, $\mathfrak{z}_{\mathcal{R}}^{-} = 0.54$ rad/s, and $\mathfrak{z}_{\mathcal{R}}^{-} = -0.54$ rad/s.

Figure 6A shows the movement trajectory outlines of the dual-arm robot in a 3D space. It is evident that the end-executor real trajectory aligns seamlessly with the ideal path. Moreover, the terminal statuses of the joint angles for both robotic arms precisely coincide with their initial ones, as corroborated by the results presented in Figures 6C, D. Similarly, in Figure 6B,

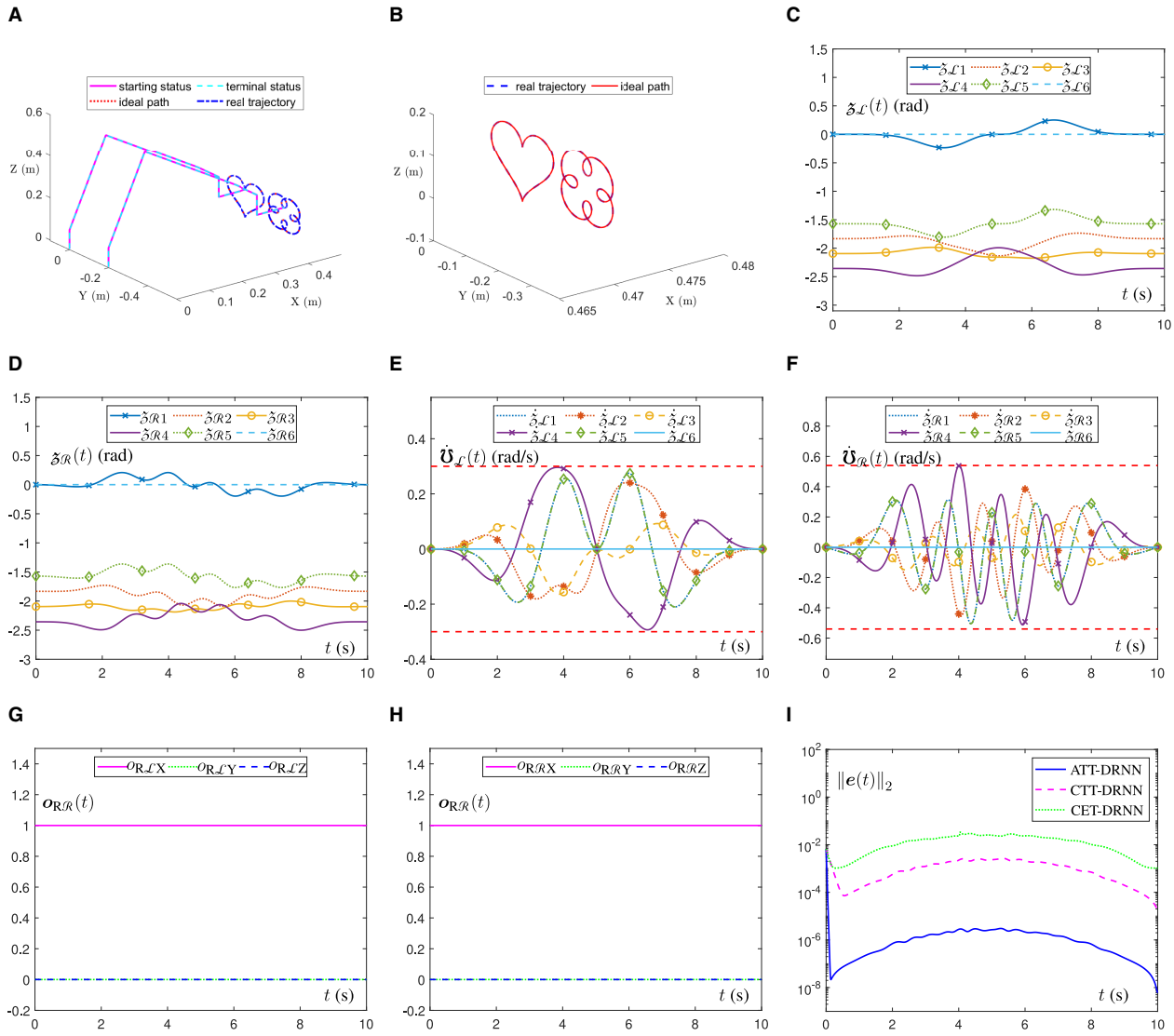


FIGURE 6
The ATT-DRNN algorithm (43) controls the posture collaboration motion of the UR5 dual-arm robot with two arms placed on the identical side for the heart-shaped and auspicious cloud trajectory tracking. **(A)** Starting and terminal statuses of the dual-arm robot and the end-executor real and ideal trajectories in a 3D space. **(B)** Outlines of real trajectory and ideal path in a 3D space. **(C)** Variations of the LA joint angles $\mathfrak{z}_{\mathcal{L}}(t)$. **(D)** Variations of the RA joint angles $\mathfrak{z}_{\mathcal{R}}(t)$. **(E)** Variations of the LA joint velocities $\dot{\mathfrak{z}}_{\mathcal{L}}(t)$. **(F)** Variations of the RA joint velocities $\dot{\mathfrak{z}}_{\mathcal{R}}(t)$. **(G)** Variations of the LA end-executor orientation $o_{\mathrm{R}\mathcal{L}}(t)$. **(H)** Variations of the RA end-executor orientation $o_{\mathrm{R}\mathcal{R}}(t)$. **(I)** The RE $\|e(t)\|_2$ generated by three different algorithms.

the ATT-DRNN algorithm (43) controls the dual-arm robot to realize the posture collaboration motion and accomplish the task of tracking the heart-shaped and auspicious cloud trajectories separately. Subsequently, Figures 6E, F delineate the joint-velocity profiles of the left and right robotic arms. It is apparent that none of the joint-velocity values exceed the predetermined physical limits initially determined. Besides, the end-executor orientation variation curves are shown in Figures 6G, H, which maintain a constant state during the task execution. Furthermore, in Figure 6I, the RE $\|e(t)\|_2$ generated by the ATT-DRNN algorithm (43) maintains at approximately 10^{-6} and converges to a extremely small value of 10^{-8}. By contrast, the RE $\|e(t)\|_2$ generated by the CTT-DRNN algorithm (36) keeps at roughly 10^{-3} and converges to about 10^{-5} and that generated by the CET-DRNN algorithm in Wu and Zhang (2023) can merely converge to about 10^{-3}.

In addition, Figure 7 shows the position error variation curves of the end-executor when the UR5 dual-arm robot tracks the heart-shaped and auspicious cloud trajectory under the control of different algorithms. In Figure 7A, the dual-arm robot controlled by the ATT-DRNN algorithm (43) can accomplish the trajectory following task accurately with the maximal position error of the end-executor being less than 1.0×10^{-6} m. In Figure 7B, the dual-arm robot controlled by the CTT-DRNN algorithm (36) can realize the maximal position error of the end-executor no more than 1.2×10^{-4} m. In Figure 7C, the dual-arm robot controlled by the CET-DRNN algorithm in Wu and Zhang (2023) can merely ensure that the position error of the end-executor is within 1.8×10^{-3} m.

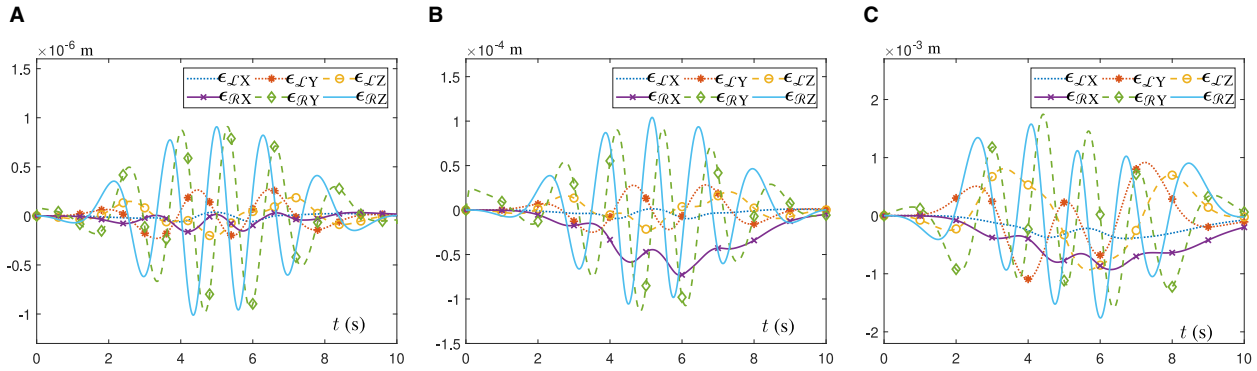


FIGURE 7
Variations of the end-executor position error when the UR5 dual-arm robot with two arms placed on the identical side to achieve the posture collaboration motion for the heart-shaped and auspicious cloud trajectory tracking. **(A)** Generated by the ATT-DRNN algorithm (43). **(B)** Generated by the CTT-DRNN algorithm (36). **(C)** Generated by the CET-DRNN algorithm (Wu and Zhang, 2023).

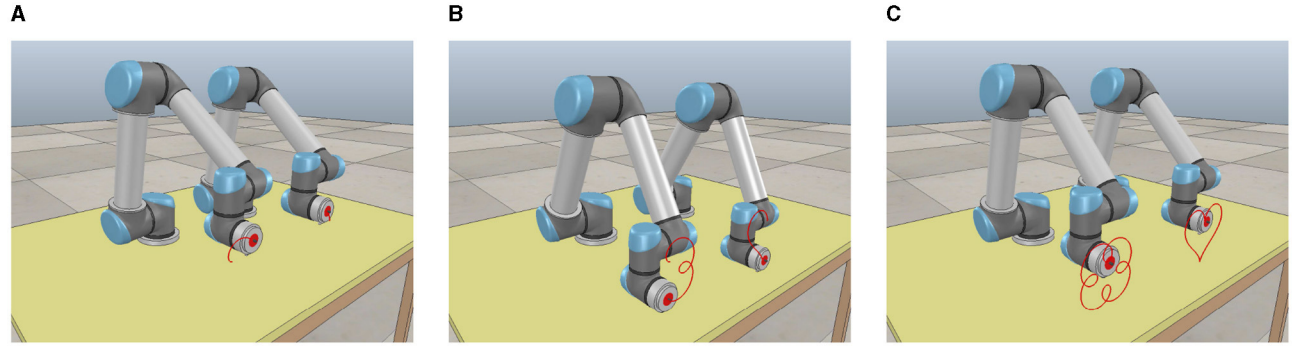


FIGURE 8
Snapshots of the UR5 dual-arm robot with two arms placed on the identical side during the trajectory tracking task of the heart-shaped and auspicious cloud path via the ATT-DRNN algorithm (43) solving the DAPCMC scheme (27)-(29). **(A)** Capturing at the starting moment. **(B)** Capturing at the intermediate moment. **(C)** Capturing at the terminal moment.

To further simulate the movement status of the dual-arm robot vividly and intuitively in the physical scene, we utilize the virtual robot experiment platform to show the real-time status of the dual-arm robot following the ideal paths with the help of the ATT-DRNN algorithm (43) in solving the DAPCMC scheme (27)–(29). Snapshots describing the movement process (i.e., starting moment, intermediate moment, and terminal moment) are shown in Figure 8.

In summary, the aforementioned two control cases of dual-arm robots substantiate that the proposed DAPCMC scheme (27)–(29) and its corresponding ATT-DRNN algorithm (43) can be utilized for the posture collaboration control of the industrial robots with joint physical limits considered and further demonstrate the potential of the proposed scheme and algorithm to optimize the efficiency and precision of repetitive trajectory tracking in practical applications.

5 Conclusion

In this study, the ATT-DRNN algorithm (43) has been devised for solving the DAPCMC scheme (27)–(29) with a novel JLCS (7), (8). Additionally, theoretical analyses and results have indicated the excellent performance of the ATT-DRNN algorithm (43) and the ACRNN model (41) in terms of the convergence rate and precision. Then, three illustrative examples with comparisons have further demonstrated that the proposed DAPCMC scheme (27)–(29) in a 3D space with the innovative JLCS (7), (8) offers a new solution measure for realizing the posture collaboration motion control of constrained dual-arm robots and accomplishing repetitive trajectory following missions, and it can be worked out by the ATT-DRNN algorithm (43) efficiently and accurately.

Finally, some possible research directions in the future are put forward.

- The whole design process of the ATT-DRNN algorithm (43) is set in an ideal noiseless environment. Therefore, enhancing the ATT-DRNN algorithm (43) with relevant anti-noise technologies to make it possess strong robustness in various noise environments is an interesting future research direction.
- The ATT-DRNN algorithm (43) involves an explicit inverse operation, which is computationally expensive. Thus,

proposing an inverse-free ATT-DRNN algorithm is another future research direction.

- Two robotic arms of the same model are used to form a dual-arm robot in this study. Thus, achieving the collaboration motion control by composing a heterogenous multi-arm robot system is a meaningful future research direction.
- Popularizing the ATT-DRNN design scheme to more kinds of engineering applications (e.g., UAV flight control) is also a significant future research direction.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

YZ: Data curation, Formal analysis, Software, Validation, Visualization, Writing – original draft. YH: Funding acquisition, Project administration, Resources, Supervision, Writing – review & editing. BQ: Conceptualization, Funding acquisition, Investigation, Methodology, Resources, Supervision, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFC3011105, in part by the National Natural Science Foundation of China under Grant 62006254, and in part by the Shenzhen Outbound Postdoctoral Program under Grant SZBH202127.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Arents, J., and Greitans, M. (2022). Smart industrial robot control trends, challenges and opportunities within manufacturing. *Appl. Sci.-Basel* 12:937. doi: 10.3390/app12020937

Bombile, M., and Billard, A. (2022). Dual-arm control for coordinated fast grabbing and tossing of an object: proposing a new approach. *IEEE Robot. Autom. Mag.* 29, 127–138. doi: 10.1109/MRA.2022.3177355

Cai, J., and Yi, C. (2023). An adaptive gradient-descent-based neural networks for the on-line solution of linear time variant equations and its applications. *Inf. Sci.* 622, 34–45. doi: 10.1016/j.ins.2022.11.157

Cheng, Z., Feng, W., Zhang, Y., Sun, L., Liu, Y., Chen, L., et al. (2023). A highly robust amphibious soft robot with imperceptibility based on a water-stable and self-healing ionic conductor. *Adv. Mater. Weinheim.* 13:2301005. doi: 10.1002/adma.202301005

Chico, A., Cruz, P. J., Vásconez, J. P., Benalcázar, M. E., Álvarez, R., Barona, L., et al. (2021). "Hand gesture recognition and tracking control for a virtual UR5 robot manipulator," in *2021 IEEE Fifth Ecuador Technical Chapters Meeting (ETCM)* (Cuenca, Ecuador: IEEE), 1–6.

Dai, J., Chen, Y., Xiao, L., Jia, L., and He, Y. (2022). Design and analysis of a hybrid GNN-ZNN model with a fuzzy adaptive factor for matrix inversion. *IEEE Trans. Ind. Inform.* 18, 2434–2442. doi: 10.1109/TII.2021.3093115

Ekrem, Ö., and Aksoy, B. (2023). Trajectory planning for a 6-axis robotic arm with particle swarm optimization algorithm. *Eng. Appl. Artif. Intell.* 122:106099. doi: 10.1016/j.engappai.2023.106099

Fu, Z., Zhang, Y., and Tan, N. (2023). Gradient-feedback ZNN for unconstrained time-variant convex optimization and robot manipulator application. *IEEE Trans. Ind. Inform.* 19, 10489–10500. doi: 10.1109/TII.2023.3240737

Hu, C., Kang, X., and Zhang, Y. (2018). Three-step general discrete-time Zhang neural network design and application to time-variant matrix inversion. *Neurocomputing* 306, 108–118. doi: 10.1016/j.neucom.2018.03.053

Isidori, A. (1989). *Nonlinear Control Systems: An Introduction*. Springer-Verlag, Berlin: Germany.

Jiang, Y., Wang, Y., Miao, Z., Na, J., Zhao, Z., and Yang, C. (2022). Composite-learning-based adaptive neural control for dual-arm robots with relative motion. *IEEE Trans. Neural Netw. Learn. Syst.* 33, 1010–1021. doi: 10.1109/TNNLS.2020.3037795

Jin, L., Liu, L., Wang, X., Shang, M., and Wang, F.-Y. (2024). Physical-informed neural network for MPC-based trajectory tracking of vehicles with noise considered. *IEEE Trans. Intell. Veh.* 9, 4493–4503. doi: 10.1109/TIV.2024.3358229

Kastritsi, T., and Doulgeri, Z. (2021). A controller to impose a RCM for hands-on robotic-assisted minimally invasive surgery. *IEEE Trans. Med. Robot. Bionics* 3, 392–401. doi: 10.1109/TMRB.2021.3077319

Khan, Z. H., Siddique, A., and Lee, C. W. (2020). Robotics utilization for healthcare digitization in global COVID-19 management. *Int. J. Environ. Res. Public Health* 17:3819. doi: 10.3390/ijerph17113819

Li, W. (2020). Predefined-time convergent neural solution to cyclical motion planning of redundant robots under physical constraints. *IEEE Trans. Ind. Electron.* 67, 10732–10743. doi: 10.1109/TIE.2019.2960754

Li, W., Chiu, P. W. Y., and Li, Z. (2023). A novel neural approach to infinity-norm joint-velocity minimization of kinematically redundant robots under joint limits. *IEEE Trans. Neural Netw. Learn. Syst.* 34, 409–420. doi: 10.1109/TNNLS.2021.3095122

Liao, B., Zhang, Y., and Jin, L. (2016). Taylor $O(h^3)$ discretization of ZNN models for dynamic equality-constrained quadratic programming with application to manipulators. *IEEE Trans. Neural Netw. Learn. Syst.* 27, 225–237. doi: 10.1109/TNNLS.2015.2435014

Liu, M., Luo, W., Cai, Z., Du, X., Zhang, J., and Li, S. (2023a). Numerical-discrete-scheme-incorporated recurrent neural network for tasks in natural language processing. *CAAI Trans. Intell. Technol.* 8, 1415–1424. doi: 10.1049/cit2.12172

Liu, M., Wu, H., Shi, Y., and Jin, L. (2023b). High-order robust discrete-time neural dynamics for time-varying multi-linear tensor equation with M-tensor. *IEEE Trans. Ind. Inform.* 19, 9457–9467. doi: 10.1109/TII.2022.3228394

Liufu, Y., Jin, L., Shang, M., Wang, X., and Wang, F.-Y. (2024). ACP-incorporated perturbation-resistant neural dynamics controller for autonomous vehicles. *IEEE Trans. Intell. Veh.* doi: 10.1109/TIV.2023.3348632

McCartney, G., and McCartney, A. (2020). Rise of the machines: Towards a conceptual service-robot research framework for the hospitality and tourism industry. *Int. J. Contemp. Hosp. Manag.* 32, 3835–3851. doi: 10.1108/IJCHM-05-2020-0450

Qiu, B., Guo, J., Yang, S., Yu, P., and Tan, N. (2023). A novel discretized ZNN model for velocity layer weighted multicriteria optimization of robotic

manipulators with multiple constraints. *IEEE Trans. Ind. Inform*. 19, 6717–6728. doi: 10.1109/TII.2022.3197270

Shi, Y., and Zhang, Y. (2018). Discrete time-variant nonlinear optimization and system solving via integral-type error function and twice ZND formula with noises suppressed. *Soft Comput*. 22, 7129–7141. doi: 10.1007/s00500-018-3020-5

Shi, Y., Zhao, W., Li, S., Li, B., and Sun, X. (2023). Novel discrete-time recurrent neural network for robot manipulator: A direct discretization technical route. *IEEE Trans. Neural Netw. Learn. Syst*. 34, 2781–2790. doi: 10.1109/TNNLS.2021.3108050

Song, Q., Wu, Y., and Soh, Y. C. (2008). Robust adaptive gradient-descent training algorithm for recurrent neural networks in discrete time domain. *IEEE Trans. Neural Netw*. 19, 1841–1853. doi: 10.1109/TNN.2008.2001923

Stolfi, A., Gasbarri, P., and Sabatini, M. (2017). A combined impedance-PD approach for controlling a dual-arm space manipulator in the capture of a non-cooperative target. *Acta Astronaut*. 139, 243–253. doi: 10.1016/j.actaastro.2017.07.014

Tanyıldızı, A. K. (2023). Design, control and stabilization of a transformable wheeled fire fighting robot with a fire-extinguishing, ball-shooting turret. *Machines* 11:492. doi: 10.3390/machines11040492

Vivas, A., and Sabater, J. M. (2021). "UR5 robot manipulation using Matlab/Simulink and ROS," in *2021 IEEE International Conference on Mechatronics and Automation (ICMA)* (Takamatsu: IEEE), 338–343.

Wang, S., Jin, L., Du, X., and Stanimirovi, P. S. (2021). Accelerated convergent zeroing neurodynamics models for solving multi-linear systems with *M*-tensors. *Neurocomputing* 458, 271–283. doi: 10.1016/j.neucom.2021.06.005

Wang, Y., Li, H., Zhao, Y., Chen, X., Huang, X., and Jiang, Z. (2023). A fast coordinated motion planning method for dual-arm robot based on parallel constrained DDP. *IEEE-ASME Trans. Mechatron*. doi: 10.1109/TMECH.2023.3323798

Wei, L., Jin, L., and Luo, X. (2022). Noise-suppressing neural dynamics for time-dependent constrained nonlinear optimization with applications. *IEEE Trans. Syst., Man, Cybern., Syst*., 52, 6139–6150. doi: 10.1109/TSMC.2021.3138550

Wu, W., and Zhang, Y. (2023). Novel adaptive zeroing neural dynamics schemes for temporally-varying linear equation handling applied to arm path following and target motion positioning. *Neural Netw*. 165, 435–450. doi: 10.1016/j.neunet.2023.05.056

Xiao, L., Dai, J., Jin, L., Li, W., Li, S., and Hou, J. (2021). A noise-enduring and finite-time zeroing neural network for equality-constrained time-varying nonlinear optimization. *IEEE Trans. Syst. Man Cybern. Syst*. 51, 4729–4740. doi: 10.1109/TSMC.2019.2944152

Yan, J., Jin, L., Yuan, Z., and Liu, Z. (2022). RNN for receding horizon control of redundant robot manipulators. *IEEE Trans. Ind. Electron*. 69, 1608–1619. doi: 10.1109/TIE.2021.3062257

Yang, M., Zhang, Y., and Hu, H. (2021). Posture coordination control of two-manipulator system using projection neural network. *Neurocomputing* 427, 179–190. doi: 10.1016/j.neucom.2020.11.012

Yang, M., Zhang, Y., Zhang, Z., and Hu, H. (2020). Adaptive discrete ZND models for tracking control of redundant manipulator. *IEEE Trans. Ind. Inform*. 16, 7360–7368. doi: 10.1109/TII.2020.2976844

Yang, Q., Du, X., Wang, Z., Meng, Z., Ma, Z., and Zhang, Q. (2023). A review of core agricultural robot technologies for crop productions. *Comput. Electron. Agric*. 206:107701. doi: 10.1016/j.compag.2023.107701

Yang, S., Wen, H., Hu, Y., and Jin, D. (2020). Coordinated motion control of a dual-arm space robot for assembling modular parts. *Acta Astronaut*. 177, 627–638. doi: 10.1016/j.actaastro.2020.08.006

Zhang, Y., and Zhang, Z. (2013). *Repetitive Motion Planning and Control of Redundant Robot Manipulators*. New York: Springer Science & Business Media, Springer-Verlag.

Zhang, Z., Lin, Y., Li, S., Li, Y., Yu, Z., and Luo, Y. (2018). Tricriteria optimization-coordination motion of dual-redundant-robot manipulators for complex path planning. *IEEE Trans. Control Syst. Technol*. 26, 1345–1357. doi: 10.1109/TCST.2017.2709276

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 04 June 2024
DOI 10.3389/fnbot.2024.1406658

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Yiqin Deng,
Shandong University, China
Chenglin Li,
South China Agricultural University, China

*CORRESPONDENCE
Bangjie Li
ro_soldier@outlook.com

RECEIVED 25 March 2024
ACCEPTED 20 May 2024
PUBLISHED 04 June 2024

CITATION
Wu J, Li H, Li B, Zheng X and Zhang D (2024) Optimization of robotic path planning and navigation point configuration based on convolutional neural networks.
Front. Neurorobot. 18:1406658.
doi: 10.3389/fnbot.2024.1406658

Optimization of robotic path planning and navigation point configuration based on convolutional neural networks

Jian Wu[1], Huan Li[2], Bangjie Li[1]*, Xiaolong Zheng[1] and Daqiao Zhang[1]

[1]Xi'an Institute of High-Tech, Xi'an, China, [2]School of Software, Xinjiang University, Urumqi, China

This study introduces a novel approach for enhancing robotic path planning and navigation by optimizing point configuration through convolutional neural networks (CNNs). Faced with the challenge of precise area coverage and the inefficiency of traditional traversal and intelligent algorithms (e.g., genetic algorithms, particle swarm optimization) in point layout, we proposed a CNN-based optimization model. This model not only tackles the issues of speed and accuracy in point configuration with Gaussian distribution characteristics but also significantly improves the robot's capability to efficiently navigate and cover designated areas with high precision. Our methodology begins with defining a coverage index, followed by an optimization model that integrates polygon image features with the variability of Gaussian distribution. The proposed CNN model is trained with datasets generated from systematic point configurations, which then predicts optimal layouts for enhanced navigation. Our method achieves an experimental result error of <8% on the test dataset. The results validate effectiveness of the proposed model in achieving efficient and accurate path planning for robotic systems.

KEYWORDS

robotic path planning, precise area coverage, optimized point configuration, convolutional neural networks, navigation

1 Introduction

In the domain of robotic path planning and navigation, the strategic configuration of points, particularly those with Gaussian distribution characteristics within polygonal areas, is fundamental to enhancing efficiency and precision in covering specific regions (Chao et al., 2023; Cui et al., 2023; Wu X. et al., 2023). The concept of coverage fraction is pivotal in evaluating the effectiveness of these configurations. Due to the inherent randomness and the possibility of overlapping point coverage, the direct derivation of analytical models for calculating the coverage fraction proves to be a formidable challenge. This complexity necessitates a shift toward discretization, transforming the continuous challenge into a discrete problem that can be methodically approximated. Traditional methods, such as sequential traversal, while methodical, are markedly inefficient and unable to satisfy the exigencies of rapid, real-time decision-making essential in autonomous navigation. Lu et al. (2019) and Jie et al. (2023) reveal attempts to address these limitations, with genetic algorithms and particle swarm optimization offering efficiency improvements.

However, these methods still suffer from significant drawbacks, including sensitivity to computation times and unpredictability in outcomes, which hinder their applicability in scenarios demanding high precision and responsiveness. Addressing these challenges, our research plays a pivotal role in the exploitation of deep learning networks (Simsekli et al., 2019), which is renowned for their robust learning capacities and adaptability. This study unfolds in progressive stages: it commences with establishing an optimal point configuration algorithm derived from the traversal methodology (Luo et al., 2020; Hu et al., 2021; Heßler and Irnich, 2022). It then advances to devising a method for generating random polygons, analyzing the elements that impact traversal searches, and determining the optimal traversal strides for generating models for optimal point configuration. The cornerstone of our approach is the application of feature extraction and dimensionality reduction techniques on the conditions triggering configuration and facilitating the creation of a convolutional neural network (CNN)-based model for point configuration (Jing et al., 2022). This model is meticulously trained on a dataset designed to reflect various polygon shapes and configurations, paving the way for an innovative approach to path planning and navigation. Our findings, which are supported by simulation and practical implementation, demonstrate the model's unparalleled effectiveness and efficiency. The CNN-based approach significantly outperforms the traversal engineering algorithm, genetic algorithm, and particle swarm optimization (Langazane and Saha, 2022; Aote et al., 2023) in terms of speed, accuracy, and real-time adaptability, heralding a new era in robotic navigation. By optimizing point configuration through deep learning, robots can now navigate and cover specific areas with unprecedented precision, marking a milestone in the quest for advanced robotic path planning and navigation solutions. This introduction sets the stage for a detailed exploration of our methodology, the neural network model, and the profound implications of our study in the broader context of robotics and autonomous systems. The contributions of this study are as follows:

1. We proposed an optimal point configuration algorithm derived from the traversal methodology.
2. We proposed a method for generating random polygons, analyzing the elements that impact traversal searches, and determining the optimal traversal strides for generating models for optimal point configuration.
3. We use a convolutional neural network (CNN)-based model for point configuration and application of feature extraction and dimensionality reduction techniques on the conditions triggering configuration.

The structure of this study is as follows: In Section 2, we review the work related to this study. In Sections 3–5, we describe our proposed algorithm in detail. In Section 6, we report the simulation realization and result analysis.

2 Related work

There are many path planning methods, and their application ranges vary according to their own advantages and disadvantages. Based on the study of commonly used path-planning algorithms in various fields, the algorithms are classified into four categories according to the sequence of discovery and the basic principles of the algorithms: traditional algorithms, graphical algorithms, intelligent bionic algorithms, and other algorithms.

2.1 Traditional algorithms

Traditional path planning algorithms include simulated annealing (SA) algorithms and artificial potential field algorithms.

SA (Wang et al., 2018) algorithm is an efficient approximation algorithm for large-scale combinatorial optimization problems. It uses the neighborhood structure of the solution space to perform a stochastic search. It has the advantages of simple description, flexible use, high operation efficiency, and less restriction of initial conditions, but it has the defects of slow convergence and randomness.

The artificial potential field (Khatib, 1986; Sciavicco and Siciliano, 1988) algorithms imitate the motion of objects under gravitational repulsion and perform path optimization by establishing the gravitational field repulsive field function. The advantage is that the planned path is a smooth and safe simple description, but there is the problem of local optimization.

2.2 Graphical algorithms

Traditional algorithms often have the problem of difficult modeling when solving real problems, and graphical methods provide the basic method of modeling, but graphical methods generally have a lack of search capability and often need to be combined with specialized search algorithms. Graphical algorithms include C-space algorithms and grid algorithms.

C-space algorithms (Yu and Gupta, 2004) expand the obstacles as polygons in the motion space and search for the shortest path by taking the start point, the endpoint, and the feasible straight line between all the vertices of the polygons as the range of the path. The advantage of the c-space algorithm over the spatial method is that it is intuitive and easy to find the shortest path; the disadvantage is that once the start point and the goal point are changed, it is necessary to re-construct the viewable graph, which is a lack of flexibility.

Grid algorithm (Mansor and Morris, 1999) is to use encoded raster to represent the map; the raster containing obstacles is labeled as an obstacle raster, and vice versa is a free raster, which is used as the basis for path search. The Grid algorithm is generally used as an environmental modeling technique for path planning, and it is difficult to solve the problem of complex environmental information as a path planning method.

2.3 Intelligent bionics algorithm

When dealing with path planning problems in the case of complex dynamic environmental information, revelations from nature can often play a good role. Intelligent bionics algorithms are algorithms discovered through bionic research and commonly used

ones include ant colony algorithms, neural network algorithms, particle swarm algorithms, and genetic algorithms.

Ant colony algorithm (Wu L. et al., 2023) achieves its goal by iterating to simulate the behavior of ant colony foraging. It has the advantages of good global optimization ability, intrinsic parallelism, and ease of implementation by computer, but it is computationally intensive and easy to fall into local optimal solutions, although it can be improved by adding elite ants and other methods.

Neural network algorithm (Nair and Supriya, 2020; Yu et al., 2020) is an excellent algorithm in the field of artificial intelligence, but its application in path planning is not successful because the complex and changing environment in path planning is difficult to be described by mathematical formulas. Although neural networks have excellent learning ability, poor generalization ability is its fatal flaw. However, because of its strong learning ability and good robustness, its combined application with other algorithms has become a hot research topic in the field of path planning.

Genetic Algorithm (GA) (Shao, 2021; Luan and Thinh, 2023) is an important research branch of contemporary artificial intelligence science. It is an iterative process search algorithm realized according to the principle of genetics. The biggest advantage is that it is easy to combine with other algorithms and give full play to its own iterative advantages, and the disadvantage is that the computational efficiency is not high.

Particle Swarm Optimization (Das and Jena, 2020; Zhang et al., 2020) is an iterative algorithm that simulates the behavior of birds in flight. Similar to the genetic algorithm, it starts from a random solution and iteratively searches for an optimal solution, but it has simpler rules than the genetic algorithm, and it does not have the "crossover" and "mutation" operations of the genetic algorithm. It searches for the global optimum by following the currently searched optimal value. It has the advantages of a simple algorithm, easy to implement, good robustness, not very sensitive to the size of the population, and fast convergence, but it is easy to fall into the local optimal solution.

3 Calculation of fraction of coverage based on polygon discretization

3.1 Definition of fraction of coverage

When points with Gaussian distribution are arranged in any polygon, and the measurement index is selected as the polygon fraction of coverage, the calculation result of the index is affected by the Gaussian distribution characteristics of the points, the size of the polygon, and the control range of the Gaussian points and other factors (Chen et al., 2021).

The calculation of point coverage is influenced by the randomness of the points. The method for calculating the point coverage rate is shown in Equation (1), and the joint probability density distribution function is shown in Equation (2). Given the influence of repeated coverage, it is difficult to directly solve the polygon fraction of coverage using the analytical method so that it can be calculated by discretization.

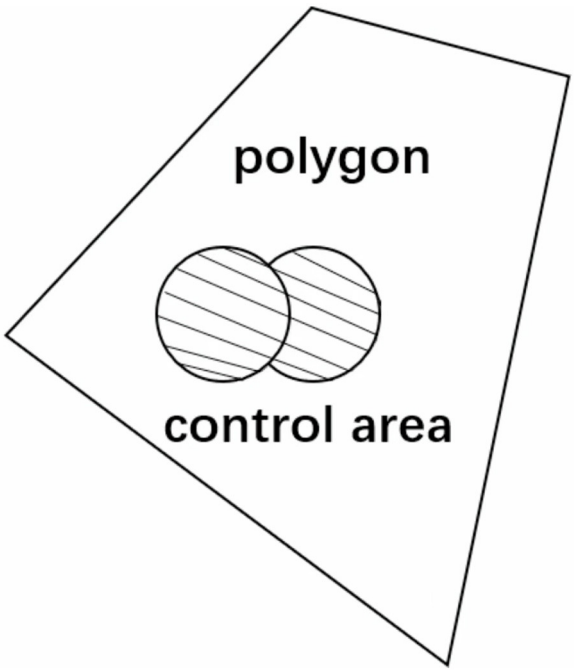


FIGURE 1
Schematic diagram of Coverage Index Calculation.

$$p = \frac{s_f}{s} \tag{1}$$

where s_f refers to the point coverage, which represents the area of the intersection area between the shadow part and the polygon in the schematic diagram; s is the polygon area. The Schematic Diagram of Coverage Index Calculation is shown in Figure 1.

$$f(x,z) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x-m_x)^2+(z-m_z)^2}{2\sigma^2}\right) \tag{2}$$

where m_x, m_z represent the point configuration coordinates and σ represents the root mean square error of point Gaussian distribution.

3.2 Polygon discretization

For a polygon with an arbitrary shape, taking the lower left vertex as the coordinate origin and any one of the two sides connected with the change point as the Z-axis, a Cartesian coordinate system is established to discretize the polygon (Lei et al., 2020), and then, the coordinate calculation of any grid center point of the polygon outer envelope rectangle is shown in Equation (3).

$$\begin{cases} x_{ij} = x_{\min} + \frac{x_{\max}-x_{\min}}{n \cdot i} \\ z_{ij} = z_{\min} + \frac{z_{\max}-z_{\min}}{m \cdot j} \\ (i = 1, 2, \ldots, n; j = 1, 2, \ldots, m) \end{cases} \tag{3}$$

where n, m represent the number of discrete polygons in x and z directions; $x_{\min}$, $x_{\max}$ represent the boundary range of the polygon

in the x direction; $z_{\min}$, $z_{\max}$ represent the boundary range of the polygon in the z direction.

The ray method is used to judge all the discrete points of the polygon outer envelope rectangle, in turn, and then, the polygon can be discretized (Cheng et al., 2019; Chappell et al., 2021; Siyu et al., 2022). Steps to determine whether a point is within a polygon based on the ray method:

- Step 1: Taking the center coordinate of the discrete grid point as the starting point, the ray is made along any direction, and the intersection relationship between the ray and the line segment composed of two adjacent points of the polygon is judged, in turn.
- Step 2: If the discrete grid point is on the polygon vertex, it is judged that the grid point is inside the polygon; if the ray and the line segment overlap, it is judged that the grid point is inside the polygon; regarding the odd and even case of the number of intersection points, if it is odd, the point is inside the polygon, and otherwise, the point is outside the polygon.

3.3 Calculation of fraction of coverage

Based on the polygon discretization, the probability of the grid is approximately replaced by the probability of the grid center point. Therefore, when n points with Gaussian distribution are arranged within any polygon, the coverage probability of the kth point for any discrete point of the polygon is calculated as shown in Equation (4), and a given point is the configuration result (Kamra and Liu, 2021), the fraction of coverage of n points for this grid point is calculated as shown in Equation (5).

$$\begin{cases} p_{ij}^{k} = \iint\limits_{J \le d^2} f(x,z)\,dx\,dz \\ J(x,z) = (x_{ij} - x^2) + (z_{ij} - z)^2 \end{cases} \tag{4}$$

where d represents the control range of the point, x_{ij} and z_{ij} denote the x and z coordinates of the discrete point, respectively.

$$p_{ij} = 1 - \prod_{k=1}^{n} (1 - p_{\mathrm{ij}}^{k}) \tag{5}$$

where p_{ij}^{k} represents the coverage of the Kth point and n represents n points with Gaussian distribution within the polygon.

4 Sample generation model of optimal point configuration

4.1 Sample generation process of optimal point configuration

Given the input polygons, the Gaussian distribution characteristics of the points, configuration information, and control range, we can use Equations (3–5) to calculate the fraction of coverage. The calculation is carried out by traversing all the point combinations on the inner grid points of the polygon, in turn, so that the maximum fraction of coverage and the corresponding point configuration results can be obtained; that is, a deep learning training sample can be obtained. Many training samples for point configuration can be generated by introducing variations in polygon shapes, Gaussian distribution characteristics, and control range. The sample generation process is shown in Figure 2.

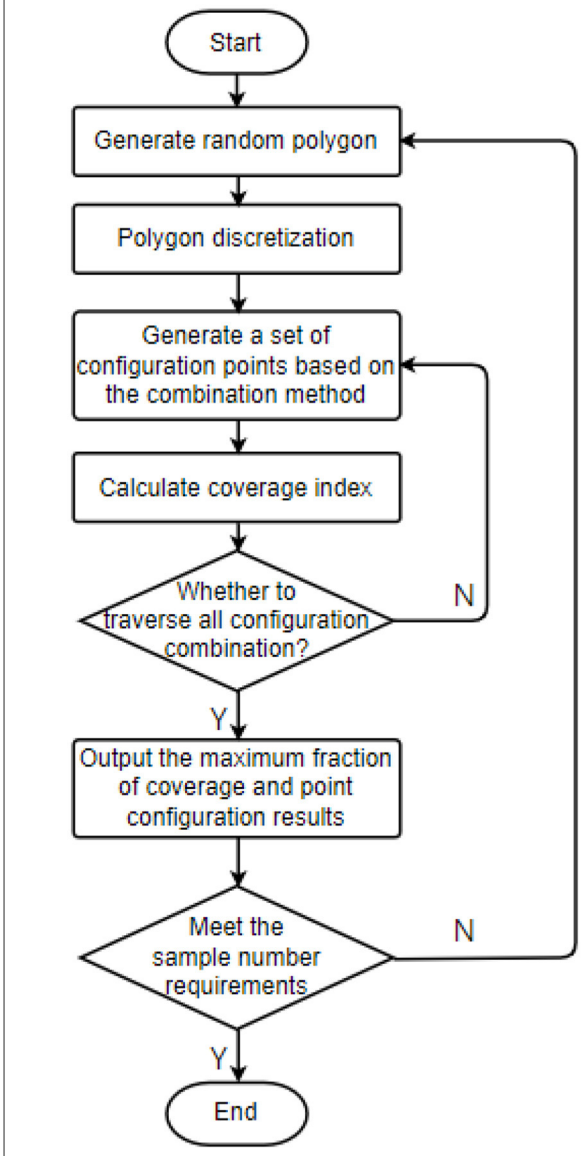


FIGURE 2
Sample generation process of point configuration.

4.2 Random polygon generation

In the region m *times* n, several random points are constructed according to the polygon edge number by using the coordinates of random function generation points (De Goes et al., 2020). Equation

(6) was used to generate the coordinates of the polygon's vertices randomly.

$$\begin{cases} x_i = m \cdot rand \\ z_i = n \cdot rand \end{cases} \tag{6}$$

where rand represents a function of a random number with a probability density between [0, 1] that follows a uniform distribution.

Polygon construction: Take the point with the largest X coordinate as the starting point (x_0, z_0). If there is more than one maximum X coordinate, take the point with the smallest Z coordinate, calculate the tangent value of the included angle between the connecting line between the (x_0, z_0) point and other points and the horizontal line in turn, then sort by the tangent value, and label the points in turn to form a random polygon.

TABLE 1 Test of coordinates of polygon vertices.

SN	x	z	SN	x	z
1	4.755	1.545	5	−38.907	18.308
2	7.598	4.822	6	−50	0.2153
3	−1.63	25.948	7	−4.646	−73.853
4	−20.572	28.315	8	24.721	−76.084

TABLE 2 Test results under $\sigma = 5m$ and $R = 20m$.

Traversal stride	0.001 n	0.005 n	0.01 n
Point configuration results	(−16, −27)	(−20, −15)	(−20, −30)
Fraction of coverage	0.2464	0.2464	0.2464
Time consumed for calculation	2.835 s	143 s	33 s

4.3 Traversal stride optimization

4.3.1 Traversal stride optimization process

With the increase in the number of configuration points, the number of traversal searches increases exponentially, and the grid discretization method leads to the low efficiency of sample generation in the index calculation. While generating data sets by running code, the traversal size can be optimized according to the Gaussian distribution characteristics of points. The calculation process of Equations (4, 5) shows that the calculation of the fraction of coverage is influenced by polygon size, point control radius, and mean square error and can be changed proportionally, such as point configuration coordinates, polygon vertex coordinates, point control radius, and mean square error are all expanded by 10 times. The calculation result of a fraction of coverage is the same as the original condition. Therefore, during the optimization of the traversal stride, the outer boundary size of the polygon can be fixed, and the influence of point control radius and mean square error on traversal stride under this condition is studied, and the influence law under general conditions is extended.

In this study, when the mean square error is different, by changing the traversal stride and comparing the code running efficiency and the point configuration accuracy under different traversal strides, an appropriate stride is determined to meet the requirements of both running efficiency and accuracy within an acceptable range. The test process is as follows:

- Step 1: Randomly generate a polygon in a fixed outer boundary range $n \times n$;
- Step 2: Take the typical control radius R and the typical mean square error σ; the calculation method is shown in Equation (7);

$$\begin{cases} R = k \times \frac{n}{100} (k = 1, 2, \ldots, 50) \\ \sigma = k \times \frac{n}{100} (k = 1, 2, \ldots, 50) \end{cases} \tag{7}$$

- Step 3: Calculate the results of optimal configuration points under the conditions of configuring one point, two points, three points, and four points;
- Step 4: Compare the calculation results of the configuration ratio with the typical traversal stride. The calculation method is shown in Equation (8);

$$d = k \times \frac{n}{100} (k = 1, 2, \ldots, 50) \tag{8}$$

- Step 5: Determine the optimization criterion of stride by polynomial fitting calculation and obtain the function form of Equation (9).

$$d = f\left(R, \sigma, n_f, number_m\right) \tag{9}$$

4.3.2 Determination of traversal stride criteria

Simulation conditions: Randomly generate an octagon in the range of 200 m × 200 m for the experiment, and the coordinates of polygon vertices are shown in Table 1.

When the selected traversal strides are d = 0.001 n, d = 0.005 n, and d = 0.01 n, respectively, the results obtained with these parameter values are shown in Table 2.

The stride determination criteria are obtained by fitting the calculation of Equation (9). In conclusion, the calculation efficiency can be improved by optimizing the traversal stride when the engineering algorithm generates data sets. A large number of data about the coordinates of polygon vertices, optimal configuration point coordinates, and the fraction of coverage can be calculated through engineering algorithms. These data can be used as annotation data of pictures for deep learning. After traversal calculation, data about polygon graphics, configuration point coordinates, and the fraction of coverage will be obtained, which will be stored and used as sample sets for deep learning model training.

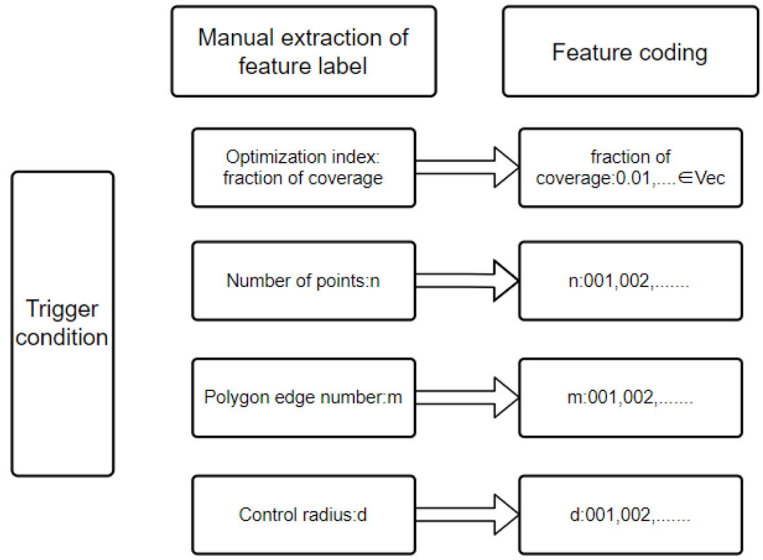


FIGURE 3
Schematic Diagram of trigger condition feature coding.

5 Optimization model of point configuration based on deep learning

5.1 Feature extraction of trigger conditions

The trigger conditions mainly include fraction of coverage index, number of points, polygon edge number, Gaussian distribution characteristics of points, and control radius, as shown in Figure 3. In this study, the trigger condition features are extracted manually. The characteristic labels with practical significance are extracted according to the characteristics of the problem. The manually extracted labels can help us better understand the practical significance of the trigger conditions. To facilitate the construction of training data sets, it is the basis and premise that the training model can accurately recommend scientific and reasonable point configuration by constructing a fixed-length input vector and using the spliced feature coding as the input of the deep neural network to help the model better understand the trigger conditions of the plan (Yang and Shami, 2020).

5.2 Construction of point configuration model based on deep neural network

Deep neural networks are mainly divided into three categories:

- *Fully connected deep neural network* is similar to the multilayer Perceptron network. The neurons between hidden layers transmit information in the form of full connection, and the number of layers of neural networks determines the learning ability of the model (Arora et al., 2019).
- *Deep convolutional neural network* mainly deals with data with strong spatial locality and uses convolution kernel as "medium"; the output vector dimension of each layer is determined by convolution kernel, which is mainly used in image recognition, target detection, image segmentation, and other fields (Elhassouny and Smarandache, 2019).
- *Recurrent neural network* has a strong spatial correlation in processing data, mainly used in fields such as speech recognition, machine translation, and video processing (Yu et al., 2019). In view of the feature coding of the trigger, and the condition is a one-dimensional vector extracted from text data, it has no time and space correlation. Therefore, this study builds a point configuration model by the fully connected deep neural network, in which the input is the feature vector extracted from the model, the output is the grid point of point configuration, and each neural network layer in the hidden layer is composed of many neurons. The output of this model is calculated using Equation (10).

$$\text{Output} = \text{ReLU}\left(\left[x_1 \;\; x_2 \;\; \cdots \;\; x_n\right] \cdot \vec{w} + \vec{b}\right) \tag{10}$$

5.3 Polygon dimension reduction processing based on deletion point approximation method

In the actual polygon point configuration, the polygon edge number may exceed the maximum limit of feature label design (Venkateswara Reddy et al., 2022). The idea of deletion point approximation is used to reduce the polygon dimension, assuming that the vertices of the polygon are represented by (V_0~V_n), and the realization process of polygon dimension reduction is as follows:

- Step 1: Calculate the polygon area by Ear Clipping;

  Definition: Ear Tip refers to the three consecutive vertices V_0, V_1, and V_2 of the polygon. If the connecting line V_0, V_2 is a diagonal of the polygon, V_1 is an Ear Tip. **Calculation of polygon area based on Ear Clipping;**

(a) Establish a bidirectional linked list V_0 of polygon vertices;

(b) Construct an initial convex vertex set **C** and a concave vertex set **R**, and construct an initial **Ear Tips set E**;

(c) **Delete one element V_i from Ear Tips set** and also delete from the vertex set in the polygon at a time, add the corresponding triangles $<V_{i-1},V_i,V_{i+1}>$ in the triangle linked list, update the temporary vertices, and calculate whether new convex vertices and **Ear Tip** are generated;

(d) Repeat Step 3 until only three vertices remain in the linked list.

- Step 2: Delete V_0, V_1, ...,V_n, in turn and calculate the polygon area after deleting nodes;
- Step 3: With the polygon area reduction ratio, calculated as shown in Equation (11) and sorted from small to large;

$$p = 1 - \frac{s_k}{s} \tag{11}$$

where p represents the reduction ratio of the polygon's area, s_k represents the area reduction ratio for the node being considered for deletion, and s represents the initial area of the polygon.

- Step 4: Delete the node with the smallest area reduction ratio and judge whether it meets the design requirements of feature label polygons, and if so, terminate the algorithm;
- Step 5: If the condition is not met, renumber the polygon after node deletion, and then repeat steps 2–4.

6 Simulation realization and result analysis

6.1 Simulation realization

The point configuration model based on convolutional neural network mainly includes two modules: polygon shape feature extraction module and regression fitting module.

TABLE 3 Module 1 convolutional neural network structure diagram.

	Number of input channels	Number of output channels	Convolution kernel
The first convolution layer	3	32	3 × 3 × 64
The second convolution layer	32	64	3 × 3 × 128
The third convolution layer	64	128	3 × 3 × 256
The fourth convolution layer	128	256	3 × 3 × 512

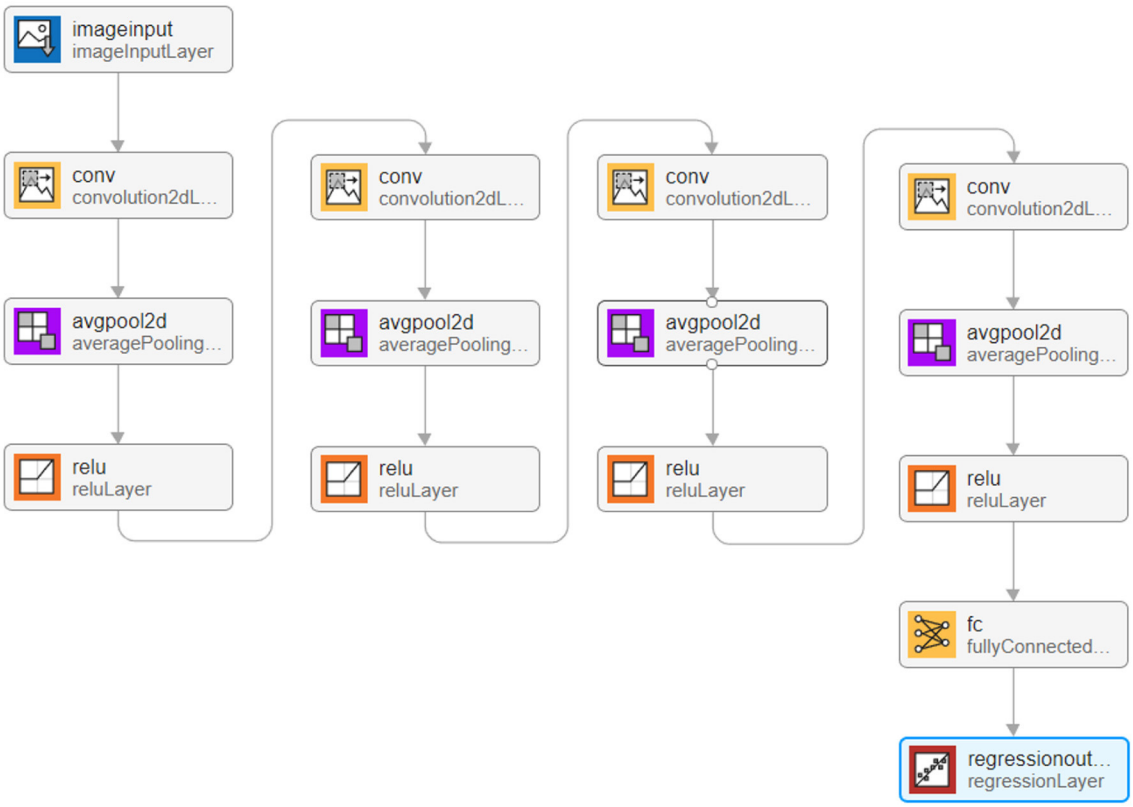


FIGURE 4
Schematic diagram of trigger condition feature coding.

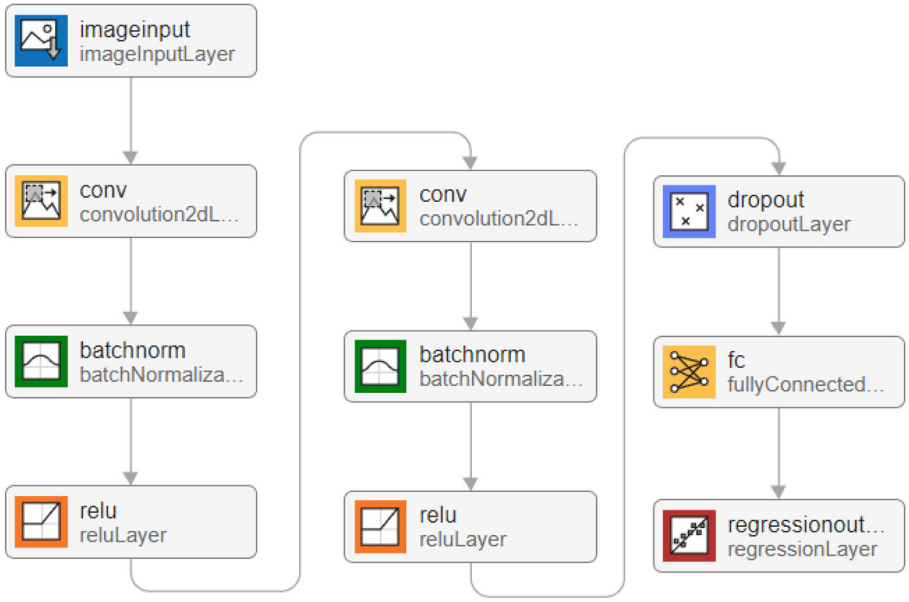


FIGURE 5
Network structure diagram of regression fitting module.

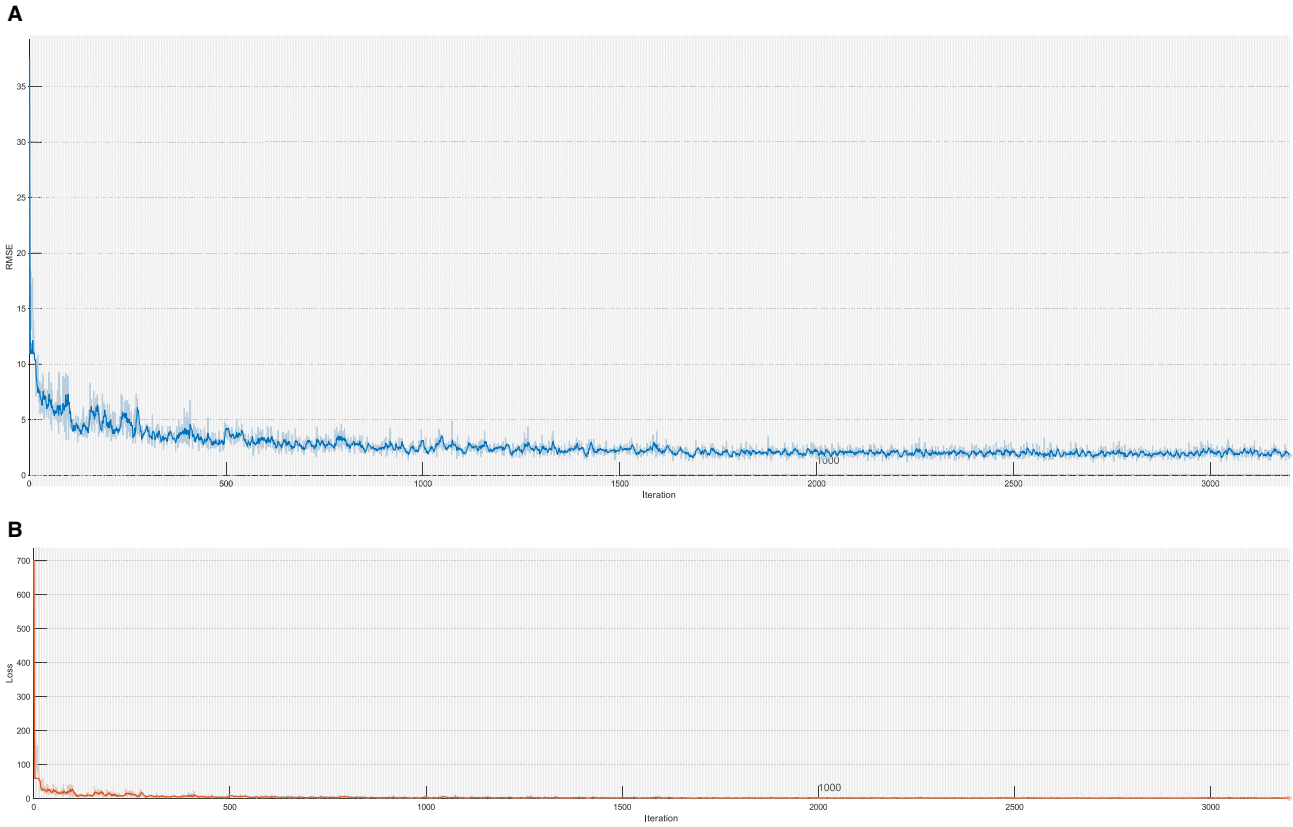


FIGURE 6
Training Convergence Process. **(A)** This is the RMSE of Training. **(B)** This is the loss of Training.

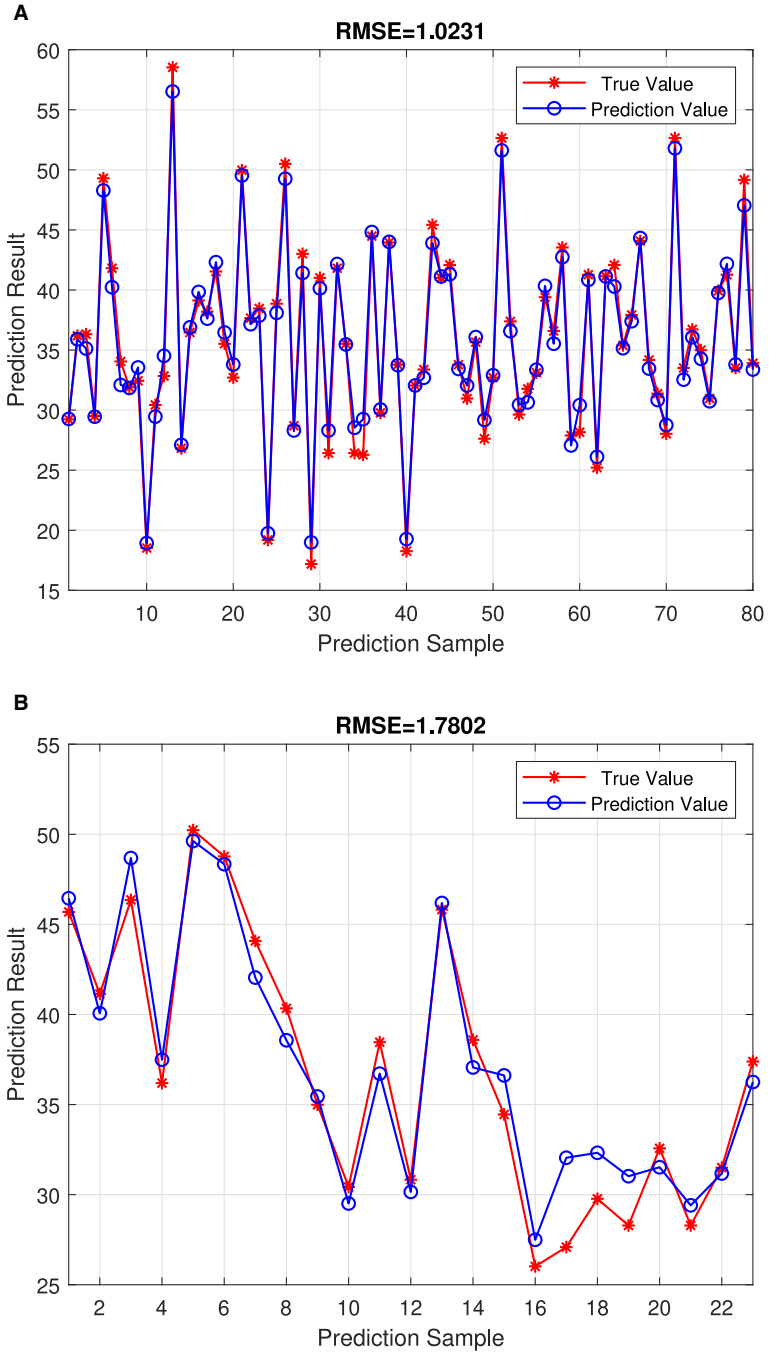


FIGURE 7
Algorithm efficiency with two-point configuration. **(A)** This is the RMSE = 1.0231 on the Training set. **(B)** This is the RMSE = 1.7802 on the Test set.

6.1.1 Polygon shape feature extraction module

The existing data sets include polygon pictures, polygon vertex coordinates, mean square error and control radius of Gaussian distribution, point configuration coordinate, and the fraction of coverage. We start by flattening the data into one dimension (it can also be flattened into 2-dimensional data, as well as 3-dimensional data, but it should always be consistent with the input layer data structure). The processed data are then fed into a convolutional layer with a convolutional kernel size of 3 × 1 channel of 16, outputting a 16-dimensional data. The BN layer and Relu activation function are then entered. Subsequent convolutional blocks operate similarly, but the dimensionality of the output is doubled. Determining the optimal number of convolutional layers and hyperparameters in our experiments is crucial for building efficient deep learning models, and given the scale of the parameters, we first determined a smaller number of network layers. We use incremental tuning of the number of network layers and hyperparameters,

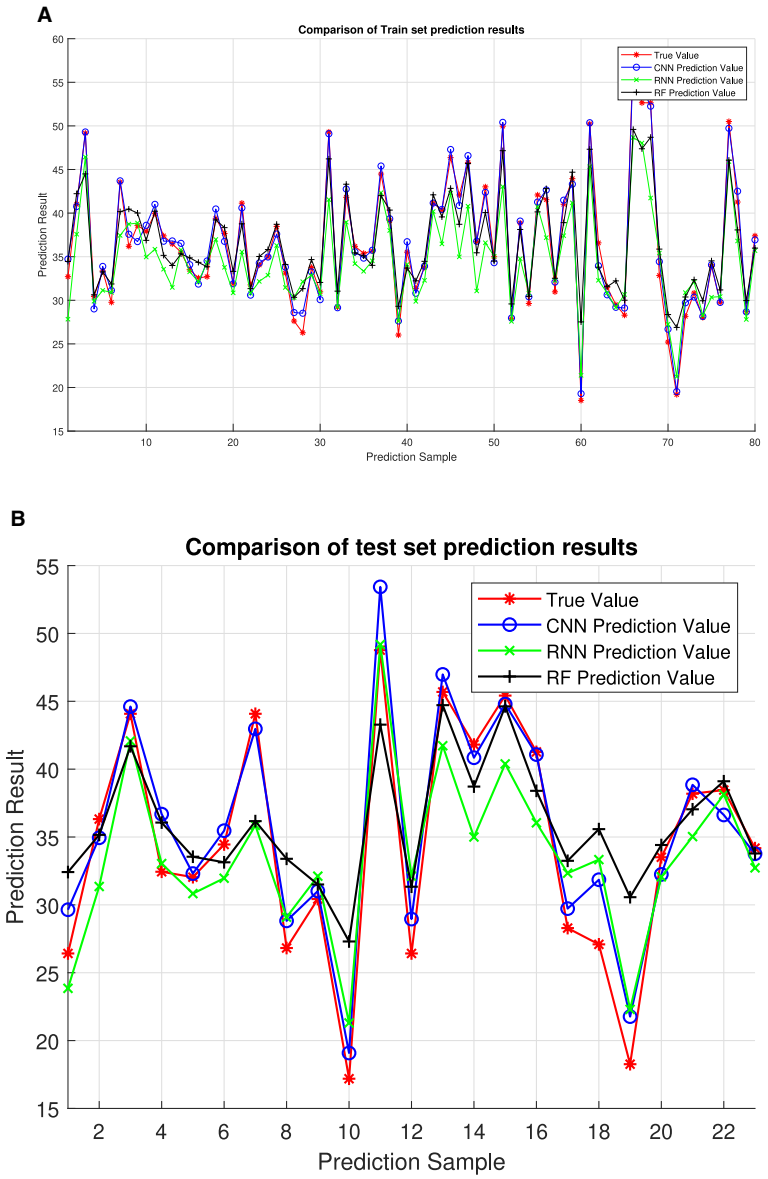


FIGURE 8
The comparison with CNN and RF. **(A)** Comparison of training set prediction result. **(B)** Comparison of test set prediction results.

starting with a smaller model and gradually increasing the number of layers and tuning the hyperparameters, evaluating the performance of the model after each increase until the performance no longer improves or begins to decline. Finally, considering the influence of polygon shape on the point configuration results, four convolution layers are used to extract the characteristics of polygon shape and output the corresponding point configuration results (Zhao et al., 2021). The network structure of the polygon shape feature extraction module is shown in Figure 4.

After data processing, the image data are input into the input layer in the form of a matrix. The image is originally composed of 24-bit RGB data, and all colored images can be represented by three channel colors: red, green, and blue. Therefore, we can also achieve image matrix transformation by using the RGB of each pixel in the image. The generated image in the sample is 875 × 656 pixels. To facilitate convolution processing, the size of all images in the data set is adjusted to 510 × 510, so the input images are converted into a matrix set of 510 × 510 × 3. Take 90% of the data as the training set and the remaining 10% as the test set.

To enable the image to recognize the shape of polygons after convolution, four convolution layers are set in the deep learning model, and the parameter settings of each layer are shown in Table 3.

ReLU is used as an activation function, and average pooling is selected in the pooling layer. The maximum number of training is 10,000, and the initial learning rate is 0.001. After four layers of convolution, $A^{1\times4}$ matrix is output after the fully connected layer, and then the point configuration result is obtained by regression analysis.

TABLE 4 Algorithm comparison results.

Number of targets	Target I	Target II	Target III	Target IV	Target V	Target VI
Control radius	20	20	20	20	20	20
Mean square error	5	5	5	5	5	5
Sample point	(53, −34)	(18, −46)	(4, −43)	(2, −4)	(−27, 0)	(−7, 17)
Configuration results	(8, −39)	(−17, −16)	(−16, −3)	(−13, −1)	(8, −30)	(53, −33)
Engineering algorithm expectation	0.309937	0.274576	0.480930	0.379013	0.514694	0.656193
Time consumed for engineering algorithm	484 s	499 s	472s	480 s	508 s	492 s
Point configuration	(50, −37)	(16, −40)	(0, −40)	(5, −39)	(−27, 5)	(−9, −18)
Results based on CNN	(5, −37)	(−15, −18)	(−14, −1)	(−11, −1)	(5, −27)	(50, −30)
Fraction of coverage	0.282662	0.256812	0.463702	0.349452	0.497282	0.656193
Time consumed for CNN	21 s	20 s	21 s	20 s	20 s	20 s
Fraction of coverage error	6%	7%	4%	8%	4%	0%

6.1.2 Regression fitting module

The parameters that affect the point configuration results include not only polygon shape but also the mean square error of Gaussian distribution and control radius of points; the mean square error of Gaussian distribution and control radius of points are not extracted in the polygon shape feature extraction module. Based on this, based on the polygon shape feature extraction module, a regression fitting module is constructed to extract the mean square error of the Gaussian distribution of points and the influence of the control radius of points on the point configuration results. The input of the regression fitting module is the point configuration coordinates output by the polygon shape feature extraction module and the mean square error of the Gaussian distribution of the corresponding points and the control radius of the points. The data input size is [3,1,1], which, respectively, represents the horizontal (vertical) coordinates of the upper-level point configuration, the mean square error of Gaussian distribution, and the control radius of the point.

The convolution kernel size is 3 × 1, and 16 feature maps are generated by the first layer of convolution. The difference is that after the convolution layer, the BN layer is selected instead of the pooling layer to standardize the data in the convolution network. The convolution kernel size of the second layer is 3 × 1, and 32 feature maps are generated, which are also subjected to the normalization layer and Relu activation function. To prevent over-fitting, the dropout layer is set to 0.2, making it 20% possible for the activation value of neurons to stop working and making the model more generalized. SGDM gradient descent algorithm (Cui et al., 2022) is used in the parameter setting; the maximum training times is 1,600, and the initial learning rate is 0.01. Overall, 90% of the data are still used as the training set, and the remaining 10% is used as the test set. The network structure of the regression fitting module is shown in Figure 5, and the convergence process is shown in Figure 6. The results show that after training and test sets, the regression fitting gradually converges, and the training effect is good.

6.2 Conclusion analysis and future work

This study presents a comprehensive analysis of the effectiveness and efficiency of the proposed algorithm for optimizing point configuration, which is exemplified through the optimal configuration of two points. Figure 7 illustrates the test results, showcasing the algorithm's efficacy in achieving desired configurations. Additionally, We adopted a two-point configuration and conducted a fitting regression comparison with other deep learning neural networks, Recurrent Neural Network (RNN) and Random Forest (RF). The experimental results, as shown in Figure 8, indicate that CNN's RMSE prediction performance is superior to that of RNN and RF. Table 4 presents a comparative analysis of six selected test samples, highlighting the superiority of the proposed approach. For the engineering algorithm model discussed in this study, configuring a single point with a stride of 5 m consumes ∼120 s. However, the time required increases to 480 s when optimizing the configuration of two points with the same stride. In contrast, the convolutional neural network (CNN) model completes calculations within the 20 s. These results unequivocally demonstrate the superior efficiency of the CNN algorithm compared with the traversal algorithm. Importantly, the efficiency advantage of the CNN algorithm remains prominent even as the number of points increases, as it does not escalate geometrically like the traversal algorithm. Harnessing the formidable learning ability and adaptability of deep learning networks, this study extracts key features from sample data generated by traversal algorithms, training the network to generate optimal point configurations. Simulation results underscore the advantages of the deep learning-based model over traversal engineering algorithms, particularly in terms of speed and real-time performance. Moreover, the calculation index error remains within 8%, indicating the model's high accuracy and reliability.

In summary, this research demonstrates the transformative potential of deep learning in optimizing point configuration for

robotic path planning and navigation. By significantly enhancing efficiency and accuracy while maintaining real-time performance, the proposed CNN-based approach offers a promising avenue for advancing autonomous systems in various domains. We believe that point configuration optimization algorithms can be applied in more fields. Future research directions are not limited to robot path planning, such as wireless sensor network (WSN) layout, graphics, and visual computing. In WSN layout, a point configuration optimization algorithm can be used to determine the optimal layout of sensors to maximize network coverage, extend network life, or improve data transmission efficiency. In graphics and visual computing, point configuration optimization algorithms can be applied to image reconstruction, three-dimensional modeling, animation production, and other fields to improve image quality or simulate physical phenomena by optimizing the position of points. In addition, some difficulties may be encountered in practical applications, such as the impact of complex environments, adaptability to dynamic environments, and limitations of computing resources. Therefore, we believe that the future direction of the point configuration optimization algorithm should be to introduce more advanced and efficient network models, reduce the number of parameters of the model, and improve the accuracy of the model.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

JW: Writing – original draft, Methodology, Conceptualization. HL: Writing – review & editing, Validation. BL: Writing – original draft, Supervision, Conceptualization. XZ: Writing – review & editing, Validation, Investigation. DZ: Writing – review & editing, Validation, Formal analysis.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Aote, S. S., Pimpalshende, A., Potnurwar, A., and Lohi, S. (2023). Binary particle swarm optimization with an improved genetic algorithm to solve multi-document text summarization problem of Hindi documents. *Eng. Appl. Artif. Intell.* 117:105575. doi: 10.1016/j.engappai.2022.105575

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., Wang, R., et al. (2019). "On exact computation with an infinitely wide neural net," in *Advances in Neural Information Processing Systems* 32.

Chao, Y., Augenstein, P., Roennau, A., Dillmann, R., and Xiong, Z. (2023). Brain inspired path planning algorithms for drones. *Front. Neurorobot.* 17:1111861. doi: 10.3389/fnbot.2023.1111861

Chappell, D., Crofts, J. J., Richter, M., and Tanner, G. (2021). A direction preserving discretization for computing phase-space densities. *SIAM J. Sci. Comput.* 43, B884–B906. doi: 10.1137/20M1352041

Chen, J., Du, C., Zhang, Y., Han, P., and Wei, W. (2021). A clustering-based coverage path planning method for autonomous heterogeneous UAVS. *IEEE Trans. Intell. Transp. Syst.* 23, 25546–25556. doi: 10.1109/TITS.2021.3066240

Cheng, D., Liao, R., Fidler, S., and Urtasun, R. (2019). "Darnet: deep active ray network for building segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7431–7439. doi: 10.1109/CVPR.2019.00761

Cui, Q., Liu, P., Du, H., Wang, H., and Ma, X. (2023). Improved multi-objective artificial bee colony algorithm-based path planning for mobile robots. *Front. Neurorobot.* 17:1196683. doi: 10.3389/fnbot.2023.1196683

Cui, Y., Xu, Y., Peng, R., and Wu, D. (2022). Layer normalization for tsk fuzzy system optimization in regression problems. *IEEE Trans. Fuzzy Syst.* 31, 254–264. doi: 10.1109/TFUZZ.2022.3185464

Das, P., and Jena, P. K. (2020). Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators. *Appl. Soft Comput.* 92, 106312. doi: 10.1016/j.asoc.2020.106312

De Goes, F., Butts, A., and Desbrun, M. (2020). Discrete differential operators on polygonal meshes. *ACM Trans. Graph.* 39, 110–111. doi: 10.1145/3386569.3392389

Elhassouny, A., and Smarandache, F. (2019). "Trends in deep convolutional neural networks architectures: a review," in *2019 International conference of computer science and renewable energies (ICCSRE)* (Agadir: IEEE), 1–8. doi: 10.1109/ICCSRE.2019.8807741

Heßler, K., and Irnich, S. (2022). A note on the linearity of Ratliff and Rosenthal's algorithm for optimal picker routing. *Oper. Res. Lett.* 50, 155–159. doi: 10.1016/j.orl.2022.01.014

Hu, A., Deng, Z., Yang, H., Zhang, Y., Gao, Y., Zhao, D., et al. (2021). An optimal geometry configuration algorithm of hybrid semi-passive location system based on mayfly optimization algorithm. *Sensors* 21:7484. doi: 10.3390/s21227484

Jie, Y., Jun, Z., Zhixiang, C., Xianyi, L., Zhiqing, Z., Zhi, L., et al. (2023). Optimization method of ballistic blast fragmentation warhead striking aircraft in aircraft shelter. *Xibei Gongye Daxue Xuebao* 41, 115–124. doi: 10.1051/jnwpu/20234110115

Jing, Y., Zhang, L., Hao, W., and Huang, L. (2022). Numerical study of a cnn-based model for regional wave prediction. *Ocean Eng.* 255:111400. doi: 10.1016/j.oceaneng.2022.111400

Kamra, N., and Liu, Y. (2021). "Gradient-based optimization for multi-resource spatial coverage problems," in *Uncertainty in Artificial Intelligence* (PMLR), 1885–1894.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Rob. Res.* 5, 90–98. doi: 10.1177/027836498600500106

Langazane, S. N., and Saha, A. K. (2022). Effects of particle swarm optimization and genetic algorithm control parameters on overcurrent relay selectivity and speed. *IEEE Access* 10, 4550–4567. doi: 10.1109/ACCESS.2022.3140679

Lei, K., Qi, D., and Tian, X. (2020). A new coordinate system for constructing spherical grid systems. *Appl. Sci.* 10:655. doi: 10.3390/app10020655

Lu, F., Jia, Z., Wang, H., and Wu, W. (2019). Aim point configuration method for area shoot of naval gun to surface target with arbitrary distribution. *Syst. Eng. Electron* 41, 1278–1285.

Luan, P. G., and Thinh, N. T. (2023). Hybrid genetic algorithm based smooth global-path planning for a mobile robot. *Mech. Based Des. Struct. Mach.* 51, 1758–1774. doi: 10.1080/15397734.2021.1876569

Luo, M., Hou, X., and Yang, J. (2020). Surface optimal path planning using an extended dijkstra algorithm. *IEEE Access* 8, 147827–147838. doi: 10.1109/ACCESS.2020.3015976

Mansor, M., and Morris, A. S. (1999). Path planning in unknown environment with obstacles using virtual window. *J. Intell. Robot. Syst.* 24, 235–251. doi: 10.1023/A:1008047425796

Nair, R. S., and Supriya, P. (2020). "Robotic path planning using recurrent neural networks," in *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (Kharagpur: IEEE), 1–5. doi: 10.1109/ICCCNT49239.2020.9225479

Sciavicco, L., and Siciliano, B. (1988). A solution algorithm to the inverse kinematic problem for redundant manipulators. *IEEE J. Robot. Autom.* 4, 403–410. doi: 10.1109/56.804

Shao, J. (2021). Robot path planning method based on genetic algorithm. *J. Phys.: Conf. Ser.* 1881:022046. doi: 10.1088/1742-6596/1881/2/022046

Simsekli, U., Sagun, L., and Gurbuzbalaban, M. (2019). "A tail-index analysis of stochastic gradient noise in deep neural networks," in *International Conference on Machine Learning* (PMLR), 5827–5837.

Siyu, G., Jun, G., Tianyi, Y., Bangyong, W., Xudong, Z., Yanzhao, X., et al. (2022). "Visual design of direct lightning risk assessment based on 3D model discretization algorithm," in *2022 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC)* (Beijing: IEEE), 776–778. doi: 10.1109/APEMC53576.2022.9888616

Venkateswara Reddy, L., Davanam, G., Pavan Kumar, T., Sunil Kumar, M., and Narendar, M. (2022). "Bio-inspired firefly algorithm for polygonal approximation on various shapes," in *Intelligent Computing and Applications: Proceedings of ICDIC 2020* (Cham: Springer), 95–107. doi: 10.1007/978-981-19-4162-7_10

Wang, L., Guo, J., Wang, Q., and Kan, J. (2018). "Ground robot path planning based on simulated annealing genetic algorithm," in *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)* (Zhengzhou: IEEE), 417–4177. doi: 10.1109/CyberC.2018.00081

Wu, L., Huang, X., Cui, J., Liu, C., and Xiao, W. (2023). Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. *Expert Syst. Appl.* 215:119410. doi: 10.1016/j.eswa.2022.119410

Wu, X., Wang, G., and Shen, N. (2023). Research on obstacle avoidance optimization and path planning of autonomous vehicles based on attention mechanism combined with multimodal information decision-making thoughts of robots. *Front. Neurorobot.* 17:1269447. doi: 10.3389/fnbot.2023.1269447

Yang, L., and Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: theory and practice. *Neurocomputing* 415, 295–316. doi: 10.1016/j.neucom.2020.07.061

Yu, J., Su, Y., and Liao, Y. (2020). The path planning of mobile robot by neural networks and hierarchical reinforcement learning. *Front. Neurorobot.* 14:63. doi: 10.3389/fnbot.2020.00063

Yu, Y., and Gupta, K. (2004). C-space entropy: a measure for view planning and exploration for general robot-sensor systems in unknown environments. *Int. J. Rob. Res.* 23, 1197–1223. doi: 10.1177/0278364904046631

Yu, Y., Si, X., Hu, C., and Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* 31, 1235–1270. doi: 10.1162/neco_a_01199

Zhang, L., Zhang, Y., and Li, Y. (2020). Mobile robot path planning based on improved localized particle swarm optimization. *IEEE Sens. J.* 21, 6962–6972. doi: 10.1109/JSEN.2020.3039275

Zhao, W., Persello, C., and Stein, A. (2021). Building outline delineation: from aerial images to polygons with an improved end-to-end learning framework. *ISPRS J. Photogramm. Remote Sens.* 175, 119–131. doi: 10.1016/j.isprsjprs.2021.02.014

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 11 July 2024
DOI 10.3389/fnbot.2024.1431643

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Yichuan Zhang,
Northeastern University, China
Ming Wan,
Liaoning University, China

*CORRESPONDENCE
Jun Wang
✉ wangjun@xj.cee-group.cn

RECEIVED 12 May 2024
ACCEPTED 31 May 2024
PUBLISHED 11 July 2024

CITATION
Zhang X, Wang J, Wang J, Wang H and Lu L (2024) Enhanced LSTM-based robotic agent for load forecasting in low-voltage distributed photovoltaic power distribution network. *Front. Neurorobot.* 18:1431643. doi: 10.3389/fnbot.2024.1431643

Enhanced LSTM-based robotic agent for load forecasting in low-voltage distributed photovoltaic power distribution network

Xudong Zhang[1], Junlong Wang[1], Jun Wang[2]*, Hao Wang[2] and Lijun Lu[2]

[1]State Grid Hebei Electric Power Company, Shijiazhuang, China, [2]Henan XJ Metering Co., Ltd, Xuchang, China

To ensure the safe operation and dispatching control of a low-voltage distributed photovoltaic (PV) power distribution network (PDN), the load forecasting problem of the PDN is studied in this study. Based on deep learning technology, this paper proposes a robot-assisted load forecasting method for low-voltage distributed photovoltaic power distribution networks using enhanced long short-term memory (LSTM). This method employs the frequency domain decomposition (FDD) to obtain boundary points and incorporates a dense layer following the LSTM layer to better extract data features. The LSTM is used to predict low-frequency and high-frequency components separately, enabling the model to precisely capture the voltage variation patterns across different frequency components, thereby achieving high-precision voltage prediction. By verifying the historical operation data set of a low-voltage distributed PV-PDN in Guangdong Province, experimental results demonstrate that the proposed "FDD+LSTM" model outperforms both recurrent neural network and support vector machine models in terms of prediction accuracy on both time scales of 1 h and 4 h. Precisely forecast the voltage in different seasons and time scales, which has a certain value in promoting the development of the PDN and related technology industry chain.

KEYWORDS

distributed photovoltaic, power distribution network, load forecasting, deep learning, long short-term memory

1 Introduction

Load forecasting of the power distribution network (PDN) is an important link in safe operation and dispatching control. With the popularization and application of energy storage technology and the addition of new dispatchable resources such as electric vehicles, a large number of interruptible and bidirectional loads appear on the load side (Dairi et al., 2020; Razavi et al., 2020; Markovics and Mayer, 2022). These load's randomness and distributed access characteristics affect the power system regulation of the PDN. Active distribution network (ADN) uses the core technology of demand response to dynamically adjust the price of electricity and incentive policies and flexibly manage and control the original load demand of users. Furthermore, it actively guides users to participate in the optimization of power dispatching to enhance the synergy and complementarity of

multiple loads. It not only considers users' satisfaction with electricity consumption but also improves the consumption ratio of distributed renewable energy (Hafiz et al., 2020; Mellit et al., 2021).

Proper planning and useful applications of load forecasting of the PDN require specific "predicting intervals". According to the delivery cycle, load forecasting can be divided into ultra-short-term, short-term, medium-term, and long-term (Eom et al., 2020). Ultra-short-term forecasting is employed for real-time control, enabling rapid adjustments to generation and load to ensure the safe and stable operation of the power grid. Short-term forecasting is widely employed in the daily operations of the utility industry, facilitating dispatch of generation and transmission, optimizing grid resource allocation and enhancing grid operational efficiency. Medium-term forecasting is primarily utilized to forecast load variations over the next few months to a year, providing valuable insights for fuel procurement, maintenance planning and grid investment decisions. Long-term forecasting focuses on load growth trends over the next 1 to 20 years, employed to forecast the need for new power plants, grid planning and providing strategic guidance for power system development.

Load forecasting of the PDN is complex for engineers and academics, and remains an ongoing area of research. Moreover, the thorough exploration of load-side controllable resources to achieve optimal dispatch of the power system by the grid has emerged as a critical research priority for contemporary power utilities. Nowadays, it is more and more common for low-voltage PDNs to adopt distributed photovoltaic (PV) access. On this basis, considering the regularity of PV power generation, the problem of voltage fluctuation can be solved by predicting the voltage variation trend.

Accurate load forecasting plays a crucial role in optimizing the scheduling and management of power resources, effectively reducing operational costs and enhancing the overall efficiency of the power system. With the rapid development of deep learning-based robotic agent technology (Ma et al., 2023, 2024a), the application of deep learning in load forecasting has gained significant attention, particularly for approaches based on recurrent neural networks (RNN). Furthermore, deep learning techniques can handle complex nonlinear relationships and massive datasets, thereby improving the accuracy and reliability of predictions, which are paramount for the stable operation of the power grid. Deep learning models, however, demand substantial data and computational resources, while their hyperparameter tuning and training process necessitate specialized knowledge and expertise. The nonlinearity and time dependence of load data increase the complexity of predictions. As the data dimensions increase, deep learning models need to possess enhanced learning capabilities, thereby avoiding overfitting and performance degradation. While significant progress has been made in load forecasting techniques, several challenges remain that require further attention to enhance the accuracy and efficiency.

To address the specific scenario of load forecasting in low-voltage distributed photovoltaic power distribution networks, we customized a load forecasting model and employed a long short-term memory (LSTM) network architecture for forecasting. To enhance feature extraction, we placed a fully connected layer, denoted as dense layer, after the LSTM layer. Additionally, we integrated the frequency domain decomposition (FDD) method to obtain the amplitude and phase of each frequency component, and utilized LSTM to individually forecast low-frequency and high-frequency components, ultimately improving the model's accuracy. This study is expected to offer a new idea for the low-voltage distributed PV-PDN to meet the forecast. The contributions of this paper can be summarized as follows:

1) FDD-enhanced LSTM for load forecasting in PV-PDN: to address the load forecasting of low-voltage distributed PV-PDN, we propose a novel FDD-enhanced LSTM model. The proposed model outperforms conventional support vector machine (SVM) and RNN models, particularly in long-term forecasting scenarios. This method represents a significant advancement in the application of deep learning techniques in the distribution network domain, providing a novel approach to enhance grid reliability and operational efficiency.
2) A new benchmark for load forecasting in PV systems: the integration of FDD and LSTM networks has revolutionized load forecasting in low-voltage distributed PV systems, establishing a new benchmark for forecasting methodologies in distributed PV systems.
3) Comparative analysis of FDD-enhanced LSTM for load forecasting: to objectively evaluate the performance of the enhanced LSTM model in complex low-voltage distributed PV forecasting scenarios, we conducted a comprehensive comparative analysis of the mean absolute error (MAE) across different time scales. The results demonstrate the model's superior performance and reliability in complex voltage forecasting environments.

The rest of the paper is organized as follows: Section 2 reviews the related work of load forecasting and scene image monitoring analysis. Section 3 describes the proposed methods in detail. Section 4 reports the experimental result and analysis. Section 5 represents the conclusion and future work.

2 Related work

Low-voltage load forecasting is an intelligent technique that utilizes historical load data, weather information and socioeconomic factors to forecast future low-voltage load levels. This technique possesses extensive application value in power grid scheduling, grid planning and electricity pricing.

Statistical and time series methods are widely employed techniques for short-term load forecasting, with linear models being the most prevalent approach. Linear models typically employ linear parameters. Litjens et al. (2018) have utilized some of the simplest linear models, including seasonal persistence models and simple average models, often in conjunction with meteorological data. Borges et al. employed linear models with varying feature subsets for short-term load forecasting and missing data imputation in substation data (Borges et al., 2020).Their model utilized historical load data, meteorological data and neighboring substation data. While standard linear regression has

proven successful in demand forecasting for all levels of low-voltage networks, nonlinear regression models have also gained attention due to their inherent flexibility. Hayes et al. employed a nonlinear autoregressive exogenous (NARX) model for smart meter load forecasting and demonstrated its superior performance compared to traditional NARX models and neural network models (Hayes et al., 2014). Tsekouras et al. (2007) employed nonlinear multiple regression, selecting a model based on testing various combinations of nonlinear functions for mid-term load forecasting. Nonlinear models, despite their wide applicability, are susceptible to overfitting issues.

Among time series forecasting models, ARIMA stands out due to its exceptional performance and has been widely adopted across various applications (Marinescu et al., 2013). Researchers have successfully integrated online ARIMA models into short-term forecasting of electricity systems in public school buildings (Lee et al., 2013). Leveraging historical load and temperature data, this model effectively captures energy efficiency, forecasts energy consumption and detects anomalies in energy usage. Furthermore, Espinoza et al. proposed a unified modeling framework based on periodic autoregressive models, enabling the effective integration of data from multiple entities to achieve load curve forecasting and clustering analysis (Espinoza et al., 2005).

With the continuous advancement of deep learning (Ma et al., 2021, 2024b; Li et al., 2023; Jin et al., 2024; Liufu et al., 2024), deep learning-based load forecasting has also gained widespread attention from researchers. Deep learning-based load forecasting methods, with their ability to capture complex data patterns and extract deep-level features, have gradually become a research hotspot in the field of power load forecasting and have achieved remarkable results. Shivam et al. (2021) discuss a predictive energy management strategy for residential PV-battery systems using RNN model, it has a deep inner hidden layer, which imitates the neural network inside humans to think like the human brain. Luo et al. (2021) enhance photovoltaic power generation forecasting by incorporating domain knowledge into deep learning models (Kim et al., 2020). The limitation of machine learning (ML) lies in the need for more learning ability for high-dimensional data. The purpose of representative learning is to simplify complex original data, remove invalid or redundant information from original data, and refine effective information to form features. The purpose of representative learning is to simplify complex raw data, remove redundant or invalid information from the data, and extract effective information to form features. In addition, SVM and LSTM have been widely used in load forecasting. Kabilan et al. (2021) and Feng et al. (2020) both employ machine learning models for short-term power prediction and quantifying daily global solar radiation, respectively, highlighting the potential of computational methods in optimizing and accurately forecasting solar energy production. Kim et al. (2020) focus on very-short-term photovoltaic forecasting for smart city energy management through multiscale LSTM-based deep learning.

In the realm of load forecasting, traditional methods have often faced limitations in capturing the intricate patterns and underlying relationships within complex electricity consumption data. To address these shortcomings, we propose a novel deep learning-based load forecasting framework that leverages the powerful capabilities of RNN and LSTM cells to effectively capture temporal dependencies.

3 Methods

3.1 Features of distributed PV-PDN

PV power generation is essentially a power technology that uses the photoelectric or photochemical effect of PV modules (semiconductor materials) to convert light energy directly into electric energy. Distributed PV power station usually refers to a power generation system with a small installed scale that uses distributed resources and is located near the user. Ordinarily, the power grid with a voltage level of <35 kV or lower is connected. The heart of a PV facility is solar panels. The semiconductor materials adopted for power generation principally cover polysilicon, monocrystalline silicon, amorphous silicon and cadmium telluride (Lopes et al., 2022). Solar panels are the core and most valuable part of a solar power system. Its role is to convert the radiant power of the sun into electrical energy, feed it into a storage battery or promote load operation. The function of the solar controller is to control the working state of the entire system and protect the battery from overcharge and discharge (Alipour et al., 2020; Korkmaz, 2021; Qadir et al., 2021). Qualified controllers should also have a temperature compensation function in places with large temperature differences.

The PV cell's equivalent circuit (EC) is shown in Figure 1. I_{ph} and I_d refer to the photo-generated and diode junction currents; C_j means the junction capacitance (negligible); R_s and R_{sh} stand for series and parallel resistors. Typically, distributed PV projects have a capacity of within a few kilowatts. Unlike centralized plants, the scale of PV plants has little effect on power generation efficiency. Therefore, its influence on the economy is also tiny, and the return on investment of small PV systems will not be lower than that of large ones.

Solar energy's direct output is generally 48 VDC, 24 VDC and 12 VDC. To power an appliance at 220 VAC, direct current (DC) generated by a solar power system needs to be converted into alternating current (AC). To avoid power backflow, it is necessary to configure an anti-flow device for alarm, and the inverter then adjusts its capacity according to the received signal. To connect the distributed PV system to the PDN, it first needs to output the PV cells through the DC/DC converter, then connected to the DC/AC inverter, and next connected to the external PDN. Taking a household small distributed PV system as an example, the typical grid-connected PV structure is displayed in Figure 2. The grid-connected access information acquisition system of small distributed PV power stations is applied to transmit the collected information to the monitoring platform and display it to users or power grid enterprises intuitively and clearly. This can provide grid enterprises with grid-connected data of PV power stations, eliminate the "blind adjustment" phenomenon of PV power generation, assist power grid operation analysis and decision-making, and promote the operation of the power grid safe and stable (Karimi et al., 2020; Ding et al., 2021; Khan et al., 2022).

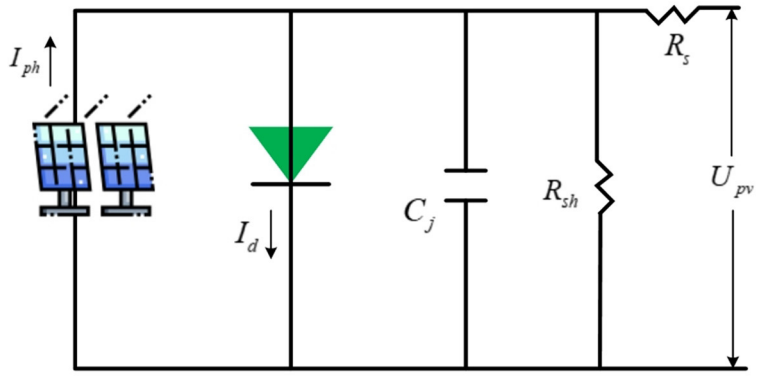


FIGURE 1
EC of PV cell.

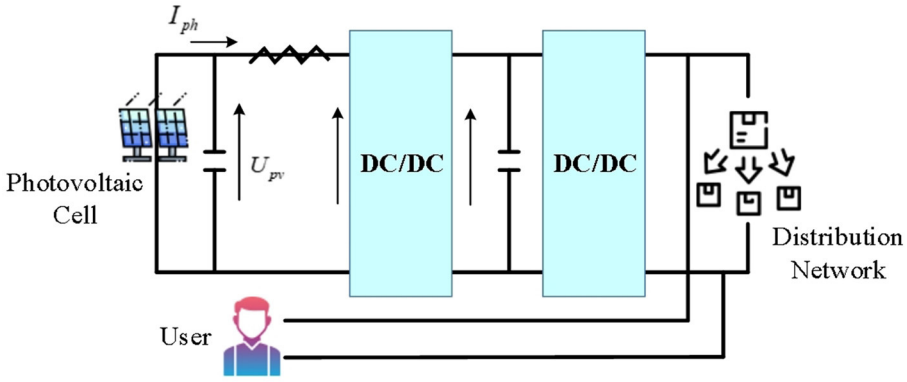


FIGURE 2
Grid-connected structure of household small distributed PV system.

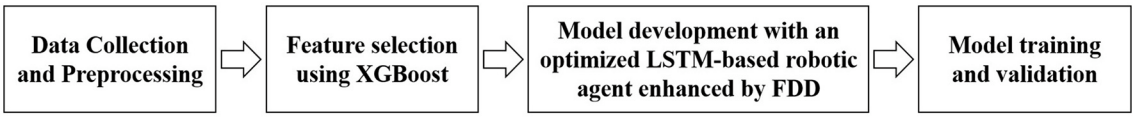


FIGURE 3
The overall diagram of the method in this study.

The overall diagram of the method in this study is shown as Figure 3. The method involves meticulous data collection and preprocessing to ensure high-quality inputs, followed by strategic feature selection via the XGBoost algorithm to optimize data relevancy. Then an advanced LSTM model is designed and refined, augmented with FDD, for enhanced predictive accuracy.

3.2 Voltage data preprocessing and feature selection

As a kind of clean energy, the high proportion of PV connected to a low-voltage PDN will bring huge power generation benefits. However, due to its own uncertainty, it may bring a series of problems to the stable and safe operation of the PDN, such as voltage over the limit, line overload and power quality reduction. Thus, it is essential to accurately evaluate the acceptance capacity of PV in a low-voltage PDN. More importantly, to further improve the benefits of PV power generation, it is urgent to improve the acceptance capacity of distributed PV based on accurate assessment (Rana and Rahman, 2020). Before voltage prediction of distributed PV-PDN, data mining and preprocessing should be carried out, ensuring that it is in a suitable form for analysis. This step involves removing outliers, handling missing values, and normalizing data, which helps reduce variability and improve the model's accuracy. It also includes feature selection and transformation to identify and utilize the most relevant information for forecasting, thereby enhancing the prediction model's effectiveness and efficiency.

The missing data is filled using the cubic spline interpolation fitting function $f_\theta(x)$, and the equation for filling the value is Equation (1):

$$D(t_{miss}) = f_\theta(t_{miss}) \tag{1}$$

t_{miss} indicates the time point at which load data is missing.

For data satisfying normal distribution, standardized methods are used for dimensionless processing, with the specific equation as follows Equation (2):

$$x^* = \frac{x - \overline{X}}{S} \tag{2}$$

x and x^* refer to the original and the processed feature data, respectively; $\overline{X}$ and S represent the mean and standard deviation of the feature, respectively.

DL model has advantages in capturing power voltage fluctuation in distributed PV voltage prediction due to their ability to model complex, nonlinear relationships within large datasets. They excel in identifying patterns and dependencies in temporal data, such as those found in voltage series, by leveraging multiple layers of processing. This capability allows DL models to provide more accurate and reliable forecasts of voltage fluctuations, which is essential for maintaining grid stability and optimizing energy distribution in distributed PV-PDN. At this time, dimensionless standardization of different power characteristics can significantly accelerate the optimization speed of the gradient descent algorithm. The maximum and minimum rescaling method of voltage and power is illustrated in Equations (3) and (4):

$$v^* = \frac{v - v_{\max}}{v_{\max} - v_{\min}} \tag{3}$$

$$p^* = \frac{p - p_{\max}}{p_{\max} - p_{\min}} \tag{4}$$

v & p and v^* & p^* represent time-series raw data and dimensionless data for voltage and power; $v_{\max}$ & $v_{\min}$ and $p_{\max}$ & $p_{\min}$ refer to the voltage data's and power data's maximum and minimum values, respectively.

Generally speaking, the power load is filled with data of similar size. Because the power load has a certain periodicity, it can be filled and replaced with similar load data of the same cycle. The power load has a regular periodicity, that is, the data of different periods at the same time should be very different. If the difference between the two data exceeds the threshold, the vertical method can be used for processing. For the PV system, the light intensity in winter is lower than that in summer, and the maximum light intensity is usually at noon, so the voltage will rise in this period. It can be seen that the time feature vector is very vital in the voltage prediction process, and it is a key factor in improving the prediction accuracy.

Considering various types of features in the voltage prediction process, this study will adopt the feature selection method based on the Extreme Gradient Boosting (XGBoost) algorithm (Bae et al., 2021), a method chosen for its efficiency and effectiveness in handling high-dimensional data. XGBoost is renowned for its ability to improve model performance by selecting the most relevant features, reducing noise and preventing overfitting. This approach aids in identifying the key predictors of voltage fluctuations in distributed PV systems, thus enhancing the predictive accuracy of the deep learning model. In the course of multiple iterations, the probability distribution (PD) of the training data used in the current iteration will be regulated based on the results of the previous iteration. That is to say, each sample of training data has a weight, which itself will be adjusted with iteration. As suggested in Figure 4, Dm is the training dataset's PD. In the first iteration, the classification error of basic classifier C1 is employed to adjust D2; In the second iteration, the base classifier C2 is used for iteration D3, and so on.

XGBoost is the use of multiple base learning. Each base learning is relatively simple. To prevent overfitting, the next learning is the result of learning the previous base learning. The loss function of XGBoost algorithm reads Equations (5) and (6):

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{m=1}^{M} \Omega(b_m) \tag{5}$$

$$\Omega(b_m) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \tag{6}$$

n refers to the number of samples; y_i and $\hat{y}_i$ represent the label value of the i-th sample and output value predicted by the model, respectively; l means the squared error function; $\Omega(b_m)$ expressed a regularized term for the tree model. T displays the leaf nodes' quantities for a single tree model; w signifies the output vector of the leaf node; γ are λ parameters that control the weights of regularized terms.

After the model is initialized, it needs to carry out M-round cycle calculation, so the objective function $Obj^{(t)}$ should be minimized during the t-round calculation Equations (7) and (8):

$$Obj^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + b_t(x_i)) + \Omega(b_t) + C \tag{7}$$

$$C = \sum_{i=1}^{t-1} (b_i) \tag{8}$$

b_t represents tree model in the t-round training; $\hat{y}_i^{(t-1)}$ denotes the predicted output value of the model obtained from the previous round; $\Omega(b_t)$ indicates the complexity of the tree model obtained in t-round; C is a constant.

When solving the objective function of a binary tree, it is necessary to know the first-order and second-order derivatives of the loss function, and on which leaf node the sample is located. It is also necessary to find the first and second derivatives of the sample at each leaf node to find the objective function. In this way, it is possible to decide whether to split the node and according to the characteristic values of which node to split.

The voltage, power of key nodes and time characteristics of prediction points in PDN are selected and taken as input feature vector x after series. When forecasting, the higher the prediction accuracy of 1 h ago, the higher the multiple time scales' prediction accuracy. The dimensionless node voltage and net power data of the complete PDN are obtained through data preprocessing. The

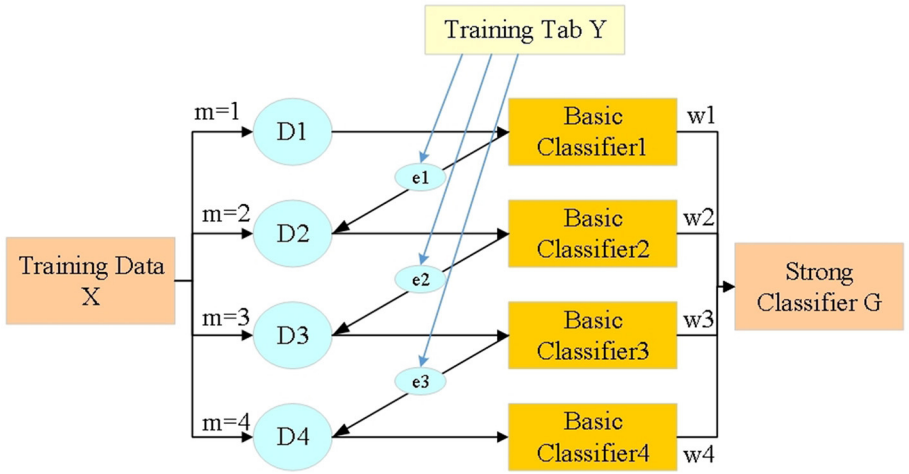


FIGURE 4
Training flow of XGBoost algorithm.

voltage eigenvector V_i of the node, the net power eigenvector P_i and the corresponding label y_i are obtained as follows Equations (9–11):

$$\boldsymbol{V_i} = \left[\boldsymbol{v_{i+t-H}},\cdots,\boldsymbol{v_{i+t-2}},\boldsymbol{v_{i+t-1}}\right] \tag{9}$$

$$\boldsymbol{V_i} = \left[\boldsymbol{p_{i+t-H}},\cdots,\boldsymbol{p_{i+t-2}},\boldsymbol{p_{i+t-1}}\right] \tag{10}$$

$$\boldsymbol{y_i} = \boldsymbol{v_{i+t}} \tag{11}$$

H represents the length of the sliding window, V_i and V_i are eigenvectors with dimension H.

The time variable of discretization is processed by unique thermal coding. Time eigenvector T_i is constructed to predict time points. Finally, the input feature vector x_i of the i-th sample can be expressed as Equation (12):

$$\boldsymbol{x_i} = [\boldsymbol{V_i},\boldsymbol{P_i},\boldsymbol{T_i}] \tag{12}$$

x_i and y_i together constitute the training sample set of the XGBoost algorithm, which can be written as Equation (13):

$$\left\{\left(\boldsymbol{x_i},\boldsymbol{y_i}\right)\right\}_{i=1}^{n} \tag{13}$$

Additionally, XGBoost suggests two ways to avoid overfitting. The first is Shrinkage, namely, the learning rate. In each tree iteration, each leaf node's weight is multiplied by a reduction coefficient. This way, the impact of each tree will not be too large, leaving more space for optimization for the trees below (Wang et al., 2017; Liu et al., 2022). Another way is Column Subsampling, which is similar to random forest selection for tree construction. There are two methods: (1) Random sampling by layer. Before splitting nodes of the same layer, some eigenvalues are randomly selected for traversal to calculate information gain (IG); (2) Some eigenvalues are randomly sampled before building a tree. Then the tree's all-node splits traverse these eigenvalues to compute IG.

The Mean Absolute Error (MAE) to validate the performance of prediction methods, which is an objective function used to measure the average absolute difference between predicted and true values in regression problems. It can measure the average error size between predicted values and true values, and has good robustness. The calculation formula for MAE is written as Equation (14):

$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}\left|\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}\right| \tag{14}$$

N represents the number of samples, y_i is the true value, and $\hat{y}_i$ is the predicted value.

The smaller the value of MAE, the smaller the average difference between the predicted value and the true value, indicating higher accuracy of the prediction.

3.3 Load forecasting of distributed PV system based on FDD + LSTM

In the context of distributed photovoltaic systems, load forecasting necessitates a multifaceted analytical approach. Key is the scrutiny of historical data to discern patterns and trends. Employing statistical methods, such as time series analysis, facilitates the understanding of complex data interrelations. Moreover, the application of machine learning algorithms, including neural networks, is essential for improving prediction accuracy given the nonlinear nature of load data. Selecting pertinent features, particularly those influenced by weather and temporal factors, is critical. Additionally, integrating renewable energy sources, notably solar power, introduces unpredictability, demanding innovative, adaptable forecasting techniques to ensure consistent power distribution.

FDD refers to taking the Fourier transform (FT) of the signal to analyze it. FT is a mathematical equation that relates a signal sampled in space or time to the same signal sampled at frequency (Polo et al., 2023). In signal processing, FT can reveal a signal's vital characteristics (i.e., its frequency component). For a vector x containing n uniform sampling points, FT is defined as

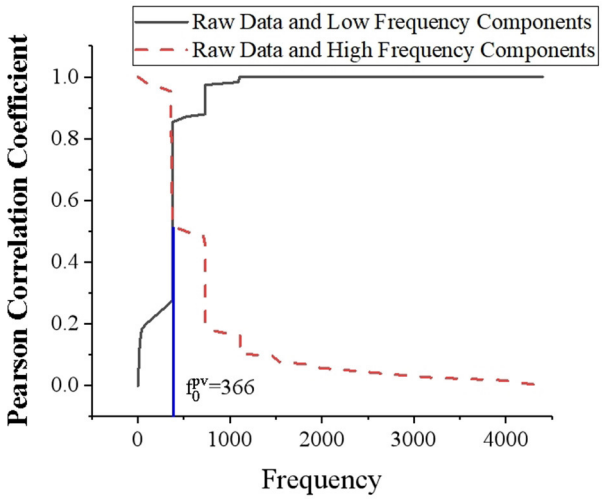


FIGURE 5
Correlation between high frequency and low-frequency components and PV power.

TABLE 1 Predicted results of low-frequency and high frequency components at different frequencies.

Component	R					
	366	732	1,098	1,463	1,830	2,196
Low-frequency	0.985	0.986	0.987	0.988	0.989	0.990
High frequency	0.960	0.678	0.351	0.101	−2.175	−3.550

Equation (15):

$$y_{k+1} = \sum_{j=0}^{n-1} \omega^{jk} x_{j+1} \tag{15}$$

ω is one of the n complex roots of unity; For x and y, indexes j and k range from 0 to $n-1$.

The Fourier analysis method is extended to aperiodic signals, and FT is introduced. When the period of a periodic signal increases infinitely, the frequency spectrum tends to become infinitely small and cannot be represented by the Fourier series. But from a physical point of view, the spectrum is still there. FT spectrum analysis divides PV power into load forecasting and high frequency components (Liu et al., 2020; Zang et al., 2020; Rai et al., 2021). The low-frequency component represents the conventional part of PV performance, which can be accurately predicted and indicates the trend characteristics. The high-frequency component exhibits the randomness of PV power and the fluctuation characteristics affected by weather and other factors, which is relatively difficult to predict. Figure 5 presents the correlation between low-frequency and high frequency components and PV power. When FDD is performed on PV power data, the more frequency is selected, the weaker the correlation between high frequency component and PV power is. However, the correlation between low-frequency component and photovoltaic power is stronger. Table 1 compares the predicted results of the two frequency components at different frequencies. The selection of frequency boundary points is based on frequency nodes with larger amplitude in the amplitude spectrum. It can be found that the core of frequency demarcation point selection is that the frequency selected by the low-frequency component should be as high as possible. Thereby, the low-frequency component accounts for more, and it is necessary to ensure that the frequency of the high frequency component is not too high, thus avoiding excessive difficulty in prediction.

Convolutional networks can process images of different lengths and widths, and Recurrent Neural networks (RNN) have a recurrent function that can process data of different lengths and sequence types. However, due to the small range that RNN can utilize, it cannot handle the long sequence data well. The output that leads directly to a long sequence forgets the input that is farther away. LSTM is a special kind of RNN, a modified version of RNN, whose structure is plotted in Figure 6. The activation function is the sigmoid; tanh is the hyperbolic tangent function; $\oplus$ and $\otimes$ represent the addition and multiplication operations of vectors. The first layer of LSTM comprises a single-loop structure, which is determined by the dimensions and number of input data and loops, rather than the connection of multiple single-loop structures. LSTM cells contain input, forget, output and unit states (Akram et al., 2020; Zhang et al., 2020; Ahmad et al., 2022). The input gate determines how much network input data requires to be saved to the unit state at the current moment. The forget gate decides how many unit states need to be transferred from the last to the present moment. The output gate controls how much of the current unit state demands to be output to the present output value.

In the discussed PV-PDN voltage prediction model based on "FDD+LSTM", to better extract data characteristics, a fully connected layer, namely Dense layer, is placed behind the LSTM layer. The specific voltage prediction process is as follows. (1) The prediction methodology employs XGBoost for feature subset selection, focusing on crucial elements like voltage, power characteristics and temporal variables. This step is pivotal in distilling the most relevant features from a vast dataset, thereby improving model efficiency and focus. The resulting feature vector $x = [V, P, T]$ is a comprehensive aggregation of these elements, forming the LSTM input alongside the target training variable y_i. (2) The backpropagation algorithm is utilized for model training, optimizing the network to reduce prediction errors and heighten voltage forecasting accuracy. This phase ensures in-depth learning from historical data, a critical aspect of the model's predictive capability. (3) Finally, the trained LSTM model, equipped with learned patterns, processes the input dataset for voltage prediction. The inclusion of the dense layer at this point is significant. It acts as a refinement stage, aligning LSTM outputs with expected voltage levels and synthesizing complex relationships. This addition enhances the model's accuracy and robustness in diverse operational scenarios within PV-PDNs.

Detailed procedure for load forecasting of distributed PV system based on FDD+LSTM:

(1) Data selection and preprocessing: historical operation data is carefully selected and subjected to data mining and preprocessing techniques. This includes handling outliers,

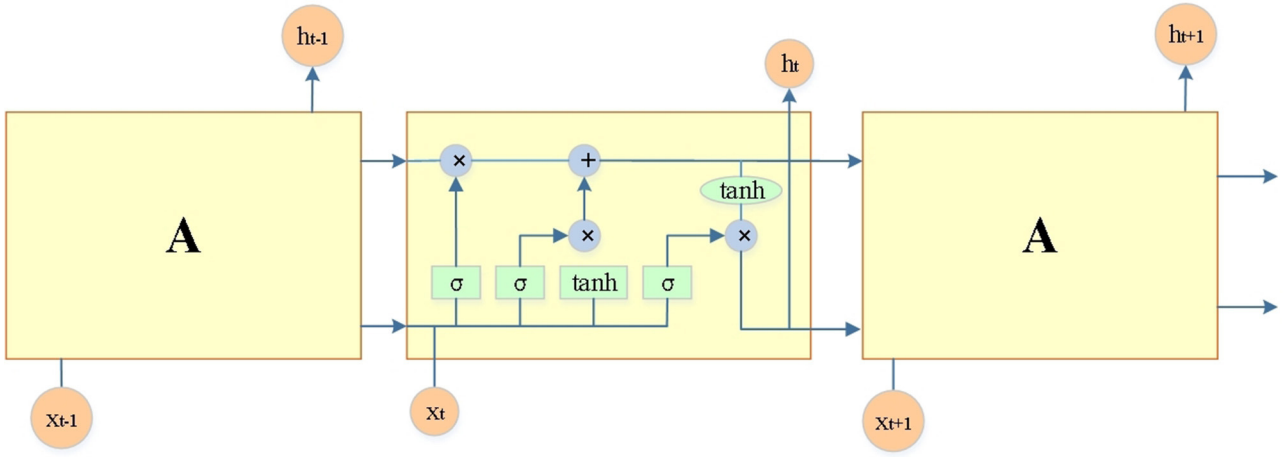


FIGURE 6
Basic structural unit of LSTM.

TABLE 2 Specific parameters of LSTM prediction model.

Model Layer	Hyper-parameter	Output tensor dimension
Input layer	–	(None, input)
LSTM layer	Neurons in memory cells: 64	(None, 64)
Dense layer	Neurons: 128 Activation function: tanh	(None, 128)
Dropout layer	Drop rate: 0.05	(None, 128)
Output layer	Neurons: 1 Activation function: sigmoid	(None, 1)

addressing missing values and normalizing the data to ensure its suitability for analysis.

(2) Feature selection: to identify the most influential variables contributing to the prediction task, we employ the XGBoost algorithm for feature selection. This approach enables us to pinpoint key predictive factors such as voltage, power characteristics and time variables that significantly impact the target variable.

(3) Model training: to achieve accurate and reliable voltage predictions, we employ the proposed "FDD+LSTM" neural network architecture and train it using the backpropagation algorithm.

(4) Load prediction: to harness the predictive ability of the trained proposed "FDD+LSTM" model, we utilize it to process the input dataset for accurate voltage forecasting. To further enhance the model's ability to extract meaningful features from the data and improve prediction accuracy, we incorporate a dense layer into the network architecture.

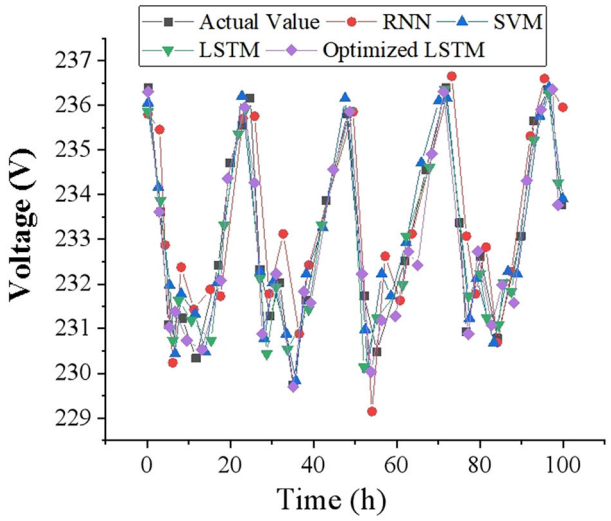


FIGURE 7
Voltage prediction results with a time scale of 1 h.

4 Results and discussion

4.1 Data selection and example analysis

This study selects the historical operation data of a low-voltage distributed PV-PDN in Guangdong Province as the research object. The time range is operation data from March 2020 to March 2022. The data sampling interval of the meter under test is 1 h, and rolling prediction is adopted. The constructed input feature vectors x_i are: the vectors of voltage, power and time characteristics are 12, 12 and 35 dimensions, respectively, and a total of 16,275 data samples are constructed with x_i and label y_i. An example analysis of the load forecasting model uses TensorFlow 14.0. The dropout layer is incorporated to prevent overfitting, followed by the connection to the output layer. The specific parameters of the "FDD+LSTM" prediction model are outlined in Table 2. The selected comparison algorithms are RNN, SVM, and LSTM to verify the validity of the prediction method proposed here.

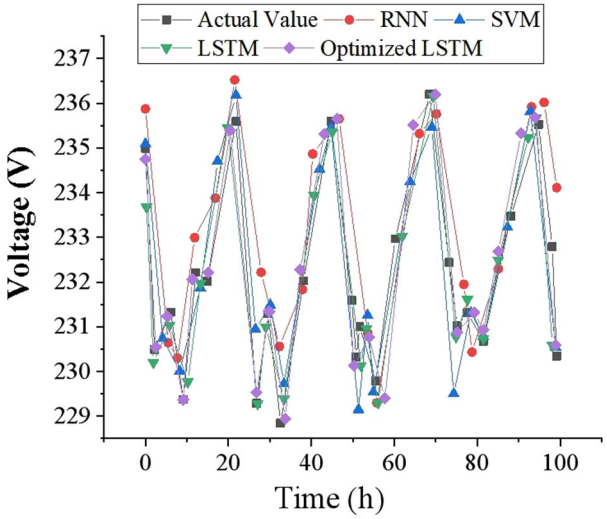


FIGURE 8
Voltage prediction outcomes with a time scale of 2 h.

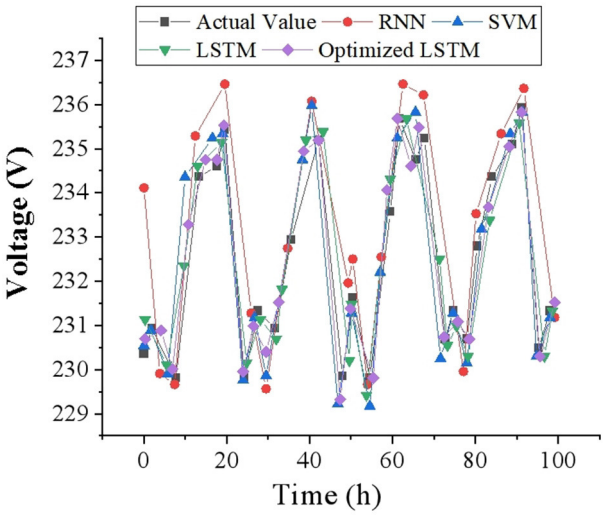


FIGURE 9
Voltage prediction results with a time scale of 4 h.

4.2 Analysis of load forecasting results in distributed PV-PDN

To intuitively reflect the accuracy of voltage prediction results, this study draws corresponding voltage prediction curves with 1, 2 and 4-h as time scales, and compares them with other prediction models. The voltage data of 100-time points is selected as a display in the test set, and the voltage prediction results at different time scales are demonstrated in Figures 7–9. It can be found that the FDD-enhanced LSTM model consistently aligns more closely with actual voltage values than SVM (Kabilan et al., 2021), RNN (Shivam et al., 2021) and LSTM (Feng et al., 2020) models, especially as the prediction time scale increases. Quantitatively, the LSTM model's MAE is significantly lower, at 0.4554 for a 1-h scale, compared to 0.535 and 1.012 for RNN and SVM, respectively. Even at a 4-h scale, the LSTM's MAE remains the lowest at 1.085. The superior forecasting precision of the optimized LSTM model can be attributed to its ability to effectively capture and learn from the temporal dependencies inherent in voltage data over time. Unlike SVM and RNN models, LSTM's architecture allows it to remember information for longer periods, making it particularly adept at handling the sequence prediction problems characteristic of voltage forecasting in distributed PV-PDNs. This is crucial for accurately predicting voltage fluctuations over different time scales, as it can account for both short-term and long-term patterns in the data. Additionally, the integration of FDD likely enhances the model's capability to deal with the non-linear and complex nature of the voltage signals, further improving prediction accuracy.

4.3 Performance evaluation of load forecasting model under different seasons

Taking the time scale of 1-h and 4-h as the basis, this study further verifies the voltage prediction of different PDN's load forecasting models in the four seasons, and the comparison results are portrayed in Figures 10, 11. It can be concluded that the prediction results of the improved LSTM model based on FDD are optimal in all seasons, especially as the prediction time scale increases. Taking summer with a time scale of 1 h as an example, the prediction MAE of the improved LSTM model is only 0.24, which reduces the prediction error of this model by about 35%. Even at a 4-h scale, the LSTM's MAE remains the lowest at 1.064 in summer. Therefore, the capability of the model in load forecasting of PV-PDN is further verified.

The proposed algorithm demonstrates significant practical value and effectiveness in the PV-PDN scenario. It can accurately predict voltage variations under different environmental conditions, and its prediction accuracy surpasses that of other models, especially as the prediction time scale increases. This capability provides strong support for the safe, reliable and efficient operation of PV power stations, helping maintenance personnel to promptly identify and resolve potential issues, thereby improving the operational efficiency and long-term stability of the PV power stations.

5 Conclusion

Driven by the rapid development of new power systems, the proportion of new energy is continuously increasing, and the scale of application and access rate of distributed PV in the low-voltage PDN are also steadily rising. The integration of distributed PV power generation, nonetheless, often exerts a substantial impact on the voltage distribution within PDN, giving rise to issues such as low voltage and voltage fluctuations. These issues severely impact the quality of daily life and production for users, further augmenting the uncertainty in grid operation and hindering the development of the social economy. Consequently, enhancing the state awareness capability of PDN is of paramount importance. Effective voltage prediction can provide data support for the safe

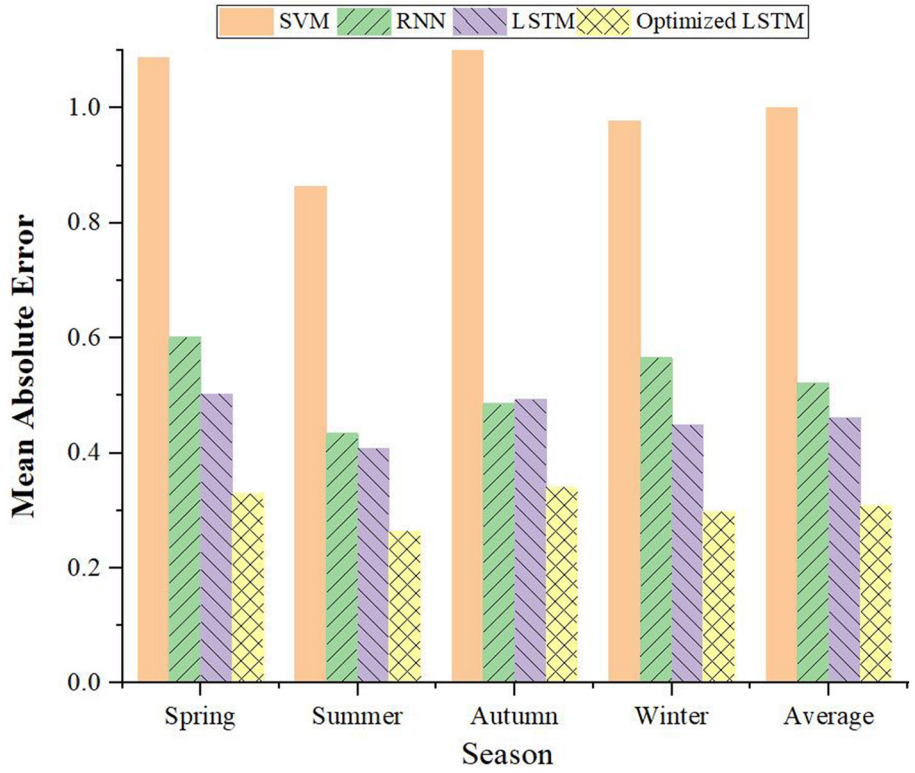


FIGURE 10
Comparison of load forecasting models for PDN in different seasons with a time scale of 1 h.

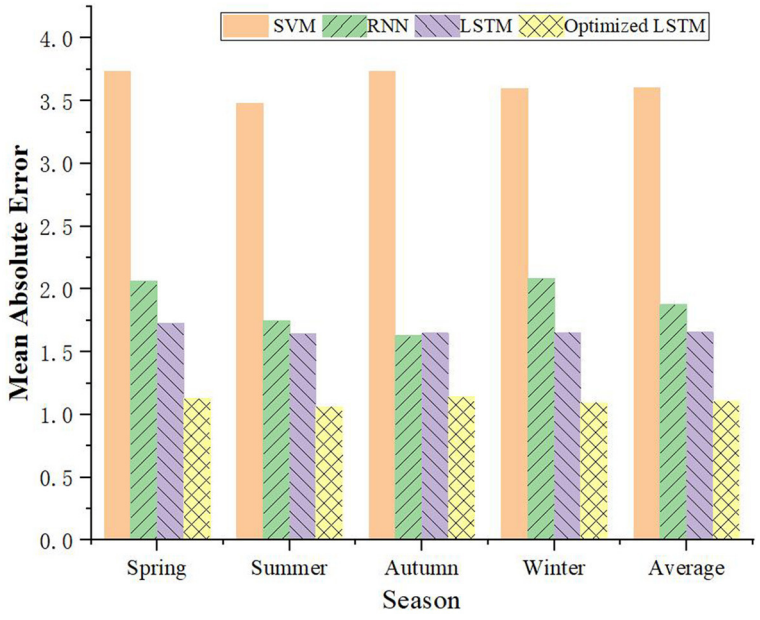


FIGURE 11
Comparison of load forecasting models for PDN in different seasons with a time scale of 4 h.

and stable operation of PDN, thereby facilitating the resolution of voltage issues arising from the integration of distributed PV systems. In recent years, LSTM networks have demonstrated remarkable application potential in the realm of power load forecasting, and it offer a novel solution for PDN voltage prediction. Thus, a LSTM is extensively used in power load forecasting model of actual PDN based on DL and FDD is proposed in this study. By fast Fourier decomposition of the original quantity, the phase

and amplitude of each frequency sine wave are acquired. Then LSTM is used to predict the low-frequency and high frequency components, respectively. The effectiveness of the proposed FDD-based LSTM model is verified by testing the historical operating data of PV-PDN. With the increase of the prediction time scale of the improved model, the error of the predicted results does not increase significantly. At a 1-h time scale, the MAE of the improved LSTM model is only 0.4554, much lower than that of other models. However, the proposed model requires a large amount of data for training and cannot be directly deployed on edge clients with limited computational resources for prediction. In the future, with the continuous development of edge computing and deep learning technologies, optimizing model computation efficiency to accommodate hardware constraints of edge devices and developing lightweight deep learning algorithms to reduce resource consumption, deploying prediction models at the edge side will become more feasible.

Data availability statement

The dataset for this research was provided by a collaborating institution and contains sensitive information and usage restrictions. If other researchers need to obtain this dataset for further research or other reasonable requests, please contact the corresponding author.

Author contributions

XZ: Conceptualization, Formal analysis, Investigation, Methodology, Validation, Writing – original draft, Writing – review & editing. JunlW: Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing. JunW: Methodology, Resources, Supervision, Writing – original draft, Writing – review & editing. HW: Data curation, Validation, Writing – original draft, Writing – review & editing. LL: Conceptualization, Validation, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

XZ and JunlW were employed by State Grid Hebei Electric Power Company.

JunW, HW, and LL were employed by Henan XJ Metering Co., Ltd.

Publisher's note

References

Ahmad, T., Madonski, R., Zhang, D., Huang, C., and Mujeeb, A. (2022). Data-driven probabilistic machine learning in sustainable smart energy/smart energy systems: Key developments, challenges, and future research opportunities in the context of smart grid paradigm. *Renew. Sustain. Energy Rev.* 160, 112128. doi: 10.1016/j.rser.2022.112128

Akram, M. W., Li, G., Jin, Y., Chen, X., Zhu, C., and Ahmad, A. (2020). Automatic detection of photovoltaic module defects in infrared images with isolated and develop-model transfer deep learning. *Solar Energy* 198, 175–186. doi: 10.1016/j.solener.2020.01.055

Alipour, M., Aghaei, J., Norouzi, M., Niknam, T., Hashemi, S., and Lehtonen, M. (2020). A novel electrical net-load forecasting model based on deep neural networks and wavelet transform integration. *Energy* 205:118106. doi: 10.1016/j.energy.2020.118106

Bae, D. J., Kwon, B. S., and Song, K. B. (2021). XGBoost-based day-ahead load forecasting algorithm considering behind-the-meter solar PV generation. *Energies* 15:128. doi: 10.3390/en15010128

Borges, C. E., Kamara-Esteban, O., Castillo-Calzadilla, T., Andonegui, C. M., and Alonso-Vicaria, A. (2020). Enhancing the missing data imputation of primary substation load demand records. *Sustain. Energy, Grids Netw.* 23:100369. doi: 10.1016/j.segan.2020.100369

Dairi, A., Harrou, F., Sun, Y., and Khadraoui, S. (2020). Short-term forecasting of photovoltaic solar power production using variational auto-encoder driven deep learning approach. *Appl. Sci.* 10:8400. doi: 10.3390/app10238400

Ding, S., Li, R., and Tao, Z. A. (2021). novel adaptive discrete grey model with time-varying parameters for long-term photovoltaic power generation forecasting. *Energy Convers. Manage.* 227:113644. doi: 10.1016/j.enconman.2020.113644

Eom, H., Son, Y., and Choi, S. (2020). Feature-selective ensemble learning-based long-term regional PV generation forecasting. *IEEE Access* 8, 54620–54630. doi: 10.1109/ACCESS.2020.2981819

Espinoza, M., Joye, C., Belmans, R., and De Moor, B. (2005). Short-term load forecasting, profile identification, and customer segmentation: a methodology based on periodic time series. *IEEE Trans. Power Syst.* 20, 1622–1630. doi: 10.1109/TPWRS.2005.852123

Feng, Y., Hao, W., Li, H., Cui, N., Gong, D., and Gao, L. (2020). Machine learning models to quantify and map daily global solar radiation and photovoltaic power. *Renew. Sustain. Energy Rev.* 118:109393. doi: 10.1016/j.rser.2019.109393

Hafiz, F., Awal, M. A., de Queiroz, A. R., and Hussain, I. (2020). Real-time stochastic optimization of energy storage management using deep learning-based forecasts for residential PV applications. *IEEE Trans. Ind. Appl.* 56, 2216–2226. doi: 10.1109/TIA.2020.2968534

Hayes, B. P., Gruber, J. K., and Prodanovic, M. A. (2014). closed-loop state estimation tool for MV network monitoring and operation. *IEEE Trans. Smart Grid* 6, 2116–2125. doi: 10.1109/TSG.2014.2378035

Jin, L., Liu, L., Wang, X., Shang, M., Wang, F. -Y., et al. (2024). "Physical-informed neural network for mpc-based trajectory tracking of vehicles with noise considered," in *IEEE Transactions on Intelligent Vehicles*.

Kabilan, R., Chandran, V., Yogapriya, J., Karthick, A., Gandhi, P., Mohanavei, V., et al. (2021). Short-term power prediction of building integrated photovoltaic (BIPV) system based on machine learning algorithms. *Int. J. Photoen.* 2021, 1–11. doi: 10.1155/2021/5582418

Karimi, A. M., Fada, J. S., Parrilla, N. A., Pierce, B., Koyutürk, M., French, R. H., et al. (2020). Generalized and mechanistic PV module performance prediction

from computer vision and machine learning on electroluminescence images. *IEEE J. Photovolt.* 10, 878–887. doi: 10.1109/JPHOTOV.2020.2973448

Khan, W., Walker, S., and Zeiler, W. (2022). Improved solar photovoltaic energy generation forecast using deep learning-based ensemble stacking approach. *Energy* 240:122812. doi: 10.1016/j.energy.2021.122812

Kim, D., Kwon, D., Park, L., Kim, J., and Cho, S. (2020). Multiscale LSTM-based deep learning for very-short-term photovoltaic power generation forecasting in smart city energy management. *IEEE Syst. J.* 15, 346–354. doi: 10.1109/JSYST.2020.3007184

Korkmaz, D. (2021). SolarNet: a hybrid reliable model based on convolutional neural network and variational mode decomposition for hourly photovoltaic power forecasting. *Appl. Energy* 300:117410. doi: 10.1016/j.apenergy.2021.117410

Lee, Y. M., An, L., Liu, F., Horesh, R., Chae, Y. T., and Zhang, R. (2013). Applying science and mathematics to big data for smarter buildings. *Ann. N. Y. Acad. Sci.* 1295, 18–25. doi: 10.1111/nyas.12193

Li, N., Ma, L., Yu, G., Xue, B., Zhang, M., and Jin, J. (2023). Survey on evolutionary deep learning: Principles, algorithms, applications, and open issues. *ACM Comp. Surv.* 56, 1–34. doi: 10.1145/3603704

Litjens, G., Worrell, E., and Van Sark, W. (2018). Assessment of forecasting methods on performance of photovoltaic-battery systems. *Appl. Energy* 221, 358–373. doi: 10.1016/j.apenergy.2018.03.154

Liu, W., Tang, P., Liu, H., and Zhao, P. (2022). Intelligent voltage prediction of active distribution network with high proportion of distributed photovoltaics. *Energy Rep.*8, 894–903. doi: 10.1016/j.egyr.2022.08.142

Liu, Z. F., Li, L. L., Tseng, M. L., and Lim, M. (2020). Prediction short-term photovoltaic power using improved chicken swarm optimizer-extreme learning machine model. *J. Clean. Prod.* 248:119272. doi: 10.1016/j.jclepro.2019.119272

Liufu, Y., Jin, L., Shang, M., Wang, X., and Wang, F.-Y. (2024). ACP-incorporated perturbation-resistant neural dynamics controller for autonomoius vehicles. *IEEE Trans. Intell. Vehicles* 9, 4675–4686. doi: 10.1109/TIV.2023.3348632

Lopes, S. M. A., Cari, E. P. T., and Hajimirza, S. A. (2022). Comparative analysis of Artificial Neural Networks for Photovoltaic Power Forecast using remotes and local measurements. *J. Solar Energy Eng.* 144:021007. doi: 10.1115/1.4053031

Luo, X., Zhang, D., and Zhu, X. (2021). Deep learning based forecasting of photovoltaic power generation by incorporating domain knowledge. *Energy* 225:120240. doi: 10.1016/j.energy.2021.120240

Ma, L., Kang, H., Yu, G., Li, Q., and He, Q. (2024a). Single-domain generalized predictor for neural architecture search system. *IEEE Trans. Comp.* 73, 1400–1413. doi: 10.1109/TC.2024.3365949

Ma, L., Li, N., Guo, Y., Wang, X., Yang, S., Huang, M., et al. (2021). Learning to optimize: reference vector reinforcement learning adaption to constrained many-objective optimization of industrial copper burdening system. *IEEE Trans. Cybernet.* 52, 12698–12711. doi: 10.1109/TCYB.2021.3086501

Ma, L., Li, N., Yu, G., Geng, X., Cheng, S., Wang, X., et al. (2023). Pareto-wise ranking classifier for multi-objective evolutionary neural architecture search. *IEEE Trans. Evol.* 28, 570–581. doi: 10.1109/TEVC.2023.3314766

Ma, L., Li, N., Zhu, P., Tang, K., Khan, A., Wang, F., et al. (2024b). A novel fuzzy neural network architecture search framework for defect recognition with uncertainties. *IEEE Trans. Fuzzy Syst.* 32, 3274–3285. doi: 10.1109/TFUZZ.2024.3373792

Marinescu, A., Harris, C., Dusparic, I., Clarke, S., and Cahill, V. (2013). "Residential electrical demand forecasting in very small scale: an evaluation of forecasting methods," in *2013 2nd International Workshop on Software Engineering Challenges for the Smart Grid (SE4SG)* (San Francisco, CA: IEEE), 25–32. doi: 10.1109/SE4SG.2013.6596108

Markovics, D., and Mayer, M. J. (2022). Comparison of machine learning methods for photovoltaic power forecasting based on numerical weather prediction. *Renew. Sustain. Energy Rev.* 161:112364. doi: 10.1016/j.rser.2022.112364

Mellit, A., Pavan, A. M., and Lughi, V. (2021). Deep learning neural networks for short-term photovoltaic power forecasting. *Renew. Energy* 172, 276–288. doi: 10.1016/j.renene.2021.02.166

Polo, J., Martín-Chivelet, N., Alonso-Abella, M., Sanz-Salz, C., and de la Cuz, M. (2023). Exploring the PV power forecasting at building façades using gradient boosting methods. *Energies* 16:1495. doi: 10.3390/en16031495

Qadir, Z., Khan, S. I., Khalaji, E., Munawar, H. S., Al-Turjman, F., Mohmud, P., et al. (2021). Predicting the energy output of hybrid PV–wind renewable energy system using feature selection technique for smart grids. *Energy Rep.* 7, 8465–8475. doi: 10.1016/j.egyr.2021.01.018

Rai, P., Londhe, N. D., and Raj, R. (2021). Fault classification in power system distribution network integrated with distributed generators using CNN. *Electric Power Syst. Res.* 192:106914. doi: 10.1016/j.epsr.2020.106914

Rana, M., and Rahman, A. (2020). Multiple steps ahead solar photovoltaic power forecasting based on univariate machine learning models and data re-sampling. *Sustain. Energy, Grids Netw.* 21:100286. doi: 10.1016/j.segan.2019.100286

Razavi, S. E., Arefi, A., Ledwich, G., Nourbakhsh, G., Smith, D. B., Minakshi, M., et al. (2020). From load to net energy forecasting: short-term residential forecasting for the blend of load and PV behind the meter. *IEEE Access* 8, 224343–224353. doi: 10.1109/ACCESS.2020.3044307

Shivam, K., Tzou, J. C., and Wu, S. C. A. (2021). multi-objective predictive energy management strategy for residential grid-connected PV-battery hybrid systems based on machine learning technique. *Energy Conv. Manage.* 237:114103. doi: 10.1016/j.enconman.2021.114103

Tsekouras, G. J., Dialynas, E. N., Hatziargyriou, N. D., and Kavatza, S. (2007). A non-linear multivariable regression model for midterm energy forecasting of power systems. *Electric Power Syst. Res.* 77, 1560–1568. doi: 10.1016/j.epsr.2006.11.003

Wang, S., Dong, P., and Tian, Y. A. (2017). novel method of statistical line loss estimation for distribution feeders based on feeder cluster and modified XGBoost. *Energies* 10:2067. doi: 10.3390/en10122067

Zang, H., Cheng, L., Ding, T., Cheung, K., Wei, Z., and Sun, G. (2020). Day-ahead photovoltaic power forecasting approach based on deep convolutional neural networks and meta learning. *Int. J. Elect. Power Energy Syst.* 118:105790. doi: 10.1016/j.ijepes.2019.105790

Zhang, Y., Qin, C., Srivastava, A. K., Jin, C., and Sharma, R. (2020). Data-driven day-ahead PV estimation using autoencoder-LSTM and persistence model. *IEEE Trans. Ind. Appl.* 56, 7185–7192. doi: 10.1109/TIA.2020.3025742

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 15 July 2024
DOI 10.3389/fnbot.2024.1431034

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Xiufang Chen,
Lanzhou University, China
Jiawang Tan,
Chinese Academy of Sciences (CAS), China

*CORRESPONDENCE
Yanqiu Li
liyanqiu@jlenu.edu.cn

RECEIVED 11 May 2024
ACCEPTED 10 June 2024
PUBLISHED 15 July 2024

CITATION
Li Y, Liu H and Gao H (2024) Online learning fuzzy echo state network with applications on redundant manipulators.
Front. Neurorobot. 18:1431034.
doi: 10.3389/fnbot.2024.1431034

Online learning fuzzy echo state network with applications on redundant manipulators

Yanqiu Li[1]*, Huan Liu[1] and Hailong Gao[2]

[1]School of Data Science and Artificial Intelligence, Jilin Engineering Normal University, Changchun, China, [2]Department of Basic Sciences, Jilin Jianzhu University, Changchun, China

Redundant manipulators are universally employed to save manpower and improve work efficiency in numerous areas. Nevertheless, the redundancy makes the inverse kinematics of manipulators hard to address, thus increasing the difficulty in instructing manipulators to perform a given task. To deal with this problem, an online learning fuzzy echo state network (OLFESN) is proposed in the first place, which is based upon an online learning echo state network and the Takagi–Sugeno–Kang fuzzy inference system (FIS). Then, an OLFESN-based control scheme is devised to implement the efficient control of redundant manipulators. Furthermore, simulations and experiments on redundant manipulators, covering UR5 and Franka Emika Panda manipulators, are carried out to verify the effectiveness of the proposed control scheme.

KEYWORDS

echo state network (ESN), fuzzy inference system (FIS), online learning, redundant manipulators, optimization

1 Introduction

To improve production efficiency and set themselves free from manpower, robots have come into being and undergone expeditious and substantial progress, with plentiful and triumphant applications in numerous areas (Sun et al., 2023b; Liu et al., 2024). Therefore, redundant manipulators that possess more degrees of freedom (DOFs) than non-redundant ones to fulfill a specific task stand out and have been subject to in-depth and comprehensive investigations (Liao et al., 2016; Liu et al., 2023). More precisely, by virtue of the additional DOFs, they are capable of executing some secondary tasks while performing the primary task, such as obstacle avoidance, optimizing joint torques, and enhancing operability (Jin et al., 2017a; Sun et al., 2022a). For that reason, research on the mechanisms and applications of redundant manipulators is in full swing. However, it is worth mentioning that the additional DOFs result in troubles and challenges for controlling manipulators efficiently and precisely (Zhang et al., 2019; Zhao et al., 2020). Therefore, it imports the demand to devise and construct a potent control scheme of redundant manipulators (Jin et al., 2017b; Liao et al., 2022).

With a sophisticated and ingenious nervous system, humans are capable of performing a variety of complicated and intractable missions by learning from recent experiences, which is the most prominent difference and superiority compared with other creatures (Wang et al., 2016; Liao et al., 2024b). Therefore, this has opened up a new avenue for the control of manipulators. That is, manipulators can accomplish the assigned task with high efficiency by simulating the learning ability of humans. Taking the neural network (NN) (Su et al., 2023a; Wei and Jin, 2024) and fuzzy inference system (FIS) (Vargas et al., 2024) into account, both of them attempt to simulate the thinking and decision-making processes of humans in a certain way. Therefore, they have garnered the attention of researchers,

and a lot of effort has been put into integrating them with manipulator control systems to improve the completion of the task and meet the requirements of different scenarios. For instance, Yoo and Ham (2000) present adaptive control schemes for manipulators, in which the parameter uncertainty is handled via the FIS. Afterward, aiming at the tracking control of the end-effector for manipulators, an FIS-based controller is designed by Yilmaz et al. (2022), in which the centers and widths of the membership functions are adjusted adaptively, thus promoting the learning power of the controller. Recently, Yilmaz et al. (2023) devised an FIS-based output-feedback controller for the joint space tracking of manipulators, in which the demands for joint velocity and knowledge of manipulators are eliminated.

In recent times, a surge of research has come into view in the realm of the echo state network (ESN), a sort of recurrent neural network (RNN), which overcomes certain problems hindering the investigations and applications of RNNs, such as gradient vanishing and gradient exploding (Rodan and Tino, 2011; Chen et al., 2023). The core of ESN lies in the reservoir, which is a large, sparse network in charge of capturing the dynamic behavior of input information. Particularly in the ESN, both input and reservoir weights are generated at random, and one needs to put effort into obtaining the output weights by figuring out the weighted sum of outputs (Lukoševičius, 2012). Considering another network, the extreme learning machine (ELM) (Huang et al., 2006) is a feedforward network with a hidden layer. Weights and biases for the hidden layer are appointed randomly, while the training of the network focuses on determining output weights through the least squares method. Therefore, from the perspective of this point, the ELM, ESN, and FIS share a certain similarity, and thus, a great deal of work has been carried out that builds and verifies the bridges between them (Sun et al., 2007; Ribeiro et al., 2020). By integrating these networks and taking advantage of their strengths, some extraordinary work is presented and utilized in various domains to address different issues. Concentrating on function approximation and classification problems, a fuzzy ELM with the capacity for online learning was devised by Rong et al. (2009). Compared with other existing mechanisms it presents remarkable superiority with decent accuracy and reduced training time. Motivated by this, aiming at efficient control of redundant manipulators, this study proposes an online learning fuzzy ESN (OLFESN). To be more specific, the proposed OLFESN is designed, based on an online learning strategy for ESN, to erect an efficient control scheme for redundant manipulators, while the FIS is also incorporated to improve the accuracy and efficiency of the proposed network. Then, a corresponding control scheme for redundant manipulators is constructed. The rest of this study is organized as follows: Section 2 makes known some preliminary steps to lay the foundation for this study. Then, the OLFESN is proposed, based on which the control scheme for redundant manipulators is devised in Section 3. In Section 4, simulations and experiments are carried out to investigate the feasibility and effectiveness of the proposed control scheme. In the end, Section 5 concludes this study.

2 Preliminaries

In this section, the forward kinematics of redundant manipulators, the Takagi–Sugeno–Kang (TSK) fuzzy system, and ESN are briefly reviewed, which are the bases of the proposed OLFESN.

2.1 Forward kinematics of redundant manipulators

The forward kinematics equation that depicts the non-linear transformation of redundant manipulators from the joint angle $\mathbf{q} \in \mathbb{R}^a$ to the Cartesian position $\mathbf{r} \in \mathbb{R}^b$ with $a > b$ can be depicted as

$$\Upsilon(\mathbf{q}) = \mathbf{r}, \tag{1}$$

where $\Upsilon(\bullet)$ signifies the non-linear mapping function, which depends upon the structural properties of redundant manipulators (Sun et al., 2022b; Zhang et al., 2022). Where after, evaluating the derivative of Equation (1) in terms of time contributes to

$$J(\mathbf{q})\dot{\mathbf{q}} = \dot{\mathbf{r}}, \tag{2}$$

in which $J(\mathbf{q}) = \partial\Upsilon(\mathbf{q})/\partial\mathbf{q} \in \mathbb{R}^{b\times a}$ denotes the Jacobian matrix; $\dot{\mathbf{q}}$ denotes the angular velocity; $\dot{\mathbf{r}}$ denotes the velocity of the end-effector (Yan et al., 2024). Heretofore, the non-linear transformation (Equation 1) is converted to the affine system (Equation 2) with the convenience of gaining the redundancy solution of redundant manipulators (Sun et al., 2023a).

2.2 Takagi–Sugeno–Kang fuzzy system

In the TSK fuzzy system with given input $\alpha = [\alpha_1;\alpha_2;\cdots;\alpha_m] \in \mathbb{R}^m$, the k-th rule can be depicted as Kerk et al. (2021) and Zhang et al. (2023):

$$\begin{aligned} &\textbf{Rule k:IF } \alpha_1 \textbf{is } \mathbf{A_{1k}}, \alpha_2 \textbf{ is } \mathbf{A_{2k}}, \cdots, \alpha_m \textbf{is } \mathbf{A_{mk}}, \\ &THEN\ \chi_k = \beta_{0k} + \beta_{1k}\alpha_1 + \beta_{2k}\alpha_2 + \cdots + \beta_{mk}\alpha_m, \end{aligned} \tag{3}$$

where $k = 1, 2, \cdots, \tilde{k}$ is the index of the fuzzy rule with $\tilde{k}$ being the number of fuzzy rules; A_{mk} denotes the fuzzy subset of the m-th element of input α in the k-th rule; χ_k signifies the output of the k-th rule; $\beta_{\tilde{m}k}(\tilde{m} = 0, 1, \cdots, m)$ is the consequent coefficient of the k-th rule. Considering the m-th element of input in the k-th rule, the degree to which it matches the fuzzy subset A_{mk} is measured by its membership function $\zeta_{A_{mk}}(\alpha_m)$, which can be any bounded non-constant piecewise continuous function (Rezaee and Zarandi, 2010). Let $\bigotimes$ denote the fuzzy conjunction operation, and then the firing strength (if part) of the k-th rule is defined as

$$O_k(\alpha, \mathbf{p_k}) = \zeta_{A_{1k}}(\alpha_1, p_{1,k}) \otimes \zeta_{A_{2k}}(\alpha_2, p_{2,k}) \otimes \cdots \otimes \zeta_{A_{mk}}(\alpha_m, p_{m,k}), \tag{4}$$

where $\mathbf{p}_k$ is the parameter of membership function $\zeta(\bullet)$ in the k-th rule. Normalizing (Equation 4), there is

$$\Psi(\alpha, \mathbf{p_k}) = \frac{O_k(\alpha, \mathbf{p_k})}{\sum_{k=1}^{\tilde{k}} O_k(\alpha, \mathbf{p_k})}. \tag{5}$$

Ultimately, for the input α, the output of the TSK fuzzy model can be obtained as

$$\tilde{y} = \frac{\sum_{k=1}^{\tilde{k}} \theta_k O_k(\alpha, \mathbf{p_k})}{\sum_{k=1}^{\tilde{k}} O_k(\alpha, \mathbf{p_k})} = \sum_{k=1}^{\tilde{k}} \theta_k \Psi(\alpha, \mathbf{p_k}), \tag{6}$$

with $\theta_k = (\theta_{k1}, \theta_{k2}, \cdots, \theta_{km})$.

2.3 Echo state network

The ESN is composed of an input layer, a reservoir, and an output layer, which enjoy l, r, and o neurons, respectively (Calandra et al., 2021). For a complete network, the input layer, reservoir, and output layer re connected by input weights $W_{in} \in \mathbb{R}^{r\times l}$ and output weights $W_{out} \in \mathbb{R}^{o\times r}$, respectively, while the internal neurons of the reservoir are connected to each other by dint of $W_{res} \in \mathbb{R}^{r\times r}$ (Chen et al., 2024a). In particular, the spectral radius of W_{res} needs to be <1 to capture the echo state property. At the time of step i, designate input and reservoir states as $\mathbf{x}_i = [x_1; x_2; \cdots; x_l] \in \mathbb{R}^l$ and $\iota_i = [\iota_1; \iota_2; \cdots; \iota_r] \in \mathbb{R}^r$, respectively. The reservoir is updated through

$$\iota_i = f\left(W_{in}x_i + W_{res}\iota_{(\mathbf{i}-\mathbf{1})}\right), \tag{7}$$

and the output of the network is

$$y_i = g(W_{out}\iota_i), \tag{8}$$

with $\mathbf{y}_i = \left[y_1; y_2; \cdots; y_o\right] \in \mathbb{R}^o$. Furthermore, for working out the output weights, keep track of reservoir state and outputs in matrices $\Lambda = \left[\iota_1, \iota_2, \cdots, \iota_{\tilde{i}}\right] \in \mathbb{R}^{r\times\tilde{i}}$ and $Y = \left[\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{\tilde{i}}\right] \in \mathbb{R}^{o\times\tilde{i}}$, respectively, during training, where $\tilde{i}$ denotes the number of training samples. Where after, by solving

$$\min_{W_{\text{out}}} : \|Y - W_{out}\Lambda\|_{2,}^2 \tag{9}$$

the output weights are obtained

$$W_{out} = Y\Lambda^T(\Lambda\Lambda^T)^{-1} \tag{10}$$

where the superscripts $^{\mathrm{T}}$ and $^{-1}$ represent transpose and inversion operations of a matrix, respectively (Su et al., 2023b; Liao et al., 2024a).

3 Online learning fuzzy echo state network

Stimulated by the commonalities between ESN and FIS, OLFESN is proposed in this section. Then, an OLFESN-based control scheme for redundant manipulators is devised.

3.1 OLFESN

Considering (Equation 4), the firing strength (if any) in the TSK fuzzy system involves multiple fuzzy conjunction operations, providing sufficient computing power for thoroughly exploring and utilizing input information. Furthermore, each rule is normalized to ensure that different rules have a comparable contribution to the system. Similarly, in the ESN, it is the reservoir that is responsible for implementing the above function, by which the low-dimensional input is mapped to a high-dimensional dynamic space. In addition, the outputs of different reservoirs are adjusted to the same extent with the aid of the activation function $f(\bullet)$, which plays the same role as Equation (5). Therefore, the reservoir is adopted to reveal the firing strength normalized in the proposed OLESN. Specifically, the OLESN with $\tilde{k}$ reservoirs is established as follows:

Given training samples $\beth = \left\{\left(\mathbf{x}_i, \mathbf{y}_i\right)\right\}_{i=1}^{\tilde{i}}$, the state of the k-th reservoir is updated via

$$\iota_{ki} = f_k(W_{in}\mathbf{x}_i + W_{res}\iota_{k(i-1)}),\ i = 1, 2, \cdots, \tilde{i}, \tag{11}$$

where $f_k(\bullet)$ denotes the activation function of the k-th reservoir, and $\tilde{i}$ is the number of training samples. Collect all states of the k-th reservoir in $\Xi_k = [\iota_{k1}, \iota_{k2}, \cdots, \iota_{k\tilde{i}}$, and then integrate all $\tilde{k}$ reservoirs elicited

$$\Lambda = \Xi_1\Xi_2\cdots\Xi_{\tilde{k}}. \tag{12}$$

Thus, the output of the fuzzy ESN (FESN) can be formulated as

$$Y = W_{out}\Lambda, \tag{13}$$

with $Y = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{\tilde{i}}]$. Similarly to Equation (10), output weights are obtained via

$$W_{out} = Y\Lambda^T(\Lambda\Lambda^T)^{-1}. \tag{14}$$

At this point, the derivation of FESN is complete. Therewith, taking into account the need for online learning, the OLFESN is proposed, which incorporates the FESN and the online learning strategy for ESN. To be more specific, when data shows up constantly, the OLFESN is summarized as follows:

3.2 Initialization phase

a. Given the initial training samples $\beth_0 = \left\{\left(\mathbf{x}_i, \mathbf{y}_i\right)\right\}_{i=1}^{\tilde{i}_0}$, update and transcribe the state of all $\tilde{k}$ reservoirs using Equation 11.

b. Taking advantage of Equation 12, figure out the initial state matrix Λ_0 for FESN.

c. Compute the initial output weights $W_{out}^0 = T_0\Lambda_0^T Y_0$ with $T_0 = \left(\Lambda_0^T\Lambda_0\right)^{-1}$ and $Y_0 = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{\tilde{i}_0}]$.

d. Let $p = 0$.

3.3 Sequential learning phase

a. With the new sample set

$$\beth_{p+1} = \left\{\left(\mathbf{x}_i, \mathbf{y}_i\right)\right\}_{i=(\sum_{j=0}^{p}\tilde{i}_j)+1}^{\sum_{j=0}^{p+1}\tilde{i}_j},$$

solve problem

$$\left\| W_{out}^{p+1}[\Lambda_p,\ \Lambda_{p+1}\ ] - [Y_p,\ Y_{p+1}\ ]\right\|_2^2, \tag{15}$$

where $\tilde{i}_{p+1}$ signifies the count of samples in the $(p+1)$-th set; Λ_{p+1} is the corresponding reservoir state, obtained by Equations 11, 12; $Y_{p+1} = [\mathbf{y}_{(\sum_{j=0}^{p}\tilde{i}_j)+1}, \cdots, \mathbf{y}_{(\sum_{j=0}^{p+1}\tilde{i}_j)+1}]$.

b. Let $\Psi_p = H_p^{-1}$ with $H_P = [\Lambda_p, \Lambda_{p+1}][\Lambda_p, \Lambda_{p+1}]^T$.
c. Update output weights

$$\Psi_{p+1} = \Psi_p - \Psi_p \Lambda_{p+1} (I + \Lambda_{p+1}^T \Psi_p \Lambda_{p+1})^{-1} \Lambda_{p+1}^T \Psi_p,$$
$$W_{out}^{p+1} = W_{out}^p + (Y_{p+1} - W_{out}^p \Lambda_{p+1}) \Lambda_{p+1}^T \Psi_{p+1}. \quad (16)$$

d. Let $p = p + 1$. (Back to step 2).

Remark 1: For the case that the new samples come out one by one, with the aid of the Sherman-Morrison formula (Chen et al., 2024b), Equation 16 is further simplified as

$$\Psi_{p+1} = \Psi_p - \frac{\Psi_p \iota_{p+1} \iota_{p+1}^T \Psi_p}{1 + \iota_{p+1}^T \Psi_p \iota_{k1}},$$
$$W_{out}^{p+1} = W_{out}^p + (\mathbf{y}_{p+1} - W_{out}^p \iota_{p+1}) \iota_{p+1}^T \Psi_{p+1}. \quad (17)$$

3.4 OLFESN-based control scheme

In this section, an OLFESN-based control scheme for redundant manipulators is developed for performing the given missions. At moment t, define $\theta_a(t)$ and $\Delta\theta_a(t)$ as the actual joint angle and actual joint angle increment, respectively. Meanwhile, the actual and desired positions of the end-effector are denoted by $\zeta_a(t)$ and $\zeta_d(t)$, respectively. Correspondingly, at moment $t+1$, the desired position increment for the end-effector is expressed as $\Delta\zeta(t+1) = \zeta_d(t) - \zeta_a(t)$. Incorporate $\theta_a(t)$, $\Delta\theta_a(t)$, and $\Delta\zeta(t+1)$, which is the input of the OLFESN and denoted by $\mathbf{x}(t)$ for the convenience of subsequent expressions. Then, applying Equations 11–13, we gain the joint angle increment $\Delta\theta_a(t+1)$ for the next moment, i.e., the output of OLFESN. Hence, the control signal for the next moment is acquired, i.e., $\theta_a(t+1) = \theta_a(t) + \Delta\theta_a(t+1)$.

Note that, in the OLFESN, there is a premise that sample $(\mathbf{x}(t), \mathbf{y}(t))$ is accessible all the time. However, for the proposed scheme, the desired joint angle increment $\Delta\theta_d(t+1)$, i.e., $\mathbf{y}(t)$, is unrevealed in reality. In addition, taking into account output weights W_{out}, it ought to be updated in real-time to generate the control signal. An accepted wisdom is making use of the teaching signal to update output weights W_{out}. More specifically, the error $\epsilon(t+1)$ between the desired joint angle increment $\Delta\theta_d(t+1)$ and the actual one $\Delta\theta_a(t+1)$ plays a part in the teaching signal in the proposed scheme.

Informed by Equation (2), the transformation between joint angle increment $\Delta\theta(t)$ and position increment $\Delta\zeta(t)$ of the end-effector is devised as

$$J(t)\Delta\theta(t) = \Delta\zeta(t). \quad (18)$$

Then, we have

$$\zeta_d(t+1) - \zeta_a(t+1) = J(t+1)(\theta_d(t+1) - \theta_a(t+1))$$
$$= J(t+1)(\theta_a(t) - \Delta\theta_d(t+1) - (\theta_a(t) + \Delta\theta_a(t+1)))$$
$$= J(t+1)(\Delta\theta_d(t+1) - \Delta\theta_a(t+1)). \quad (19)$$

Solving Equation 19, the teaching signal is collected as

$$\epsilon(t+1) = J^+(t+1)(\zeta_d(t+1) - \zeta_a(t+1)). \quad (20)$$

Until now, the proposed control scheme for redundant manipulators based on the above-mentioned teaching signal and OLFESN has been constructed as

$$\Psi(t+1) = \Psi(t) - \Psi(t)\Lambda(t+1)(I + \Lambda(t+1)^T\Psi(t)\Lambda(t+1))^{-1}$$
$$\Lambda(t+1)^T\Psi(t), \quad (21)$$
$$W_{out}(t+1) = W_{out}(t) + \epsilon(t+1)\Lambda(t+1)^T\Psi(t+1),$$

which is outlined and summarized in Algorithm 1.

Input: r: the number of neurons in the reservoir
  $\tilde{k}$: the number of reservoir
  $\boldsymbol{\theta}_a(0)$: the initial joint angle
  $\boldsymbol{\zeta}_a(0)$: the initial position of end-effector
  $\boldsymbol{\zeta}_d$: the desired trajectory of the end-effector
Output: $\boldsymbol{\theta}_a(t+1)$: the control signal
Initialize: $\iota_k(0) = \mathbf{0}$; $\Delta\boldsymbol{\theta}_a(0) = \mathbf{0}$; $W_{out}(0) = \mathbf{0}$;
for $t = 0 : T$ **do**
  $\Delta\boldsymbol{\zeta}(t+1) = \boldsymbol{\zeta}_d(t+1) - \boldsymbol{\zeta}_a(t)$;
  $\boldsymbol{x}(t+1) = [\boldsymbol{\theta}_a(t); \Delta\boldsymbol{\zeta}(t+1); \Delta\boldsymbol{\theta}_a(t)]$;
  $\iota_k(t+1) = f(W_{in}\boldsymbol{x}(t+1) + W_{res}\iota_k(t)), k = 1, 2, \ldots, \tilde{k}$;
  $\lambda(t+1) = \iota_1(t+1)\iota_2(t+1)\cdots\iota_{\tilde{k}}(t+1)$;
  $\Delta\boldsymbol{\theta}_a(t+1) = W_{out}(t)\lambda(t+1)$;
  $\boldsymbol{\theta}_a(t+1) = \boldsymbol{\theta}_a(t) + \Delta\boldsymbol{\theta}_a(t+1)$;
  Control the manipulator using $\boldsymbol{\theta}_a(t+1)$;
  Obtain the actual position of end-effector $\boldsymbol{\zeta}_a(t+1)$;
  Compute Jacobian matrix $J(t+1)$;
  $\boldsymbol{\epsilon}(t+1) = J^{\dagger}(t+1)(\boldsymbol{\zeta}_d(t+1) - \boldsymbol{\zeta}_a(t+1))$;
  $\Psi(t+1) = \Psi(t) - \Psi(t)\Lambda(t+1)(I + \Lambda(t+1)^{\mathrm{T}}\Psi(t)\Lambda(t+1))^{-1}$
  $\Lambda(t+1)^{\mathrm{T}}\Psi(t)$;
  $W_{out}(t+1) = W_{out}(t) + \boldsymbol{\epsilon}(t+1)\Lambda(t+1)^{\mathrm{T}}\Psi(t+1)$.
end for

Algorithm 1. Proposed Control Scheme

4 Illustrative examples

In this section, simulations on redundant manipulators are devised and executed, covering a 6-DOF manipulator and a 7-DOF one, to verify the effectiveness and feasibility of the proposed scheme (Equation 21).

4.1 UR5

A UR5 manipulator is employed with the aid of the proposed scheme (Equation 21) in this simulation, which possesses 6 DOFs and is explicitly revealed in Zheng et al. (2019) and Chico et al. (2021). The task is to track a four-leaf clover path within 20 s, where the initial angle state is $\theta(0) = [0;\ -\pi/2;\ 2\pi/3;\ 0;\ 0;\ 0]$ rad. With regard to OLFESN, the input weights W_{in} and internal connection weights of reservoir W_{res} are randomly initialized to

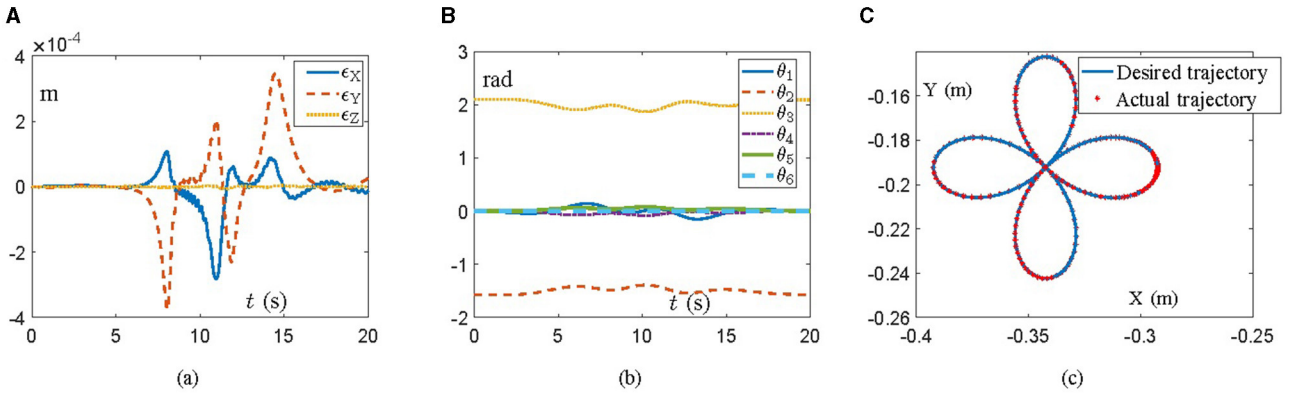


FIGURE 1
Simulative experiment results on the UR5 manipulator synthesized by the proposed scheme (Equation 21) for tracking a four-leaf clover path. **(A)** Profiles of the tracking error. **(B)** Profiles of the joint angle. **(C)** The desired trajectory and the actual trajectory.

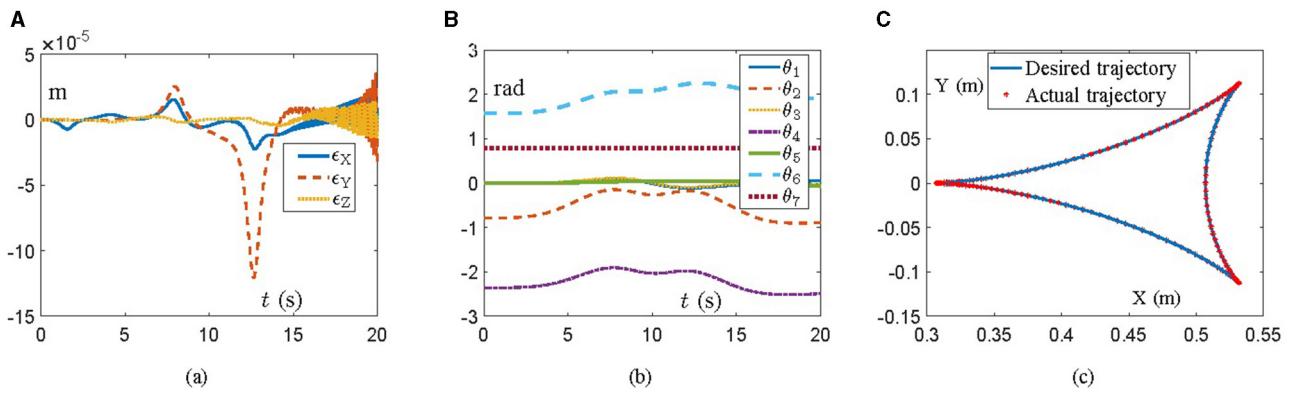


FIGURE 2
Simulative experiment results on the Frank Emika Panda manipulator synthesized by the proposed scheme (Equation 21) for tracking a tricuspid valve trajectory. **(A)** Profiles of the tracking error. **(B)** Profiles of the joint angle. **(C)** The desired trajectory and the actual trajectory.

$[-0.5, 0.5]$ by using MATLAB's 2022 *rand*(•) function. In addition, we bring in a total of three reservoirs, each with 500 neurons and the hyperbolic tangent function (tanh(•)), while the spectral radius is set to 0.8. Specifically, simulation results are exhibited in Figure 1, where Figure 1A illustrates the position errors of the end-effector during task execution. One can observe that the manipulator, with the aid of the proposed scheme (Equation 21), does the job with flying colors, and the position error of the end-effector is of the order 10^{-4} m. Correspondingly, trails of joint angles and task completion are shown in Figures 1B, C, respectively. Note that, during the task, the joint angles of the manipulator are evolving in a gentle manner, which is capable of reducing the wear between mechanical components to a certain extent, thus elongating the service life of the manipulator. In the end, Figure 1C further indicates that the task of tracking the four-leaf clover path is commendably accomplished by the manipulator, with the actual trajectory synthesized by the proposed scheme (Equation 21) excellently covering the desired one.

4.2 Franka Emika panda manipulator

In this part, the simulation of a Franka Emika Panda manipulator is designed and carried out to further verify the effectiveness and feasibility of the proposed scheme (Equation 21). The Franka Emika Panda is a 7-DOF manipulator with structural information covered by Shahid et al. (2020) and Gaz et al. (2019), which is necessary to track a tricuspid valve trajectory within 20 s. The initial angle state is $\theta(0) = [0;\ -\pi/4;\ 0;\ -3\pi/4;\ 0;\ \pi/2;\ \pi/4]$ rad, while the other parameters are in line with those in Section 4.1. Figure 2 reveals simulation results, where position errors of the end-effector are exhibited in Figure 2A. Viewing position errors, one can lightly draw the conclusion that the Franka Emika Panda manipulator controlled by the proposed scheme (Equation 21) finishes the given task successfully, with the position error being of the order 10^{-5} m. Then, pay attention to the variation of joint angles and task completion, which are depicted in Figures 2B,

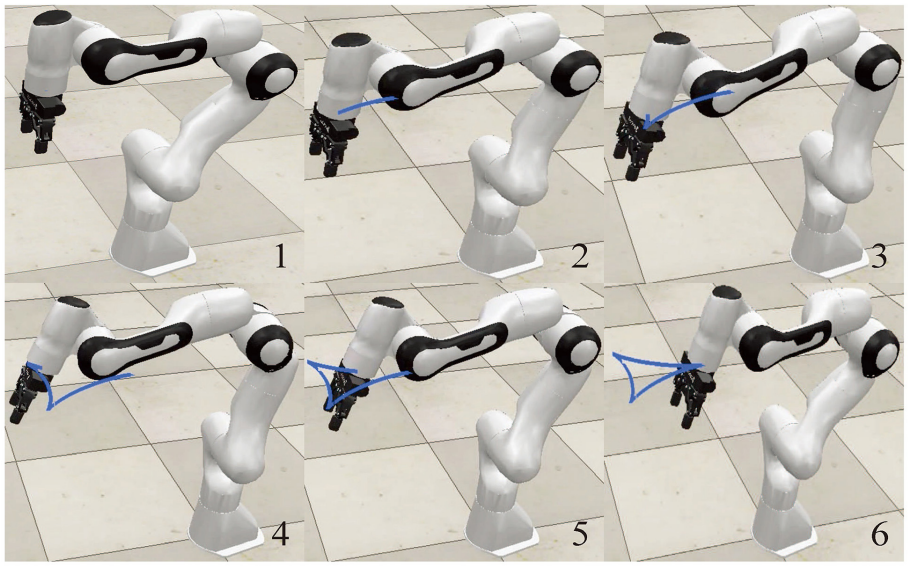


FIGURE 3
Snapshots of the Franka Emika Panda manipulator simulated on the (V-REP) platform for tracking the tricuspid valve trajectory with the aid of the proposed scheme (Equation 21).

C, respectively. All these results indicate the success of the task, which further verifies the feasibility and effectiveness of the proposed scheme (Equation 21) in the field of robot control. Furthermore, the corresponding simulation experiments are executed on the virtual robot experimentation platform (V-REP) to vividly simulate task execution. Snapshots of the Franka Emika Panda manipulator with the aid of the proposed scheme (Equation 21) are displayed in Figure 3, from which we can observe that the Franka Emika Panda manipulator safely and efficiently performs the task of tracking the tricuspid valve trajectory, thus further verifying the reliability of the above simulation results and the practicability of the proposed scheme (Equation 21).

5 Conclusion

Based on the online learning strategy for ESN and FIS, an OLFESN has been proposed, in which the new data is allowed to arrive one by one or in blocks. There are no additional restrictions on the size of blocks, thus highly extending the application scenarios of the proposed OLFESN. Subsequently, to cope with the complicated control problem of redundant manipulators, an OLFESN-based control scheme has been constructed from a kinematics point of view. In the end, simulations and experiments on the UR5 and Franka Emika Panda manipulators have been carried out and confirmed the effectiveness and feasibility of the proposed control scheme (Equation 21). Incorporating joint constraints into the proposed scheme (Equation 21) is a future research direction, that is capable of improving the safety and efficiency of task execution.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Ethics statement

The manuscript presents research on animals that do not require ethical approval for their study.

Author contributions

YL: Formal analysis, Funding acquisition, Methodology, Writing – original draft, Writing – review & editing. HL: Data curation, Methodology, Writing – original draft. HG: Data curation, Formal analysis, Writing – original draft.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Science and Technology Project of Jilin Province under Grant JJKH20240243KJ.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Calandra, M., Patanè, L., Sun, T., Arena, P., and Manoonpong, P. (2021). Echo state networks for estimating exteroceptive conditions from proprioceptive states in quadruped robots. *Front. Neurorobot.* 15:655330. doi: 10.3389/fnbot.2021.655330

Chen, X., Jin, L., and Hu, B. (2024a). A cerebellum-inspired control scheme for kinematic control of redundant manipulators. *IEEE Trans. Industr. Electr.* 71, 7542–7550. doi: 10.1109/TIE.2023.3312427

Chen, X., Jin, L., and Li, S. (2024b). An online learning strategy for echo state network. *IEEE Trans. Syst. Man Cyber. Syst.* 54, 644–655. doi: 10.1109/TSMC.2023.3319357

Chen, X., Liu, M., and Li, S. (2023). Echo state network with probabilistic regularization for time series prediction. *IEEE/CAA J. Autom. Sinica* 10, 1743–1753. doi: 10.1109/JAS.2023.123489

Chico, A., Cruz, P. J., Vásconez, J. P., Benalcázar, M. E., álvarez, R., Barona, L., et al. (2021). "Hand gesture recognition and tracking control for a virtual UR5 robot manipulator," in *2021 IEEE Fifth Ecuador Technical Chapters Meeting (ETCM)*, 1–6.

Gaz, C., Cognetti, M., Oliva, A., Robuffo Giordano, P., and De Luca, A. (2019). Dynamic identification of the Franka Emika Panda robot with retrieval of feasible parameters using penalty-based optimization. *IEEE Robot. Autom. Lett.* 4, 4147–4154. doi: 10.1109/LRA.2019.2931248

Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing* 70, 489–501. doi: 10.1016/j.neucom.2005.12.126

Jin, L., Li, S., Liao, B., and Zhang, Z. (2017a). Zeroing neural networks: a survey. *Neurocomputing* 267, 597–604. doi: 10.1016/j.neucom.2017.06.030

Jin, L., Liao, B., Liu, M., Xiao, L., Guo, D., and Yan, X. (2017b). Different-level simultaneous minimization scheme for fault tolerance of redundant manipulator aided with discrete-time recurrent neural network. *Front. Neurorobot.* 11:50. doi: 10.3389/fnbot.2017.00050

Kerk, Y. W., Teh, C. Y., Tay, K. M., and Lim, C. P. (2021). Parametric conditions for a monotone TSK fuzzy inference system to be an n-ary aggregation function. *IEEE Trans. Fuzzy Syst.* 29, 1864–1873. doi: 10.1109/TFUZZ.2020.2986986

Liao, B., Han, L., Cao, X., Li, S., and Li, J. (2024a). Double integral-enhanced zeroing neural network with linear noise rejection for time-varying matrix inverse. *CAAI Trans. Intell. Technol.* 9, 197–210. doi: 10.1049/cit2.12161

Liao, B., Hua, C., Xu, Q., Cao, X., and Li, S. (2024b). Inter-robot management via neighboring robot sensing and measurement using a zeroing neural dynamics approach. *Exp. Syst. Applic.* 244:122938. doi: 10.1016/j.eswa.2023.122938

Liao, B., Wang, Y., Li, J., Guo, D., and He, Y. (2022). Harmonic noise-tolerant ZNN for dynamic matrix pseudoinversion and its application to robot manipulator. *Front. Neurorobot.* 16. doi: 10.3389/fnbot.2022.928636

Liao, B., Zhang, Y., and Jin, L. (2016). Taylor $O(h^3)$ discretization of ZNN models for dynamic equality-constrained quadratic programming with application to manipulators. *IEEE Trans. Neural Netw. Learn. Syst.* 27, 225–237. doi: 10.1109/TNNLS.2015.2435014

Liu, M., Li, Y., Chen, Y., Qi, Y., and Jin, L. (2024). "A distributed competitive and collaborative coordination for multirobot systems," in *IEEE Transactions on Mobile Computing*, 1–13. doi: 10.1109/TMC.2024.3397242

Liu, Y., Liu, K., Wang, G., Sun, Z., and Jin, L. (2023). Noise-tolerant zeroing neurodynamic algorithm for upper limb motion intention-based human–robot interaction control in non-ideal conditions. *Expert Syst. Applic.* 213:118891. doi: 10.1016/j.eswa.2022.118891

Lukoševičius, M. (2012). *A Practical Guide to Applying Echo State Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 659–686.

Rezaee, B., and Zarandi, M. F. (2010). Data-driven fuzzy modeling for Takagi–Sugeno–Kang fuzzy system. *Inf. Sci.* 180, 241–255. doi: 10.1016/j.ins.2009.08.021

Ribeiro, V. H. A., Reynoso-Meza, G., and Siqueira, H. V. (2020). Multi-objective ensembles of echo state networks and extreme learning machines for stream?ow series forecasting. *Eng. Applic. Artif. Intell.* 95:103910. doi: 10.1016/j.engappai.2020.103910

Rodan, A., and Tino, P. (2011). Minimum complexity echo state network. *IEEE Trans. Neural Netw.* 22, 131–144. doi: 10.1109/TNN.2010.2089641

Rong, H.-J., Huang, G.-B., Sundararajan, N., and Saratchandran, P. (2009). Online sequential fuzzy extreme learning machine for function approximation and classification problems. *IEEE Trans. Syst. Man Cybern.* 39, 1067–1072. doi: 10.1109/TSMCB.2008.2010506

Shahid, A. A., Roveda, L., Piga, D., and Braghin, F. (2020). "Learning continuous control actions for robotic grasping with reinforcement learning," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 4066–4072. doi: 10.1109/SMC42975.2020.9282951

Su, D., Chen, L., Du, X., Liu, M., and and Jin, L. (2023a). Constructing convolutional neural network by utilizing nematode connectome: a brain-inspired method. *Appl. Soft Comput.* 149:110992. doi: 10.1016/j.asoc.2023.110992

Su, D., Stanimirovic, P. S., Han, L. B., and Jin, L. (2023b). Neural dynamics for improving optimiser in deep learning with noise considered. *CAAI Trans. Intell. Technol.* 9, 722–737. doi: 10.1049/cit2.12263

Sun, Z., Tang, S., Jin, L., Zhang, J., and Yu, J. (2023a). Nonconvex activation noise-suppressing neural network for time-varying quadratic programming: Application to omnidirectional mobile manipulator. *IEEE Trans. Industr. Inf.* 19, 10786–10798. doi: 10.1109/TII.2023.3241683

Sun, Z., Tang, S., Zhang, J., and Yu, J. (2023b). Nonconvex noise-tolerant neural model for repetitive motion of omnidirectional mobile manipulators. *IEEE/CAA J. Autom. Sinica* 10, 1766–1768. doi: 10.1109/JAS.2023.123273

Sun, Z., Tang, S., Zhou, Y., Yu, J., and Li, C. (2022a). A GNN for repetitive motion generation of four-wheel omnidirectional mobile manipulator with nonconvex bound constraints. *Inf. Sci.* 607, 537–552. doi: 10.1016/j.ins.2022.06.002

Sun, Z., Wang, G., Jin, L., Cheng, C., Zhang, B., and Yu, J. (2022b). Noise-suppressing zeroing neural network for online solving time-varying matrix square roots problems: a control-theoretic approach. *Exp. Syst. Applic.* 192:116272. doi: 10.1016/j.eswa.2021.116272

Sun, Z.-L., Au, K.-F., and Choi, T.-M. (2007). A neuro-fuzzy inference system through integration of fuzzy logic and extreme learning machines. *IEEE Trans. Syst. Man Cybern.* 37, 1321–1331. doi: 10.1109/TSMCB.2007.901375

Vargas, O., and S., De León Aldaco, S. E., Alquicira, J. A., Vela-Valdés, L. G., and Núñez, A. R. L. (2024). Adaptive network-based fuzzy inference system (ANFIS) applied to inverters: a survey. *IEEE Trans. Power Electr.* 39, 869–884. doi: 10.1109/TPEL.2023.3327014

Wang, H., Lee, I.-S., Braun, C., and Enck, P. (2016). Effect of probiotics on central nervous system functions in animals and humans: a systematic review. *J. Neurogastroenterol. Motil.* 22, 589–605. doi: 10.5056/jnm16018

Wei, L., and Jin, L. (2024). "Collaborative neural solution for time-varying nonconvex optimization with noise rejection," in *IEEE Transactions on Emerging Topics in Computational Intelligence*. doi: 10.1109/TETCI.2024.3369482

Yan, J., Jin, L., and Hu, B. (2024). "Data-driven model predictive control for redundant manipulators with unknown model," in *IEEE Transactions on Cybernetics*. doi: 10.1109/TCYB.2024.3408254

Yilmaz, B. M., Tatlicioglu, E., Savran, A., and Alci, M. (2022). Self-adjusting fuzzy logic based control of robot manipulators in task space. *IEEE Trans. Industr. Electr.* 69, 1620–1629. doi: 10.1109/TIE.2021.3063970

Yilmaz, B. M., Tatlicioglu, E., Savran, A., and Alci, M. (2023). Robust state/output-feedback control of robotic manipulators: an adaptive fuzzy-logic-based approach with self-organized membership functions. *IEEE Trans. Syst. Man Cybern.* 53, 3219–3229. doi: 10.1109/TSMC.2022.3224255

Yoo, B. K., and Ham, W. C. (2000). Adaptive control of robot manipulator using fuzzy compensator. *IEEE Trans. Fuzzy Syst.* 8, 186–199. doi: 10.1109/91.842152

Zhang, J., Jin, L., and Wang, Y. (2023). Collaborative control for multimanipulator systems with fuzzy neural networks. *IEEE Trans. Fuzzy Syst.* 31, 1305–1314. doi: 10.1109/TFUZZ.2022.3198855

Zhang, J., Jin, L., and Yang, C. (2022). Distributed cooperative kinematic control of multiple robotic manipulators with an improved communication efficiency. *IEEE/ASME Trans. Mechatr.* 27, 149–158. doi: 10.1109/TMECH.2021.3059441

Zhang, Y., Li, S., Kadry, S., and Liao, B. (2019). Recurrent neural network for kinematic control of redundant manipulators with periodic input disturbance and physical constraints. *IEEE Trans. Cybern.* 49, 4194–4205. doi: 10.1109/TCYB.2018.2859751

Zhao, W., Li, X., Chen, X., Su, X., and Tang, G. (2020). Bi-criteria acceleration level obstacle avoidance of redundant manipulator. *Front. Neurorobot.* 14:54. doi: 10.3389/fnbot.2020.00054

Zheng, E., Li, Y., Wang, Q., and Qiao, H. (2019). "Toward a human-machine interface based on electrical impedance tomography for robotic manipulator control," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2768–2774. doi: 10.1109/IROS40897.2019.8967872

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 06 August 2024
DOI 10.3389/fnbot.2024.1446508

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Lin Wang,
Chinese Academy of Sciences (CAS), China
Jiantao Yang,
University of Shanghai for Science and Technology, China

*CORRESPONDENCE
Zhongbo Sun
✉ zhongbosun2012@163.com

RECEIVED 10 June 2024
ACCEPTED 15 July 2024
PUBLISHED 06 August 2024

CITATION
Song F, Zhou Y, Xu C and Sun Z (2024) A novel discrete zeroing neural network for online solving time-varying nonlinear optimization problems. *Front. Neurorobot.* 18:1446508. doi: 10.3389/fnbot.2024.1446508

A novel discrete zeroing neural network for online solving time-varying nonlinear optimization problems

Feifan Song[1], Yanpeng Zhou[2], Changxian Xu[3] and Zhongbo Sun[4]*

[1]School of Finance, Changchun Finance College, Changchun, China, [2]VanJee Technology Co., Ltd., Beijing, China, [3]Department of Mechanical and Electrical Engineering, Changchun University of Technology, Changchun, China, [4]Department of Control Engineering, Changchun University of Technology, Changchun, China

To reduce transportation time, a discrete zeroing neural network (DZNN) method is proposed to solve the shortest path planning problem with a single starting point and a single target point. The shortest path planning problem is reformulated as an optimization problem, and a discrete nonlinear function related to the energy function is established so that the lowest-energy state corresponds to the optimal path solution. Theoretical analyzes demonstrate that the discrete ZNN model (DZNNM) exhibits zero stability, effectiveness, and real-time performance in handling time-varying nonlinear optimization problems (TVNOPs). Simulations with various parameters confirm the efficiency and real-time performance of the developed DZNNM for TVNOPs, indicating its suitability and superiority for solving the shortest path planning problem in real time.

KEYWORDS

path planning, discrete zeroing neural network, time-varying nonlinear optimization problem, 0-stability, real-time capability

1 Introduction

In recent years, the application of mobile platforms has been increasing, enhancing the efficiency of production systems (Balk et al., 2021; Wu et al., 2022; Zhou et al., 2022). In this case, certain collisions can delay production, harm productivity, and reduce profits (Gonzalez et al., 2016; Li et al., 2022). Therefore, the path planning problem for mobile platforms has become a research hotspot. In the processes of handling, loading, and unloading, the path planning problem for the mobile platform can be transformed into a shortest path planning problem, thereby saving both time and cost. Therefore, the shortest path problem is a typical combinatorial optimization problem that seeks the shortest path from a specified starting point to a desired terminus, aiming to minimize the total path cost (Zhang and Li, 2017; Li et al., 2021; Xu et al., 2022).

Path-planning methods are classified as follows: The artificial potential field method is a virtual force approach based on physical design (Jie et al., 2017; Zhou et al., 2023a,b). In this algorithm, movement toward the target point is likened to gravitation, while movement away from obstacles is likened to repulsion. Thus, the path planning problem is transformed into an optimization problem using a gravitational repulsion field function (Robinson et al., 2020). The model is simple to establish but challenging to obtain the optimal solution due to its tendency to converge to local optima. The fuzzy logic algorithm, derived from fuzzy control, emulates path-seeking methods based on drivers' daily driving

experiences. It directly utilizes expert knowledge from a database, offering good stability when incorporating real-time external information. However, the effectiveness of the fuzzy rules in the expert database relies heavily on accumulated experience, and the algorithm may lack real-time responsiveness in rapidly changing external environments. Graph search-based methods include the D* algorithm (Raheem and Ibrahim, 2018) and the Lee algorithm (Chi et al., 2022), etc. One of the most representative algorithms is the greedy algorithm, which aims to find the target point. In order to accelerate the optimization speed and avoid constraints, Fu improved the A* algorithm on the basis of the greedy algorithm to solve the path planning problem of industrial mobile manipulators. Under safe and non-collision conditions, a local path optimization strategy is directly adopted to reduce the number and length of local paths by straightening local paths (Fong et al., 2016; Fu et al., 2018). However, the algorithm lacks real-time performance due to the extensive computational requirements of the planning problems involving high-dimensional mobile platforms and snake-like robots. Bionic algorithms are developed for such problems, such as the genetic algorithm (Yang et al., 2008), the neural network algorithm (Qiu et al., 2018; Buddhadeb et al., 2020; Wang et al., 2022), and the ant colony algorithm (Song et al., 2021). The ant colony algorithm achieves optimization by simulating the foraging behavior of ant colonies, offering advantages such as parallelism and global optimization. Nevertheless, it is easy to fall into the local optimal solution due to the large number of calculations. Hui proposes an ant colony optimization algorithm to create a collapse-free incipient path in the intricate map and then applies a turning point optimization algorithm to achieve path planning on the mobile platform (Yang et al., 2019). Xu uses a particle swarm optimization algorithm to create a linear path and then smooths the linear path. However, vibrations can arise at the intersections of each path, potentially causing the anticipated trajectory to lose its optimality. This may result in the mobile platform stopping, rotating, and then restarting (Xu et al., 2021; Song et al., 2022). Therefore, a higher-order Bezier curve is utilized to construct the desired path directly to overcome the above problems. Nevertheless, it is necessary to design an efficient algorithm with strong computing power that is less time-consuming to find the optimal path in an environment with a large scanning area and a large number of obstacles. Hence, the above algorithms are always limited by the inherent problem of the exponential growth of the search scale. As the number of nodes increases, the success rate of solving within a limited time is significantly weakened. Neural networks are powerful algorithms to solve many scientific research and engineering problems. For the shortest path planning problem, Song proposed a pulse-coupled neural network model with a special mechanism to solve the shortest path planning problem. Compared with numerous algorithms, it effectively reduces costs (Sang et al., 2016). Filipe proposes a two-layer Hopfield neural network to solve the shortest path, which requires fewer neurons and converges quickly. However, the solution of this model is not optimal when dealing with shortest path planning problems (Araujo et al., 2001). In general, the discrete zeroing neural network (DZNN) has the characteristics of parallel processing, which can be used in path planning to quickly solve the optimal path and achieve the path planning task of the mobile platform (Hopfield and Tank, 1985).

The rest of this paper covers the following four parts: Section 2 describes the mathematical models of path planning and the ZNN model (DZNNM). Section 3 provides a theoretical analysis of the stability and convergence of the proposed DZNNM. In Section 4, the superiority and real-time characteristics of the proposed DZNNM are verified by numerical simulations. Section 5 draws the conclusion and future works. The primary contributions of this paper are described as follows:

1) The path planning problem is converted to the nonlinear optimization problem with equality and inequality constraints, and the nonlinear function related to the energy function is constructed so that the solution of the lowest energy state corresponds to the solution under the optimal path.
2) The theoretical analysis shows that the proposed discrete ZNN method has 0-stability and convergence for time-varying nonlinear optimization problems (TVNOPs).
3) The simulation results demonstrate that the DZNNM is feasible, effective, and real-time in dealing with the shortest path planning problem.

2 Problem formulation and model foundation

In this section, it describes the process of transforming the path planning problem into a nonlinear optimization problem. It covers the conversion process from the continuous ZNN model to the DZNNM and the establishment of the mathematical model for path planning. Specifically, it presents the problem formulation, the ZNN model, and the energy function model for online solving of TVNOPs.

2.1 Problem formulation

Let $L = \{j|j = 1, \cdots m\}$ denote an arbitrary finite set, and let $B = (j, r), (j \in L, r \in L)$ represent a set of ordered pairs of elements arranged sequentially. (j, r) and (r, j) represent different elements if and only if r is equal to j. $T = (L, D)$ is an oriented graph, and $D \subset B$. The parameters of L are named vertexes, and the parameters of D are denoted as oriented borders. If a cost matrix c_{jr} corresponds to edges in T from vertex j to vertex r, then T is referred to as a directed graph. Generally, the cost matrix c_{jr} is not needfully symmetric. In other words, the cost from vertex r to j is not inequivalent, possibly to the cost from vertex j to r. In addition, some borders between vertexes do not exist. Namely, the number of borders may be less than the quantity of vertices. For non-existent edges, the value of the cost coefficient is defined as infinity. In this paper, the shortest path problem is to search for the shortest feasible path from the desired starting point to the designated terminus.

2.2 Mathematical model of path planning

Consider the shortest path from vertex 1 to vertex m for an oriented graph with m vertexes and m borders, and the price of

c_{jr} per border. To formulate the shortest path problem, there are two typical path representations: vertex representation and edge representation. This paper adopts the border path characterization method to express the shortest path problem. The shortest path problem can be transformed into the following integer linear programming problem (Xia and Wang, 2000; Yoshihiko and Willsky, 2015).

$$\begin{aligned}&\min \sum_{j=1}^{m}\sum_{r=1}^{m} c_{jr}x_{jr}\\ &\text{s. t. } \sum_{v=1}^{m} x_{jv} - \sum_{l=1}^{m} x_{lj} = \begin{cases} 1, & \text{if } j = 1\\ 0, & \text{if } j = 2,3\cdots m-1\\ -1, & \text{if } j = m \end{cases},\end{aligned} \tag{1}$$

where the minimizing objective function of an integer linear programming problem (Equation 1) is the absolute price of the route, and the restriction is -1, 0, or 1. The constraint guarantees a sequential route from the specified starting point to the particular ending point. A decision variable represented by edge dependence from vertex to vertex is defined as follows:

$$x_{jr} = \begin{cases} 1, & \text{if the edge from } j \text{ to } r \text{ is on the path,}\\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Due to the constraint coefficient matrix defined in Equation 2, it is set to either 0 or 1. If there exists a unique optimal integer solution where the variable takes on values of 0 or 1, the integer programming mentioned above can be transformed into the following linear programming (Equation 3):

$$\begin{aligned}&\min \sum_{j=1}^{m}\sum_{r=1}^{m} c_{jr}x_{jr}\\ &\text{s. t. } \sum_{v=1}^{m} x_{jv} - \sum_{l=1}^{m} x_{lj} = \delta_{j1} - \delta_{jm}\\ &x_{jr} \geq 0, j, r = 1,2,\cdots m\end{aligned}, \tag{3}$$

where δ_{jr} denotes the Kronecker function, which is defined as $j = r$, $\delta_{jr} = 1$, and $j \neq r$, $\delta_{jr} = 0$. According to the duality principle of convex optimization (Lemeshko and Sterin, 2013), the dual path planning problem (Equation 4) can be obtained as follows:

$$\begin{aligned}&\max \ \ y_1 - y_m\\ &\text{s. t. } \ y_j - y_r \leq c_{jr} \ \ j, r = 1,2\cdots m\end{aligned}, \tag{4}$$

where y_j represents the dual decision variable related to vertex j, $y_1 - y_j$ indicates the shortest length from vertex 1 to vertex j.

Generally, a suitable energy function is developed such that the lowest energy condition corresponds to the anticipative solution. According to the duality properties of linear programming, an energy function model (Equation 5) of the original duality problem is generalized by Xia and Wang (2000):

$$\begin{aligned}E(x,y) =& \frac{1}{2}\sum_{j=1}^{m}\left[\sum_{r=1}^{m}(x_{jr} - x_{rj}) - \delta_{i1} + \delta_{im}\right]^2\\ &+ \frac{1}{2}\sum_{j=1}^{m}\sum_{r=1}^{m}\left[(-x_{jr})^{+}\right]^2\\ &+ \frac{1}{2}\left[\sum_{j=1}^{m}\sum_{r=1}^{m} c_{jr}x_{jr} - y_1 + y_m\right]^2\\ &+ \frac{1}{2}\sum_{j=1}^{m}\sum_{r=1}^{m}\left[(y_j - y_r - c_{jr})^{+}\right]^2,\end{aligned} \tag{5}$$

where $(s)^{+} = \max\{0, s\}$, and $s \in R$. The first term of the above formula represents the equality constraint, the second term denotes the non-negative constraint, the third term means the square dual gap, and the last term indicates the inequality restriction in the dual problem.

For convenience, the following coefficient vectors are defined as:

$$\begin{aligned}\hat{y} &= \left(\hat{y}_1, \cdots, \hat{y}_m\right)^{\top};\\ \hat{c} &= \left(\hat{c}_{11}, \cdots, \hat{c}_{1m}, \hat{c}_{21}, \cdots, \hat{c}_{2m}, \cdots, \hat{c}_{m1}, \cdots, \hat{c}_{mm}\right)^{\top};\\ x &= \left(x_{11}, \cdots, x_{1m}, x_{21}, \cdots, x_{2m}, \cdots, x_{m1}, \cdots, x_{mm}\right)^{\top}.\end{aligned}$$

Define A is an $m \times m^2$ constraint matrix, whose row denotes j and column means r, $e_j - e_r$ is a vector, the j element is 1, and the other elements are 0.

The above formula can be rewritten as Equation 6:

$$E(x,\hat{y}) = \frac{1}{2}\left[\left(c^{\top}x - (e_1 - e_m)^{\top}\hat{y}\right)^2 + \left\|(-x)^{+}\right\|_2^2 + \left\|\left(A^{\top}\hat{y} - c\right)^{+}\right\|_2^2 + \left\|Ax + e_m - e_1\right\|_2^2\right]. \tag{6}$$

Let $\tilde{b} = e_1 - e_m$. Therefore, the above equation can be simplified as Equation 7:

$$E(x,\hat{y}) = \frac{1}{2}\left[\left(c^{\top}x - \tilde{b}^{\top}\hat{y}\right)^2 + \left\|(-x)^{+}\right\|_2^2 + \left\|\left(A^{\top}\hat{y} - c\right)^{+}\right\|_2^2 + \left\|Ax - \tilde{b}\right\|_2^2\right], \tag{7}$$

where $\|\cdot\|$ represents the 2-norm, and given $(-x)^{+} = \left[(-x_1)^{+}, \cdots, (-x_m)^{+}\right]^{\top}$.

2.3 Continuous ZNN model and discrete ZNN model

Through the above analysis, the path planning problem is transformed into the path optimization problem from the specified initiating point to the terminus. An energy function is established for the shortest path to solve the optimization problem. The state solution corresponding to the optimal node is obtained when the energy function reaches its minimum. Therefore, the shortest path problem is considered as the TVNOP. The TVNOPs described in discrete time form are as follows (Guo et al., 2017; Qiu et al., 2018; Sun et al., 2021):

$$\min \hat{f}\left(x^{\chi+1}, t_{\chi+1}\right), \left[t_{\chi}, t_{\chi+1}\right) \in [0, +\infty), \tag{8}$$

where $\hat{f}(\cdot,\cdot)$ represents a differentiable nonlinear function (Equation 8). The discrete form of the TVNOPs is transformed

from the continuous time-varying nonlinear function $\hat{f}(x(t),t)$ based on the sampling time $t=(\chi+1)\tau$. $\tau>0$ is the acquisition interval, and $\chi=0,1,2,\cdots$ is the sampling time. The existing and foregone data is used to ensure the next data iteratively, which can solve TVNOPs. In the calculation time interval $[t_\chi,t_{\chi+1})\in[0,+\infty)$, the variable $x_{\chi+1}$ and the function $\hat{f}\left(x^{\chi+1},t_{\chi+1}\right)$ can be calculated iteratively by given information x_χ and $\hat{f}\left(x^{\chi},t_{\chi}\right)$ at the next moment.

A DZNNM is acquired for solving TVNOPs online; the following continuous TVNOPs (Equation 9) are considered:

$$\min_{x(t)\in R^n}\hat{f}(x(t),t)\in R,\ t\in[0,+\infty). \tag{9}$$

On behalf of solving the time-varying optimal solution of continuous TVNOPs $x^*(t)$, the gradient of the function $\hat{f}(x(t),t)$ (Equation 10) is directly generalized as:

$$\vartheta(x(t),t)=\partial\hat{f}(x(t),t)/\partial x(t). \tag{10}$$

The above formula is expanded to the following Equation 11:

$$\left[\frac{\partial\hat{f}}{\partial x_1},\frac{\partial\hat{f}}{\partial x_2},\cdots\frac{\partial\hat{f}}{\partial x_n}\right]^\top=[\vartheta_1(x(t),t),\vartheta_2(x(t),t),\cdots,\vartheta_n(x(t),t)]^\top\in R^n, \tag{11}$$

where the superscript $\top$ represents the transposition operational character of a matrix or a vector. The gradient $\vartheta(x(t),t)$ is a slippy differentiable nonlinear function created by the objective function $\hat{f}(x(t),t)$. On behalf of solving the theoretical solution of TVNOPs, the gradient of the objective function tends to 0, and the zeroing dynamical system (Equation 12) is defined as:

$$\dot{\vartheta}_t(x(t),t)=\frac{\mathrm{d}\vartheta(x(t),t)}{\mathrm{d}t}=-\lambda\vartheta(x(t),t), \tag{12}$$

where the parameter $\lambda>0$, $\dot{\vartheta}_t(x(t),t)$ is the derivative of the gradient $\vartheta(x(t),t)$ in connection with time. While the error $\vartheta(x(t),t)$ reaches 0, the solution $x(t)$ of the TVNOPs arrives at the corresponding theoretical solution $x^*(t)$ of the continuous TVNOPs (Sun et al., 2020a,b; Wei et al., 2021). Because of the zeroing dynamic system (Equation 12), the differential equation of the ZNN model (Equation 13) is extended as:

$$\tilde{H}(x(t),t)\dot{x}(t)=-\lambda\vartheta(x(t),t)-\dot{\vartheta}_t(x(t),t), \tag{13}$$

where $\tilde{H}(x(t),t)$ is a non-singular Hessian matrix. The details can be seen as follows:

$$\tilde{H}(x(t),t)=\frac{\partial^2\hat{f}(x(t),t)}{\partial x(t)\,\partial x^\top(t)}=\begin{bmatrix}\frac{\partial^2\hat{f}(x(t),t)}{\partial x_1\partial x_1}&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_1\partial x_2}&\cdots&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_1\partial x_n}\\ \frac{\partial^2\hat{f}(x(t),t)}{\partial x_2\partial x_1}&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_2\partial x_2}&\cdots&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_2\partial x_n}\\ \vdots&\vdots&\ddots&\vdots\\ \frac{\partial^2\hat{f}(x(t),t)}{\partial x_n\partial x_1}&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_n\partial x_2}&\cdots&\frac{\partial^2\hat{f}(x(t),t)}{\partial x_n\partial x_n}\end{bmatrix}\in R^{n\times n}.$$

Due to the non-singularity of the Hessian matrix, the above equation is converted to the following Equation 14:

$$\dot{x}(t)=-\tilde{H}^{-1}(x(t),t)\left(\lambda\vartheta(x(t),t)+\dot{\vartheta}_t(x(t),t)\right). \tag{14}$$

If the Hessian matrix is a positive symmetric matrix, it represents the solution of the continuous TVNOPs. Moreover, if the matrix is singular, the Hessian matrix can be transformed into $\tilde{H}+rI$, where r is the absolute value of the maximum eigenvalue of the Hessian matrix and I is the identity matrix. Thus, the matrix $\tilde{H}(x(t),t)$ satisfies the non-singularity condition.

A DZNNM is proposed to solve the TVNOPs to solve the optimal value of the energy function $E(x,\hat{y})$. Hence, a continuous ZNN model is discretized to obtain the ZNN model in discrete form. Generally, Euler's forward difference equation $\dot{x}(t)=\left(x^{\chi+1}-x^{\chi}\right)/\tau$ is employed to discretize the continuous ZNN model (Equation 15) as follows:

$$x^{\chi+1}=x^{\chi}-\tilde{H}^{-1}\left(x^{\chi},t_\chi\right)\left(\iota\vartheta\left(x^{\chi},t_\chi\right)+\tau\dot{\vartheta}_t\left(x^{\chi},t_\chi\right)\right). \tag{15}$$

The above formula can be called as DZNNM, where $\iota=\tau\lambda\in(0,1]$ is step length, $\tilde{H}^{-1}\left(x^{\chi},t_\chi\right)$, $\vartheta\left(x_\chi,t_\chi\right)$, and $\dot{\vartheta}_t\left(x_\chi,t_\chi\right)$ are discrete forms of $\tilde{H}^{-1}(x(t),t)$, $\vartheta(x(t),t)$, and $\dot{\vartheta}_t(x(t),t)$, respectively.

3 Theoretical analyzes and results

The continuous ZNN model construction process and the DZNNM construction process are briefly described to solve the TVNOPs in the previous section. Therefore, the continuous ZNN model-building process, discretization steps, and proof process are elaborated on in this section. Define a continuously differentiable linear equation, and the matrix $Q(t)$ is a known and bounded matrix of time-varying full-rank coefficients; $w(t)$ is a time-varying vector and is differentiable at any time in connection with time t.

$$\nu(x(t),t)=Q(t)x(t)-w(t)=0. \tag{16}$$

The discrete formal equation corresponding to the above formula (Equation 16) can be managed in the time period $[\chi\tau,(\chi+1)\tau]\subseteq[t_0,t_f]$. The time-varying equation in discrete form Equation 17 is as follows:

$$Q_{\chi+1}x_{\chi+1}=w_{\chi+1}, \tag{17}$$

where the matrices $Q_{\chi+1}$ and $w_{\chi+1}$ are discrete forms of the matrices $Q(t)$ and $w(t)$, respectively. Instantaneous sampling is $t=(\chi+1)\tau$, i.e., $\chi=0,1,2\cdots$ denotes the regenerative target.

The following vector-valued error function $\nu(x(t),t)=Q(t)x(t)-w(t)$ is defined to handle the above equation based on the design steps by Zhang et al. (2015). The continuous ZNN model (Equation 18) of the linear equation dynamical system has the following form:

$$\dot{x}(t)=Q^{-1}(t)(\dot{w}(t)-\dot{Q}(t)x(t)-\eta(Q(t)x(t)-w(t))). \tag{18}$$

Among them is the design parameter $\eta>0$, which can be used to control the convergent ratio. $Q^{-1}(t)$ denotes inverse and it is equivalent to $H^{-1}(t)$. Generally, the ZNN model is discretized by Euler's forward difference formula (Equation 19) as follows:

$$x_{\chi+1}=x_\chi+Q_\chi^{-1}(\tau\dot{w}_\chi-\tau\dot{Q}_\chi x_\chi-h(Q_\chi x_\chi-w_\chi)). \tag{19}$$

Definition 1 (Guo and Zhang, 2012; Jin and Zhang, 2015; Zhang et al., 2015). The roots of the characteristic polynomial

$P_M(\psi) = \sum_{i=0}^{M} \omega_i \psi^i$ are used to verify whether the M-step method $\sum_{i=0}^{M} \omega_i \sigma_{\chi+i} = \tau \sum_{i=0}^{M} \zeta_i \varpi_{\chi+i}$ has 0-stability .

The M-step method has 0-stability if the solution of the equation $p_M(\psi) = 0$ lies on or within the unit circle (i.e., $|\psi| \leq 1$). The convergence order $O(\tau^p)$ of the M-step method matches the truncation error order p ($p > 0$) of the equation solution.

Definition 2 (Jin and Zhang, 2015; Jin et al., 2018). If and only if M-step has 0-stability and is consistent over time $t \in [t_0, t_f]$, it is convergent (i.e., $\sigma_{[(t-t_0)/\tau]} \to \sigma^*(t)$ with $\tau \to 0$) .

Definition 3. The 0-stable consistency of the M-step method converges to the order of its truncation error.

Based on the aforementioned definitions, we will analyze the 0-stability and convergence performance of the DZNNM.

Theorem 1. The DZNNM is 0-stable.

Proof. A DZNNM (Equation 15) is viewed as the one-step neural network dynamics on account of Definition 1. According to Definition 1, the characteristic polynomial (Equation 20) of the ZNN model separated and dispersed by forward Euler interpolation is as follows:

$$P_1(\psi) = \psi - 1. \tag{20}$$

The root (Equation 21) of the above equation is

$$\psi_1 = 1. \tag{21}$$

Therefore, according to Definition 1, the DZNNM is 0-stable. The proof is fulfilled.

Theorem 2. The DZNNM (Equation 15) converges to the order of the truncation error $O(\tau^2)$.

Proof. The forward Euler interpolation formula (Equation 22) is as follows:

$$\dot{x}_\chi = \frac{x_{\chi+1} - x_\chi}{\tau} + O(\tau). \tag{22}$$

The continuous ZNN model (Equation 18) is discretized by forward Euler interpolation, and the following formula (Equation 23) is obtained:

$$x_{\chi+1} = x_\chi + Q_\chi^{-1}(\tau \dot{w}_\chi - \tau \dot{Q}_\chi x_\chi - h(Q_k x_\chi - w_\chi)) + O(\tau^2). \tag{23}$$

In the light of the above analysis, the truncation error of the DZNNM is $O(\tau^2)$, so the DZNNM has consistency, convergence, and 0-stability. According to Definition 2 and Definition 3, the order of convergence of the model is $O(\tau^2)$. The proof is fulfilled.

Theorem 3. For the TVNOPs in discrete form, the steady-state position error $\lim_{\chi \to \infty} \|Q_\chi x_\chi - w_\chi\|_2$ of the DZNNM has order $O(\tau^2)$.

Proof. According to Theorem 1, Theorem 2, and Definition 3, as χ tends to infinity, we can get $x_\chi^* + O(\tau^2) = x_\chi$. Therefore, the following derivation process (Equation 24) is obtained:

$$\begin{aligned} \|Q_\chi x_\chi - w_\chi\|_F &= \|Q_k(x^* + O(\tau^2)) - w_\chi\|_F \\ &= \|Q_\chi x^* - w_\chi + Q_\chi O(\tau^2))\|_F, \end{aligned} \tag{24}$$

where $\|\|_F$ is a Fubini norm. The following Equation (25) is obtained by further arrangement:

$$\|Q_\chi x_\chi - w_\chi\|_F = \|Q_\chi O(\tau^2))\|_F = O(\tau^2). \tag{25}$$

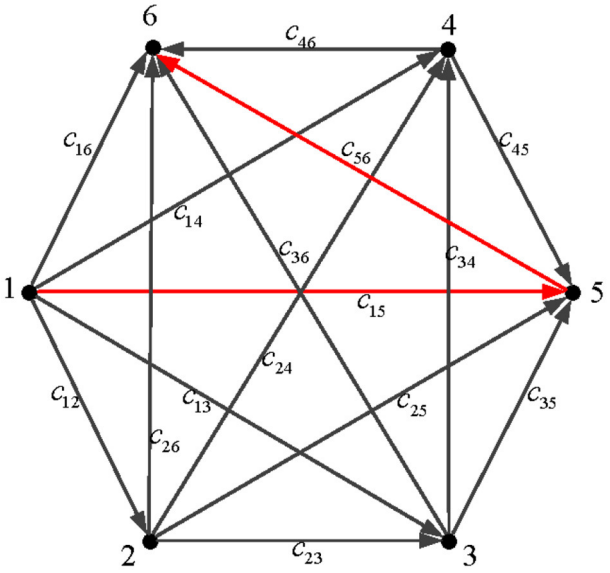


FIGURE 1
The transport diagram of six-node path planning.

TABLE 1 Cost coefficients of six-node path planning problems.

	c_1	c_2	c_3	c_4	c_5	c_6
c_1	100	16	8	16	9	8
c_2	100	100	9	10	11	9
c_3	100	100	100	10	11	12
c_4	100	100	100	100	9	10
c_5	100	100	100	100	100	13
c_6	100	100	100	100	100	100

This proof is thus completed.

4 Numerical simulations and verifications

Consider the shortest path planning problem, where each node has five possible directions to move from a fixed initial point to the terminus. To make the transportation process more reasonable, the following conditions are assumed to be true:

1) For the path planning problem with a single starting point and a single target point, node 1 is the starting point and node 6 is the endpoint, as shown in Figure 1.
2) In order to meet the actual transportation situation, some transportation roads do not exist. For example, it cannot travel from node 4 to node 4. Therefore, the given value of the cost c_{jr} of such a path is large in the simulation.
3) In the transportation process, it should not go in the reverse direction. For instance, there is no arrow to go from node 2 to node 1, indicating that the situation is not considered.
4) Assume that each node can go to a node whose number is greater than its own.

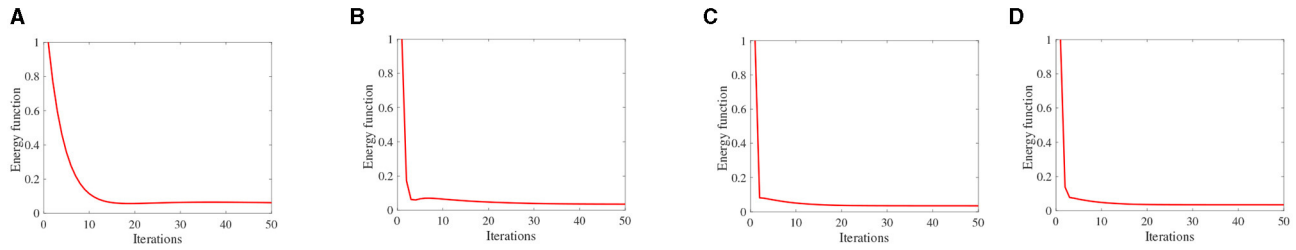


FIGURE 2
Energy function based on the six-node primal duality problem, **(A)** $\iota = 0.1$, **(B)** $\iota = 0.5$, **(C)** $\iota = 0.9$, and **(D)** $\iota = 1.0$.

TABLE 2 **Energy loss from node 1 to other nodes.**

$\|E_{12}\|$	$\|E_{13}\|$	$\|E_{14}\|$	$\|E_{15}\|$	$\|E_{16}\|$
5.5558	2.1937	1.6309	1.3792	2.5478

TABLE 3 **Energy loss from node 5 to other nodes.**

$\|E_{51}\|$	$\|E_{52}\|$	$\|E_{53}\|$	$\|E_{54}\|$	$\|E_{56}\|$
15.3249	20.9741	8.3363	3.4891	2.9722

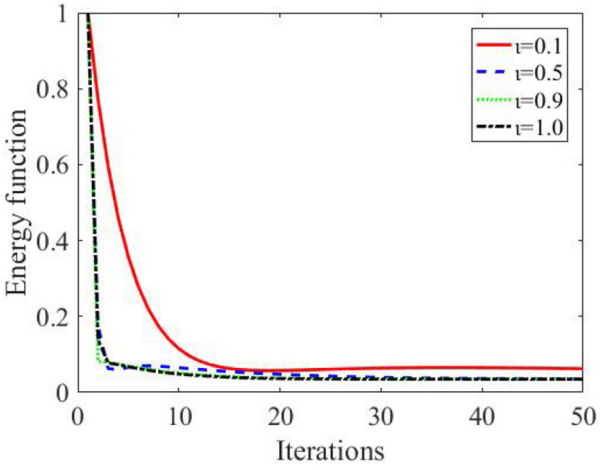


FIGURE 3
Energy loss under different parameters.

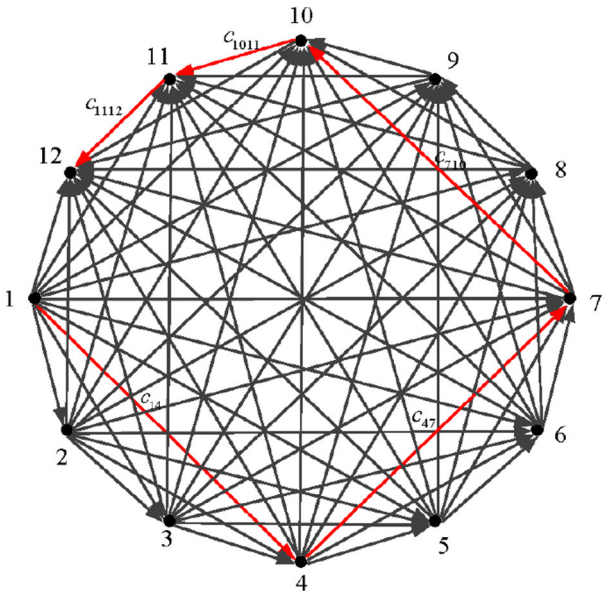


FIGURE 4
Twelve-node mobile platform path transportation diagram.

4.1 Six-node path planning simulations

The path planning problem is to find the shortest path between source node 1 and terminal node 6, so as to minimize the cost $E(x, y)$ in the transportation process. The cost coefficient matrix $\hat{c}$ needs to be set in the process of establishing the path planning mathematical model in Section 2.2, which is given in Table 1. The cost coefficient matrix is $c = [c_{11}, c_{12}, \cdots c_{16}, \cdots, c_{61}, \cdots c_{66}]^{\top}$, and the original value matrix of the system is defined as follows: $x(0) = [0, 0, \cdots 0]^{\top}_{36*1}$, $y(0) = [1, 1, \cdots 1]^{\top}_{6*1}$. In Section 2.4, it is noted that different parameters of the DZNNM generally exhibit different convergence rates. Therefore, the parameters are set as $\iota = 0.1$, $\iota = 0.5$, $\iota = 0.9$, and $\iota = 1.0$, respectively. The energy function of each path is calculated successively to determine the shortest path in the transportation process.

The simulation results show that the DZNNM is exploited to solve the shortest path planning problem. As the number of iterations increases, the energy function $E(x, y)$ decreases to 0, indicating that the DZNNM can effectively address the path planning problem with a single starting point and a single target point. While the energy function gets its minimum value, the optimal solution $\hat{x}$ can be solved at this time. The optimal solution $\hat{x}$ is substituted into the energy function $E(x, y)$ to obtain the cost of each path so as to determine the shortest path. Starting from node 1, it uses the energy function to calculate the energy consumption from node 1 to node 2, node 3, node 4, node 5, and node 6. The specific energy loss is shown in the Tables 2, 3.

As can be seen in Table 2, it can be concluded that the cost from node 1 to node 5 is the smallest. Therefore, the energy consumption of node 5 compared to other nodes is calculated. According to the analysis of the actual transportation situation and assumed conditions, when the mobile platform moves to node 5, it can only transport objects to target point 6. In order to verify the

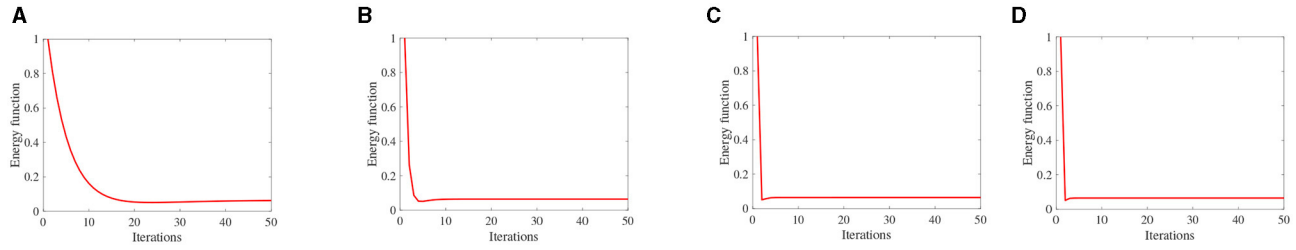


FIGURE 5
Energy function based on the 12-node primal duality problem, **(A)** $\iota = 0.1$, **(B)** $\iota = 0.5$, **(C)** $\iota = 0.9$, and **(D)** $\iota = 1.0$.

TABLE 4 Cost coefficients of six-node path planning problems.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}
c_1	100	5	4	2	9	8	10	12	25	30	22	23
c_2	100	100	9	10	11	9	9	10	12	15	15	15
c_3	100	100	100	10	11	12	12	8	10	12	15	13
c_4	100	100	100	100	9	10	2	8	12	13	14	13
c_5	100	100	100	100	100	13	12	9	11	12	14	16
c_6	100	100	100	100	100	100	14	13	15	17	18	15
c_7	100	100	100	100	100	100	100	15	14	3	13	19
c_8	100	100	100	100	100	100	100	100	15	17	18	15
c_9	100	100	100	100	100	100	100	100	100	14	15	16
c_{10}	100	100	100	100	100	100	100	100	100	100	16	3
c_{11}	100	100	100	100	100	100	100	100	100	100	100	17
c_{12}	100	100	100	100	100	100	100	100	100	100	100	100

effectiveness of the algorithm, the optimal solution is substituted into the expression of the energy function to solve the energy loss from node 5 to each node. Table 3 can testify to the validity of the DZNNM. Therefore, the shortest path is from node 1 to node 5, and finally to target point 6. Figure 2, Tables 2, 3 indicate that the DZNNM is effective in processing TVNOPs. As the number of iterations increases, the energy function decreases to 0 in a short number of times, which reflects the high efficiency and real-time performance of the DZNNM.

As the parameter ι increases, the energy function rapidly converges to 0, reflecting the fast convergence and effectiveness of the DZNNM, as shown in Figure 3. In practical application, adjusting parameters can accelerate the convergence rate of the whole optimal path, which can quickly accelerate and complete the path planning.

TABLE 5 The energy loss from node 1 to each other.

$\|E_{11}\|$	$\|E_{12}\|$	$\|E_{13}\|$	$\|E_{14}\|$	$\|E_{15}\|$	$\|E_{16}\|$
∞	0.0364	0.0256	0.0113	0.0369	0.0265
$\|E_{17}\|$	$\|E_{18}\|$	$\|E_{19}\|$	$\|E_{110}\|$	$\|E_{111}\|$	$\|E_{112}\|$
0.0217	0.0136	0.0204	0.0645	0.3339	2.1625

TABLE 6 The energy loss from node 4 to each other.

$\|E_{45}\|$	$\|E_{46}\|$	$\|E_{47}\|$	$\|E_{48}\|$
0.0121	0.9318	0.0097	0.0125
$\|E_{49}\|$	$\|E_{410}\|$	$\|E_{411}\|$	$\|E_{412}\|$
1.1156	0.1319	0.0218	1.2085

4.2 Path planning simulations of 12 nodes

To demonstrate the correctness of the energy function mathematical model as well as the validity and real-time capability of the DZNNM, the 12-node path planning problem is further discussed. The transport diagram for the problem is shown in Figure 4. For the sake of comparison, the assumptions of this problem are the same as those of the six-node path planning problem.

For convenience, the initial matrix is defined as follows:

$x(0) = [0, 0, \cdots 0]^{\top}_{144*1}$,

$y(0) = [1, 1, \cdots 1]^{\top}_{12*1}$,

$c = [c_{11}, c_{12}, \cdots c_{112}, c_{21}, c_{22}, \cdots c_{212}, \cdots, c_{121}, \cdots c_{1212}]^{\top}$.

The simulation results of solving the shortest path problem with 12 nodes using the DZNNM are as follows: The simulations show

TABLE 7 Energy loss from node 7 to each other.

$\|E_{78}\|$	$\|E_{79}\|$	$\|E_{710}\|$	$\|E_{711}\|$	$\|E_{712}\|$
0.0218	0.05424	0.0023	0.0719	1.7784

TABLE 8 Energy loss from node 10 to each other.

$\|E_{1011}\|$	$\|E_{1012}\|$	$\|E_{1112}\|$
0.0443	0.2757	1.5876

that the path planning problem is increased to 12 nodes, and the DZNNM can effectively solve the discrete TVNOPs, as shown in Table 4. It can be seen from Figure 5 that the addition of path nodes does not influence the convergence rate of the proposed DZNNM.

It can reflect the correctness of the path-planning mathematical model as well as the superiority and real-time performance of the DZNNM. In addition, the values in Table 5 show that the energy loss from node 1 to any other node, so it can be concluded that the energy consumption from node 1 to node 4 is the smallest in the path planning process. Figure 5 and Table 5 demonstrate that the DZNNM exhibits convergence performance, 0-stability, and superior capability in handling TVNOPs.

Combined with the data in Table 6, the second path consumes the least energy to move from node 4 to node 7. The data in Table 7 show that the optimal choice in the third path is to move from node 7 to node 10, and the energy consumed is 0.0023. Table 8 shows the energy loss of the last two sections of the path. In Table 8, the minimum energy consumption from node 10 to node 11 is reflected by numerical values. Meanwhile, the value of energy consumption from node 11 to target point 12 is given as 1.5876. Combined Table 5 with Table 8, it can be concluded that the motion path in the twelve-node path planning problem is $1 \rightarrow 4 \rightarrow 7 \rightarrow 10 \rightarrow 11 \rightarrow 12$. The proposed DZNNM is suitable and effective for discrete TVNOPs. In addition, the convergence rate does not decrease with the increase of the nodes in the path-planning problems, and the convergence rate can be accelerated by scaling the design parameters appropriately. These characteristics make the DZNNM suitable for solving large-scale path-planning problems in real-time applications.

5 Conclusion and future work

A DZNNM is developed and analyzed to handle the shortest path planning problem from a single starting point to a single terminus. For the shortest path planning problem, a discrete nonlinear function related to the energy function is constructed so that the solution of the lowest energy function corresponds to the solution of the shortest path. The shortest path planning problem is transformed into the TVNOPs through strictly mathematical analysis. In addition, the convergence, 0-stability, and theoretical results of the proposed DZNNM are discussed and analyzed, which reflect that the DZNNM can effectively deal with the shortest path-planning problems. Simulation results show that the proposed DZNNM has high precision and real-time performance in dealing with path planning problems. Ultimately, the future research direction is to develop mathematical models under complex conditions and solve multi-starting point and multi-objective point path planning problem.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/supplementary material.

Author contributions

FS: Data curation, Formal analysis, Methodology, Writing — original draft, Writing — review & editing. YZ: Data curation, Methodology, Formal analysis, Writing — original draft. CX: Formal analysis, Methodology, Writing — original draft, Writing — review & editing. ZS: Funding acquisition, Methodology, Supervision, Writing — review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. The work was supported in part by the National Natural Science Foundation of China under grant numbers 62373065, 61873304, 62173048, and 62106023, and also in part by the Key Science and Technology Projects of Jilin Province, China, under grant number YDZJ202402015CXJD, and also in part by the Changchun Financial Specialist Platform Project under grant number 2024JZ003.

Acknowledgments

The authors would like to thank the reviewers and the technical editor for their valuable comments and suggestions on revising this paper.

Conflict of interest

YZ was employed by VanJee Technology Co., Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Araujo, F., Ribeiro, B., and Rodrigues, L. (2001). A neural network for shortest path computation. *IEEE Trans. Neural Netw.* 12, 1067–1073. doi: 10.1109/72.950136

Balk, B. M., Koster, M. B., and Kaps, C. (2021). An evaluation of cross-efficiency methods: with an application to warehouse performance. *Appl. Math. Comput.* 406:126261. doi: 10.1016/j.amc.2021.126261

Buddhadeb, P., Arijit, N., Nirmal, B., and Diptendu, S. (2020). A novel hybrid neural network-based multirobot path planning with motion coordination. *IEEE Trans. Vehicul. Technol.* 69, 1319–1327. doi: 10.1109/TVT.2019.2958197

Chi, W., Ding, Z., Wang, J., Chen, G., and L, S. (2022). A generalized voronoi diagram-based efficient heuristic path planning method for RRTS in mobile robots. *IEEE Trans. Indus. Electr.* 62, 5429–5436. doi: 10.1109/TIE.2021.3078390

Fong, S., Deb, S., and Chaudhary, A. (2016). A review of metaheuristics in robotics. *Comput. Electr. Eng.* 43, 278–291. doi: 10.1016/j.compeleceng.2015.01.009

Fu, B., Chen, L., Zhou, Y., and Zheng, D. (2018). An improved a* algorithm for the industrial robot path planning with high success rate and short length. *Robot. Auton. Syst.* 106, 26–37. doi: 10.1016/j.robot.2018.04.007

Gonzalez, J., Perez, V., and Milanes, F. (2016). A review of motion planning techniques for automated vehicles. *IEEE Trans. Intell. Transport. Syst.* 17, 1135–1145. doi: 10.1109/TITS.2015.2498841

Guo, D., Nie, Z., and Yan, L. (2017). Novel discrete-time Zhang neural network for time-varying matrix inversion. *IEEE Trans. Syst. Man Cybernet.* 47, 2301–2310. doi: 10.1109/TSMC.2017.2656941

Guo, D., and Zhang, Y. (2012). Zhang neural network, Getz—Marsden dynamic system, and discrete-time algorithms for time-varying matrix inversion with application to robots' kinematic control. *Neurocomputing* 97, 22–32. doi: 10.1016/j.neucom.2012.05.012

Hopfield, J., and Tank, D. (1985). Neural computation of decisions in optimization problems. *Biol. Cybernet.* 52, 141–152. doi: 10.1007/BF00339943

Jie, J., Khajepour, A., Melek, W., and Huang, Y. (2017). Path planning and tracking for vehicle collision avoidance based on model predictive control with multi constraints. *IEEE Trans. Vehicul. Technol.* 66, 952–964. doi: 10.1109/TVT.2016.2555853

Jin, L., and Zhang, Y. (2015). Discrete-time Zhang neural network for online time-varying nonlinear optimization with application to manipulator motion generation. *IEEE Trans. Neural Netw. Learn. Syst.* 26, 1525–1531. doi: 10.1109/TNNLS.2014.2342260

Jin, L., Zhang, Y., and Qiu, B. (2018). Neural network-based discrete-time Z-type model of high accuracy in noisy environments for solving dynamic system of linear equations. *Neural Comput. Appl.* 29, 1217–1232. doi: 10.1007/s00521-016-2640-x

Lemeshko, O. V., and Sterin, V. L. (2013). "Structural and functional optimization of transport telecommunication network," in *2013 23rd International Crimean Conference Microwave Telecommunication Technology* (Sevastopol: Microwave & Telecommunication Technology), 490–491.

Li, W., Liu, K., Li, C., Sun, Z., Liu, S., and Gu, J. (2022). Development and evaluation of a wearable lower limb rehabilitation robot. *J. Bionic Eng.* 19, 688–699. doi: 10.1007/s42235-022-00172-6

Li, W., Liu, K., Sun, Z., Li, C., Chai, Y., and Gu, J. (2021). A neural network-based model for lower limb continuous estimation against the disturbance of uncertainty. *Biomed. Sign. Process. Contr.* 71:103115. doi: 10.1016/j.bspc.2021.103115

Qiu, B., Zhang, Y., and Yang, Z. (2018). New discrete-time ZNN models for least-squares solution of dynamic linear equation system with time-varying rank-deficient coefficient. *IEEE Trans. Neural Netw. Learn. Syst.* 29, 5767–5776. doi: 10.1109/TNNLS.2018.2805810

Raheem, F. A., and Ibrahim, U. (2018). Path planning algorithm using D* heuristic method based on PSO in dynamic environment. *Am. Acad. Sci. Res. J. Eng. Technol. Sci.* 49, 257–271.

Robinson, D., Sanders, D., and Mazharsolook, E. (2020). Development of a* algorithm for robot path planning based on modified probabilistic roadmap and artificial potential field. *J. Eng. Sci. Technol.* 15, 3034–3054.

Sang, Y. S., Lv, J. C., Qu, H., and Yi, Z. (2016). Shortest path computation using pulse-coupled neural networks with restricted autowave. *Knowl. Bas. Syst.* 114, 1–11. doi: 10.1016/j.knosys.2016.08.027

Song, B., Miao, H., and Xu, L. (2021). Path planning for coal mine robot via improved ant colony optimization algorithm. *Syst. Sci. Contr. Eng.* 9, 283–289. doi: 10.1080/21642583.2021.1901158

Song, B., Wang, Z., and Zou, L. (2022). A new approach to smooth path planning of mobile robot based on quartic Bezier transition curve and improved PSO algorithm. *Neurocomputing* 473, 98–106. doi: 10.1016/j.neucom.2022.01.056

Sun, Z., Li, F., Jin, L., Shi, T., and Liu, K. (2020a). Noise-tolerant neural algorithm for online solving time-varying full rank matrix Moore-Penrose inverse problems: a control-theoretic approach. *Neurocomputing* 413, 158–172. doi: 10.1016/j.neucom.2020.06.050

Sun, Z., nd L. Wei, Y. L., and Liu, K. (2020b). Two DTZNN models of $O(\tau^4)$ pattern for online solving dynamic system of linear equations: application to manipulator motion generation. *IEEE Access* 99, 36624–36638. doi: 10.1109/ACCESS.2020.2975223

Sun, Z., Shi, T., Jin, L., Pang, Z., and Yu, J. (2021). Discrete-time zeroing neural network of $o(\tau^4)$ pattern for online solving time-varying nonlinear optimization problem: application to manipulator motion generation. *J. Franklin Inst.* 358, 7203–7220. doi: 10.1016/j.jfranklin.2021.07.006

Wang, J., Liu, J., Chen, W., and Chi, W. (2022). Robot path planning via neural-network-driven prediction. *IEEE Trans. Artif. Intell.* 3, 451–460. doi: 10.1109/TAI.2021.3119890

Wei, L., Jin, L., Yang, C., Chen, K., and Li, W. (2021). New noise-tolerant neural algorithms for future dynamic nonlinear optimization with estimation on hessian matrix inversion. *IEEE Trans. Syst. Man Cybernet.* 51, 2611–2623. doi: 10.1109/TSMC.2019.2916892

Wu, K., Wang, H., MahdiAi, E., and Yuan, S. (2022). Achieving real-time path planning in unknown environments through deep neural networks. *IEEE Trans. Intell. Transport. Syst.* 23, 2093–2102. doi: 10.1109/TITS.2020.3031962

Xia, Y., and Wang, J. (2000). A discrete-time recurrent neural network for shortest-path routing. *IEEE Trans. Automat. Contr.* 45, 2129–2134. doi: 10.1109/9.887639

Xu, L., Song, B., and Cao, M. (2021). A new approach to optimal smooth path planning of mobile robots with continuous-curvature constraint. *Syst. Sci. Contr. Eng.* 9, 138–149. doi: 10.1080/21642583.2021.1880985

Xu, T., Zhu, X., and Qi, W. H. (2022). Passive analysis and finite-time anti-disturbance control for Semi-Markovian Jump Fuzzy systems with saturation and uncertainty. *Appl. Math. Comput.* 424:127030. doi: 10.1016/j.amc.2022.127030

Yang, H., Qi, J., Miao, Y., and Sun, H. (2008). A point symmetry-based clustering technique for automatic evolution of clusters. *IEEE Trans. Knowl. Data Eng.* 20, 1441–1457. doi: 10.1109/TKDE.2008.79

Yang, H., Qi, J., Miao, Y., and Sun, H. (2019). A new robot navigation algorithm based on a double-layer ant algorithm and trajectory optimization. *IEEE Trans. Indus. Electr.* 66, 8557–8566. doi: 10.1109/TIE.2018.2886798

Yoshihiko, A., and Willsky, A. (2015). *Signals and Systems*. Tokyo: Wiley Telecom.

Zhang, Y., Jin, L., Guo, D., Yin, Y., and Chou, Y. (2015). Taylor-type 1-step-ahead numerical differentiation rule for first-order derivative approximation and ZNN discretization. *J. Comput. Appl. Math.* 273, 29–40. doi: 10.1016/j.cam.2014.05.027

Zhang, Y., and Li, S. (2017). Distributed biased min-consensus with applications to shortest path planning. *IEEE Trans. Automat. Contr.* 62, 5429–5436. doi: 10.1109/TAC.2017.2694547

Zhou, M., Li, T., Zhang, C., Yu, Y., Zhang, X., and Su, C. (2023a). Sliding mode iterative learning control with iteration-dependent parameter learning mechanism for nonlinear systems and its application. *IEEE Trans. Automat. Sci. Eng.* 2023, 1–12. doi: 10.1109/TASE.2023.3336933

Zhou, M., Zhang, Y., Wang, Y., Yu, Y., Su, L., Zhang, X., and Su, C. (2023b). Data-driven adaptive control with hopfield neural network-based estimator for Piezo-actuated stage with unknown hysteresis input. *IEEE Trans. Instr. Measur.* 72, 1–11. doi: 10.1109/TIM.2023.3325870

Zhou, X. Y., Tian, Y., and Wang, H. P. (2022). Neural network state observer-based robust adaptive fault-tolerant quantized iterative learning control for the rigid-flexible coupled robotic systems with unknown time delays. *Appl. Math. Comput.* 430:127286. doi: 10.1016/j.amc.2022.127286

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 16 August 2024
DOI 10.3389/fnbot.2024.1430155

OPEN ACCESS

EDITED BY
Xin Ma,
The Chinese University of Hong Kong, China

REVIEWED BY
Puchen Zhu,
The Chinese University of Hong Kong, China
Xiaoyin Zheng,
XMotors.ai, United States

*CORRESPONDENCE
Touseef Sadiq
touseef.sadiq@uia.no
Hameedur Rahman
hameed.rahman@mail.au.edu.pk

RECEIVED 09 May 2024
ACCEPTED 31 July 2024
PUBLISHED 16 August 2024

CITATION
Hanzla M, Yusuf MO, Al Mudawi N, Sadiq T, Almujally NA, Rahman H, Alazeb A and Algarni A (2024) Vehicle recognition pipeline via DeepSort on aerial image datasets. *Front. Neurorobot.* 18:1430155. doi: 10.3389/fnbot.2024.1430155

Vehicle recognition pipeline via DeepSort on aerial image datasets

Muhammad Hanzla[1], Muhammad Ovais Yusuf[1], Naif Al Mudawi[2], Touseef Sadiq[3]*, Nouf Abdullah Almujally[4], Hameedur Rahman[1]*, Abdulwahab Alazeb[2] and Asaad Algarni[5]

[1]Faculty of Computing and AI, Air University, Islamabad, Pakistan, [2]Department of Computer Science, College of Computer Science and Information System, Najran University, Najran, Saudi Arabia, [3]Department of Information and Communication Technology, Centre for Artificial Intelligence Research, University of Agder, Grimstad, Norway, [4]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia, [5]Department of Computer Sciences, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia

Introduction: Unmanned aerial vehicles (UAVs) are widely used in various computer vision applications, especially in intelligent traffic monitoring, as they are agile and simplify operations while boosting efficiency. However, automating these procedures is still a significant challenge due to the difficulty of extracting foreground (vehicle) information from complex traffic scenes.

Methods: This paper presents a unique method for autonomous vehicle surveillance that uses FCM to segment aerial images. YOLOv8, which is known for its ability to detect tiny objects, is then used to detect vehicles. Additionally, a system that utilizes ORB features is employed to support vehicle recognition, assignment, and recovery across picture frames. Vehicle tracking is accomplished using DeepSORT, which elegantly combines Kalman filtering with deep learning to achieve precise results.

Results: Our proposed model demonstrates remarkable performance in vehicle identification and tracking with precision of 0.86 and 0.84 on the VEDAI and SRTID datasets, respectively, for vehicle detection.

Discussion: For vehicle tracking, the model achieves accuracies of 0.89 and 0.85 on the VEDAI and SRTID datasets, respectively.

KEYWORDS

deep learning, remote sensing, object recognition, unmanned aerial vehicles, DeepSort, dynamic environments, path planning

1 Introduction

Rapid economic and demographic expansion generate a dramatic surge in vehicle numbers on highways. Hence, complete road traffic monitoring is necessary for acquiring and evaluating data, essential for comprehending highway operations within an intelligent, autonomous transportation framework (Dikbayir and İbrahim Bülbül, 2020; Xu et al., 2022; Yin et al., 2022). Consequently, there's a compelling need to automate traffic monitoring systems. While various image-based solutions have been developed, obstacles exist in expanding their capabilities, especially in dynamic contexts where backdrop and objects are in flux (Weng et al., 2006; Di et al., 2023; Dai et al., 2024). Traditional approaches like background removal and frame differencing struggle when used to photographs acquired from mobile platforms owing to background motion, blurring the boundaries between background and foreground objects. Hence, improvements in computer vision and image processing, covering disciplines such as intelligent transportation, medical imaging, object

identification, semantic segmentation, and human-computer interaction, present promising paths (Angel et al., 2003; Cao et al., 2021; Ding et al., 2024).

Semantic segmentation, defining and identifying pixels by class, provides a sophisticated method (Schreuder et al., 2003; Sun et al., 2020; Ren et al., 2024). Unlike current systems confined to binary segmentation (e.g., vehicle vs. backdrop), our suggested technique utilizes multi-class segmentation, expanding scene knowledge (Ding et al., 2021; Gu et al., 2024). Moreover, utilizing aerial data offers enormous promise in boosting traffic management. However, obstacles such as varying item sizes, wide non-road regions, and different road layouts need efficient solutions to exploit mobile platform-derived data effectively (Najiya and Archana, 2018; Sun et al., 2018; Omar et al., 2021).

In this study, a unique approach for the identification and tracking of vehicles is based on aerial images. In our approach, aerial films are first transformed into frames for images (Sun et al., 2023). Defogging and gamma correction methods are then used for noise reduction and bright-ness improvement, respectively, while pre-processing is being done on those frames (Qu et al., 2022; Chen et al., 2023a; Zhao X. et al., 2024). After that, Fuzzy C Mean and DBSCAN algorithm is used for segmentation to decrease the background complexity. YOLOv8 is applied for recognition of automobiles in each extracted frame as it can detect tiny objects successfully. To track several cars inside the image's frames, all identified vehicles have been allocated an ID based on ORB attributes. Also, to estimate the traffic density on the roadways, a vehicle count has been kept throughout the picture frames. The tracking has been done using the DeepSort with Kalman filter. Moreover, the provided traffic monitoring systems were verified by the tests done on VEDAI, and SRTID datasets. The studies have exhibited amazing detection and tracking precision when compared to other state-of-the-art (SOTA) approaches.

Some of the prominent contributions of this work include:

- Our model reduces model complexity by combining pre-processing methodologies with segmentation techniques for the preparation of frames prior to detection phase.
- Evaluation of unsupervised segmentation strategies, specifically Fuzzy C-Mean (FCM) algorithm and Density-Based Spatial Clustering of Applications with Noise (DBSCAN), was undertaken, boosting segmentation effectiveness.
- Significantly enhanced accuracy, recall, and F1 Score in vehicle recognition and tracking compared to earlier techniques have been obtained.

Implementation of vehicle tracking leveraging the DeepSort algorithm, reinforced by an ID assignment and recovery module based on ORB, has been successfully accomplished, exhibiting remarkable performance proven across two publicly available datasets.

The article is structured into the following sections: Section 2 dives into the literature on traffic analysis. Section 3 goes over the proposed technique in great depth. Section 4 describes the experimental setting, offering empirical insights into the system's performance. Section 5 reviews the system's performance and considers its advantages and disadvantages. Section 6. Discuss the work's limitations. Section 7 is the conclusion, which summarizes the main results and proposes future research and development goals.

2 Literature review

Over the last several years, academics have aggressively excavated into constructing traffic monitoring systems. They have examined the behaviors of their systems utilizing multiple picture sources, including static camera feeds, satellite images, and aerial data (Li J. et al., 2023; Wu et al., 2023). Typically, the full photos undergo first preprocessing to exclude non-essential components beyond cars, followed by feature extraction (Hou et al., 2023a). Different strategies depend on techniques such as image differencing, foreground extraction, or background removal, especially when the Region of Interest (ROI) is well-defined and suitably sized within the images (Shi et al., 2023; Zhu et al., 2024). Aerial imaging can cause the size of vehicles to vary based on the height of image acquisition. Because of this, semantic segmentation techniques have become popular for detection and tracking applications. It is also common to use additional stages such as clustering and identifier assignment to improve results. Deep learning algorithms have become popular in recent years for object recognition, showing better performance in handling complex situations (Wang et al., 2024; Yang et al., 2024). To provide an overview of current models and approaches, the linked research is classified into machine-learning and deep learning-based traffic system analyses.

2.1 Machine learning-based traffic scene analysis

Machine learning has been extensively used in computer vision-related jobs for a long time, particularly in traffic control and monitoring. To find the cars in the images (Rafique et al., 2023), introduced a vehicle recognition model based on haar-like characteristics with an AdaBoost classifier. In Drouyer and de Franchis (2019), a method for monitoring traffic on highways using medium resolution satellite images is shown. The backdrop image difference approach was used to identify the items in motion after a median filter was applied to the images after road masking for the elimination of irrelevant regions. Next, the gray level of the resultant image was computed. The last phase used a thresholding strategy to identify large, bright spots as autos. According to the authors in Hinz et al. (2006), motion detection algorithms are in-effective because aerial images include motion in both the foreground and background. Therefore, an approach based on morphological operations, the Otsu partitioning method, and bottom-hat and top-hat transformations was applied for detection. After extracting the Shi Tomasi features, clusters were formed based on displacement and angle trajectories. The automobiles vanished behind the backdrop clusters. Each vehicle's robust feature vector was used for tracking. To achieve excellent precision, they used several feature maps. Vehicle detection has been accomplished utilizing two distinct methods in separate research (Chen and Meng, 2016). While the other approach employed HSV color characteristics in conjunction with the Gray Level Co-occurrence Matrix (GLCM) to identify cars, the first methodology used features from the Accelerated Segment Test (FAST) and HOG features. Vehicle tracking is achieved via the use of Forward and Backward Tracking (FBT).

The background subtraction approach is used by Aqel et al. (2017) to identify moving automobiles. Morphological adjustments are carried out to reduce the incidence of false positives. Ultimately, the

invariant Charlier moments are used to achieve categorization. The method's applicability to a variety of traffic circumstances is limited by the usage of standard image processing methods. Additionally, the automobiles that are not moving will be removed using the background subtraction approach, which will lower the true positives. Another traffic monitoring strategy has been provided by Mu et al. (2016). The model selected the area with a high Absolute Difference (SAD) value as a moving vehicle after computing the image difference. Vehicles have been found and matched across many picture frames using SIFT. The authors of Teutsch et al. (2017) used a novel technique for stacking images. The image registration process was limited to tiny autos, and the warping approach was used to blur any stationary backdrops close to moving vehicles. The main goal of this algorithm is to remove distracting backdrop features from images so that only the vehicle is visible when the surrounding region is smoothed out. These systems have a high temporal complexity, and these approaches were distinguished by their complicated properties. These methods incur high computational costs. Furthermore, the model's generalizability is weakened as scene complexity rises.

2.2 Deep learning-based traffic scene analysis

Traffic monitoring has always included manual techniques and in-car technology. Nonetheless, deep learning is more effective than traditional methods when it comes to image processing. An automobile recognition method based on the You Look Only Once (YOLOv4) deep learning algorithm has been presented by Lin and Jhang (2022). Another study Bewley et al. (2016) employed the Faster R-CNN as the target detector and developed a tracking method (SORT) for real-time systems based on the Hungarian matching algorithm and the Kalman filter to track several targets at once. The SORT tracker does not take the monitored object's appearance characteristics into account. A technique for detecting automobiles using an enhanced YOLOv3 algorithm is proposed by Zhang and Zhu (2019). To increase the detection method's accuracy, a new structure is added to the pre-trained YOLO network during training. YOLOv3, on the other hand, is among the most ancient. Using the most recent designs may enhance the detection result. Miniature CNN architecture, as described by Ozturk and Cavus (2021), is a vehicle identification model that relies on Convolutional Neural Networks (CNNs) in conjunction with morphological adjustments. The computational cost of this post-processing is high. Moreover, different aerial image databases show different levels of accuracy. A method for real-time object tracking and detection was reported by the authors in Alotaibi et al. (2022). An enhanced RefineDet-based detection module is included in the model. Additionally, the twin support vector machine model and the harmony search technique are employed for classification. Pre-processing of the data is absent from the model, which might lower the model's total computing complexity. A vehicle identification model based on deep learning is shown in Amna et al. (2020). Convolutional Neural Networks (CNN) are used by the model to recognize vehicles, while radar data is used to determine the target's location. A two-stage deep learning model is developed in different research (He and Li, 2019). In addition to detecting cars, the model also recognizes them again in subsequent frames, which is a crucial component of tracking. As opposed to traditional appearance and motion-based characteristics, the re-identification is mostly reliant on vehicle tracking context information.

There is always room for development in the field of automated traffic monitoring systems, despite the substantial research that has been done in this area. To get effective results, efficient and specialized designs are needed for the recognition of automobiles in aerial images, particularly in situations with heavy traffic. Machine learning techniques are insufficient to distinguish between objects whose pixels exhibit motion. As a result, we use deep learning strategies to raise the precision of vehicle tracking and detection.

3 Materials and methods

3.1 System methodology

This section details the planned traffic monitoring system. System architecture overview is provided in Figure 1. This work offers a vehicle recognition and tracking system based on semantic segmentation. Firstly, the videos are turned into frames and pre-processing processes, i.e., defogging for noise reduction are done to the images. Then Gamma correction is employed for adjustment of image intensity for enhanced detection. FCM and DBSCAN segmentation was done on the filtered images for separation of foreground and back-ground items. YOLOv8 is applied for vehicle detection. ORB attributes are used for the assignment of unique ID. Vehicles were traced over several frames of images using a Deepsort. For finding each tracked vehicle, ORB key point description combined with trajectories approximation are used to recover IDs. Further information on each module is given in the ensuing subsections.

3.2 Images pre-processing

To eliminate superfluous pixel information from the resulting image, noise reduction is necessary since the extra pixel's complicate recognition (Rong et al., 2022; Xiao et al., 2023). For best performance, any filter using defogging methods is applied to noise (Gao et al., 2020; Tang et al., 2024). The defogging technique measures the amount of noise in each pixel of the picture and then removes it in the following ways.

$$G(x) = X(x)Y(x) + Z(1 - p(x))$$

where pixel location is denoted by x, fog density by Z, and transmission map by $Y(x)$. Figure 2 represents defogged images:

The denoised image's intensity is then adjusted using gamma correction (Huang et al., 2018; Zhao L. et al., 2024) since a high brightness allows for the most effective detection of the area of interest. The gamma correction power-law is provided as follows:

$$Vo = TV_I^{\gamma}$$

where V_I is the non-negative value with power γ of the input, which may vary from 0 to 1, and T is a constant, usually equals to 1. V_o stands for the final image. The plotted denoised, intensity adjusted. Figure 3 shows the gamma-corrected images.

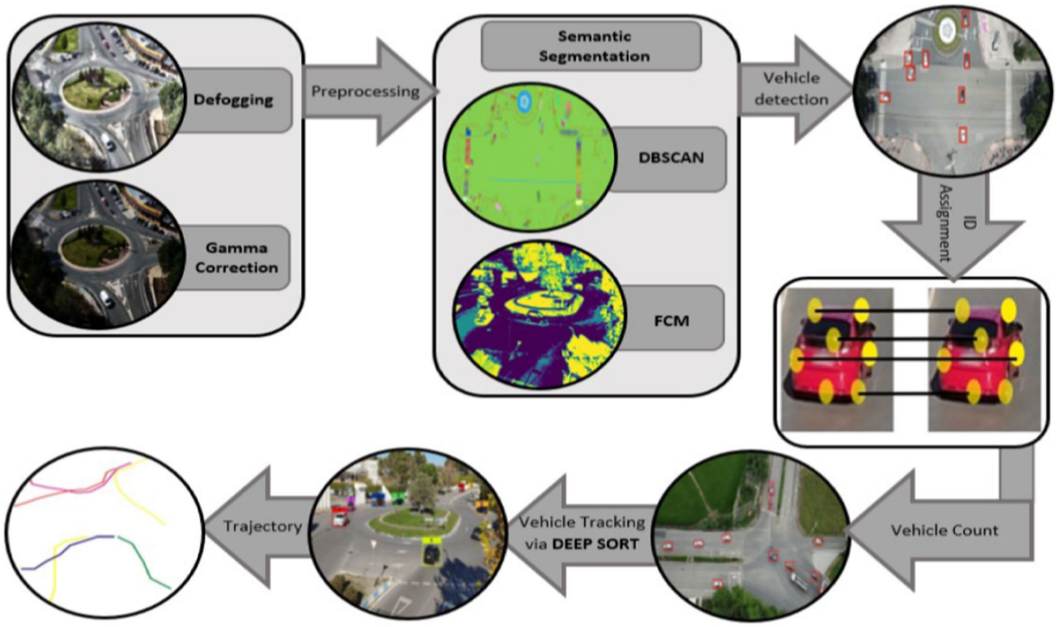


FIGURE 1
Flowchart demonstrating the proposed traffic surveillance system proposed system architecture.

FIGURE 2
Defogging results over the **(A)** original image of VEDAI dataset **(B)** defogged image **(C)** original image of SRTID dataset **(D)** defogged image.

3.3 Semantic segmentation

In many computer vision applications, including autonomous vehicles, medical imaging, virtual reality, and surveillance systems, image segmentation is essential. Images are divided into homogeneous sections using segmentation methods. Every area stands for a class or object. To improve item recognition on complicated backdrops, we compared two segmentation techniques.

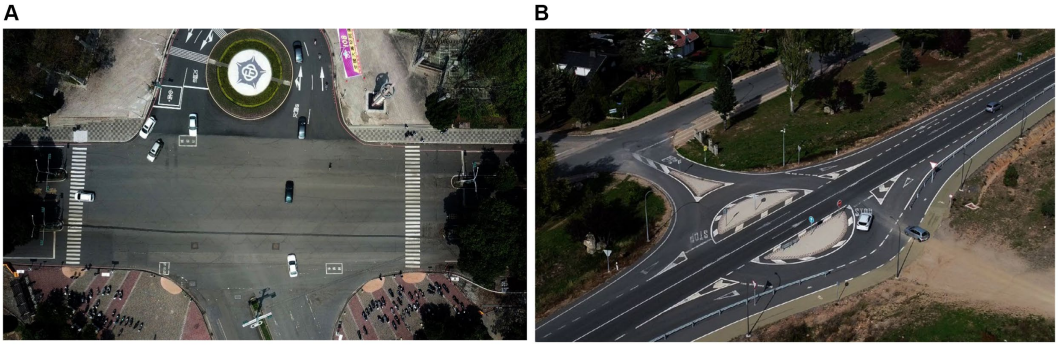


FIGURE 3
Pre-processed image using gamma correction over the **(A)** VEDAI dataset **(B)** SRTID dataset.

3.3.1 FCM segmentation

Segmentation is widely employed in a variety of computer vision applications. This is a fundamental stage. Segmentation methods separate images into homogeneous sections (Huang et al., 2019; Hao et al., 2024). Each area denotes an item or class. We used the Fuzzy C-Mean segmentation technique. FCM is a clustering method in which each picture pixel might belong to two or more groups. Fuzzy logic (Chong et al., 2023; Zheng et al., 2024) refers to pixels that belong to more than one cluster. Because we are working with many complicated road backdrops including several items and circumstances, segmentation approaches based on explicit feature extraction and training are unable to deliver a generic solution. For this purpose, we used FCM, a non-supervised clustering algorithm. During the FCM segmentation process, the objective function is optimized across numerous rounds. Throughout the iterations, the clustering centers and membership degrees were continually updated (Rehman and Hussain, 2018). The FCM method separates a finite collection of N items (S=$s1$, $s2$, sn) into C clusters. Each component of vi (i = 1, 2…, N) is a vector of d dimensions. We design a technique to divide s into C clusters using cluster centers $u1$, $u2$, and so on in the centroid set u (Xiao et al., 2024; Xuemin et al., 2024). The FCM approach uses a representative matrix (g) to represent the membership of each element in each cluster. The matrix g may be defined using equation:

$$g(i,z), 1 \le i \le N; 1 \le z \le C$$

where $g\ (i,\ z)$ represents the membership value of the element si having cluster center vz. While calculating performance index J_{fcm}, and it is used to calculate the weighted sum of the distance between cluster center and components of the associated fuzzy cluster.

$$Jfcm = (g,u) = \sum_{i=1}^{M}\sum_{Z=1}^{N} g_{iz}^{b} \left\| s_i - v_z \right\|^2, 1 < b < \infty$$

where m indicates the number of clusters, N signifies the number of pixels, si is the ith pixel, vz is the zth cluster center, and b represents the blur exponent. The degree of membership function must meet the conditions specified in the equation below.

$$0 \le b_{iz} \le 1 \sum_{z=1}^{r} b_{iz} = 1, \sum_{i=1}^{M} b_{iz} \le M, i = 1,2,..M \text{ and } z = 1,2,..,r$$

Each time the membership function matrix is updated using equation:

$$b_{iz}^{b} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{iz}^2}{d_{jz}^2} \right)^{\frac{2}{b-1}}}$$

The membership matrix (b_{iz}^{b}) is between [0,1], and the distance between cluster centroid (vi) and pixel (sz) is supplied by $\left(d_{iz}^2\right)$. The cluster centroid is determined by equation:

$$v_j = \frac{\sum_{z=1}^{N} g_{iz}^{b} s_z}{\sum_{z=1}^{N} g_{iz}^{b}}$$

A pixel receives a high membership value as it gets closer to the belonging cluster center and vice versa. Figure 4 depicts the results of the FCM segmentation.

3.3.2 Density-based spatial clustering (DBSCAN)

DBSCAN, or density-based spatial clustering, is a popular method in machine learning and data analysis (Khan et al., 2014; Deng et al., 2022). In contrast to conventional clustering techniques that need preset cluster numbers, DBSCAN utilizes a data-centric methodology. It uses data density and closeness to its advantage to detect variable-sized and irregularly formed clusters within complicated datasets (Bhattacharjee and Mitra, 2020; Liu et al., 2023). Initially, core points are determined based on having the fewest surrounding data points within a certain distance. These core locations are then expanded into clusters by adding nearby data points that satisfy density requirements (Chen et al., 2022; Zhang et al., 2023). Noise is defined as any data point that does not fit into a designated cluster or core point.

$$N_{\varepsilon}(x_i) = \left\{ x_j \in X | dist(x_i, x_j) \le \varepsilon \right\}$$

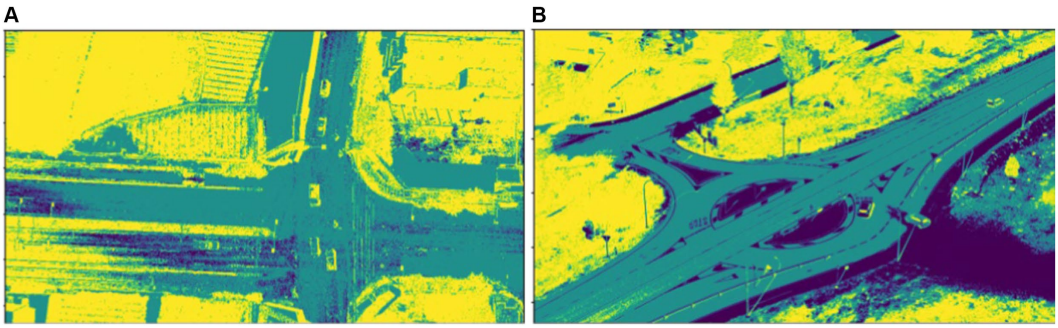


FIGURE 4
Segmentation using FCM over **(A)** VEDAI dataset and **(B)** SRTID dataset.

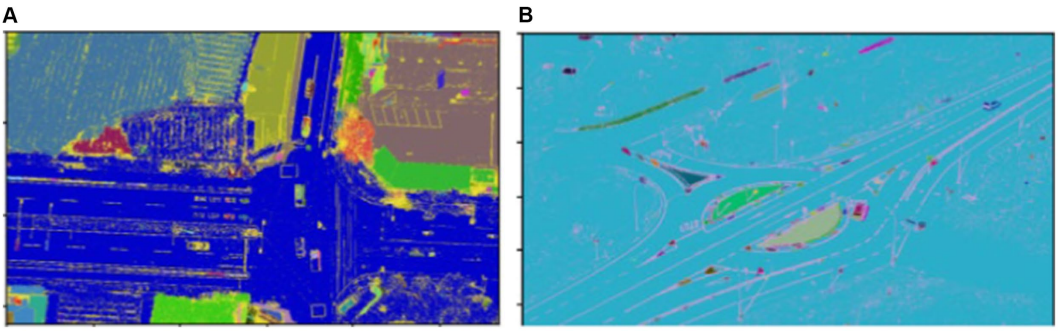


FIGURE 5
Segmentation using DBSCAN over **(A)** VEDAI dataset and **(B)** SRTID dataset.

where $N_\varepsilon(x_i)$ represent the neighborhood of a point x_i, $x_j \in X$ denotes all points x_j belonging to the dataset X, $dist(x_i, x_j)$ calculates the distance between points, ε is a threshold distance parameter, defining the maximum distance for points to be considered neighbors (see Figure 5).

$$C = \{ x_i \in X \mid\mid N_\varepsilon(x_i) \geq MinPts \}$$

$$x_i \in X \mid\mid N_\varepsilon(x_j) \text{ and } x_j \in C, X = \{ x_1, x_2, \ldots, x_n \}.$$

where x_i is the epsilon neighborhood of x_j and x_j is the core point.

The FCM and DBSCAN segmentation methods were evaluated in terms of computational cost and error rates determined using equations.

$$\text{Error Rate} = 1 - \text{accuracy}$$

FCM surpasses DBSCAN owing to its adeptness in managing datasets with varied cluster shapes and sizes. By adding fuzzy membership degrees, FCM addresses the ambiguity inherent in data point assignments, resulting in more adaptive and improved clustering. Furthermore, FCM enables increased control over cluster boundaries via parameterization, allowing for exact alterations to better fit the specific properties of the data. Table 1 exhibits FCM's better efficacy and accuracy in picture segmentation on VEDAI and SRTID datasets. Considering both computation time and error rates, FCM shines, making its findings the preferable option for following tasks such as vehicle recognition, ID allocation, recovery, counting, and tracking.

3.4 Vehicle detection

YOLOv8 is utilized for vehicle recognition and radiates as an excellent single-shot detector capable of identifying, segmenting, and classifying with fewer training parameters (Chen et al., 2023b; Wang et al., 2023). According to the CSP principle, the C2f module replaces the C3 module to align with the YOLOv8 backbone, increasing gradient flow information while keeping YOLOv5 compliant. The C2f module combines C3 with ELAN in a unique manner, drawing on YOLOv7's ELAN methodology, ensuring YOLOv8 compatibility (). The SPPF module at the backbone's end employs three consecutive 5 × 5 Maxpools before concatenation in each layer to reliably identify objects of varied sizes with lightweight efficiency (Sun et al., 2019; Li S. et al., 2023; Yi et al., 2024).

YOLOv8 integrates PAN-FPN in its neck portion, which improves feature fusion and data use at different sizes (Mostofa et al., 2020; Xu et al., 2020). The neck module combines a final decoupled head structure, many C2f modules, and two up samplings (Song et al., 2022; Wu and Dong, 2023). YOLOv8's neck is like YOLOx's head idea,

which combines confidence and regression boxes to increase accuracy. It operates as an anchor-free model, detecting the object center directly, lowering box predictions, and speeding up the Non-Maximum Suppression (NMS) process, an important post-processing step (Li et al., 2024). Figure 6 shows automobiles spotted using YOLOv8.

3.5 ID allocation and recovery based on ORB features

Prior to tracking each identified vehicle in the subsequent image frames, an ID based on ORB traits was assigned to each detected vehicle. A quick and effective feature detector is ORB (Chien et al., 2016; Chen et al., 2022). FAST (Features from Accelerated Segment Test) key point detector is used for key-point detection. It is a more sophisticated version of the BRIEF (Binary Robust Independent Elementary Features) description. It is also rotationally and scale-invariant. Equation is used to get a patch moment (Luo et al., 2024; Yao et al., 2024).

$$n_{st} = \sum x^s y^t I(u,v)$$

where x and y are the image pixels' relative intensities, represented by the values s and t. These moments may be utilized to find the center of mass using equation:

$$N = \frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}$$

where the equation defines path orientation:

$$\theta = atan\left(m_{01}, m_{10}\right)$$

The identified cars in the subsequent frames were compared using the extracted ORB features, and if a match was discovered, the ID was restored; if not, the vehicle was recorded in the system with a new ID (Cai et al., 2024). ID is restored across frames and ORB feature description is applied to the extracted cars; results are shown in Figure 7.

3.6 Vehicle counting

Using YOLOv8's vehicle detections, we incorporated vehicle counts in every image frame to conduct a thorough analysis of the traffic situation (Tian et al., 2022; Yang et al., 2023). Using a counter, each seen vehicle was painstakingly recorded under equation. Road traffic density at different times may be measured by counting the number of cars within each frame (Minh et al., 2023). This data is essential for enabling quick responses to unforeseen events like traffic jams or other circumstances that might impair traffic flow (Wu et al., 2019; Peng et al., 2023).

$$\text{Vehicle Count} = \sum_{i=1}^{N} T$$

where, T denotes the vehicle detections within a single frame, with the corresponding output visualized in Figure 8.

3.7 Vehicle tracking

We utilized the DeepSORT tracker to observe the movements of vehicles frame by frame. DeepSORT is a tracking approach that makes use of deep learning characteristics with the Kalman filter to track objects based on their appearance, motion, and velocity (Bin Zuraimi and Kamaru Zaman, 2021; Sun G. et al., 2022). Using the Mahalanobis

TABLE 1 Error rate comparison of DBSCAN and FCM.

Datasets	Error rate	
	DBSCAN	FCM
VEDAI	0.32	0.20
SRTID	0.37	0.23

FIGURE 6
Vehicle Detection over **(A)** VEDAI and **(B)** SRTID datasets marked with red boxes via the YOLOv8 algorithm.

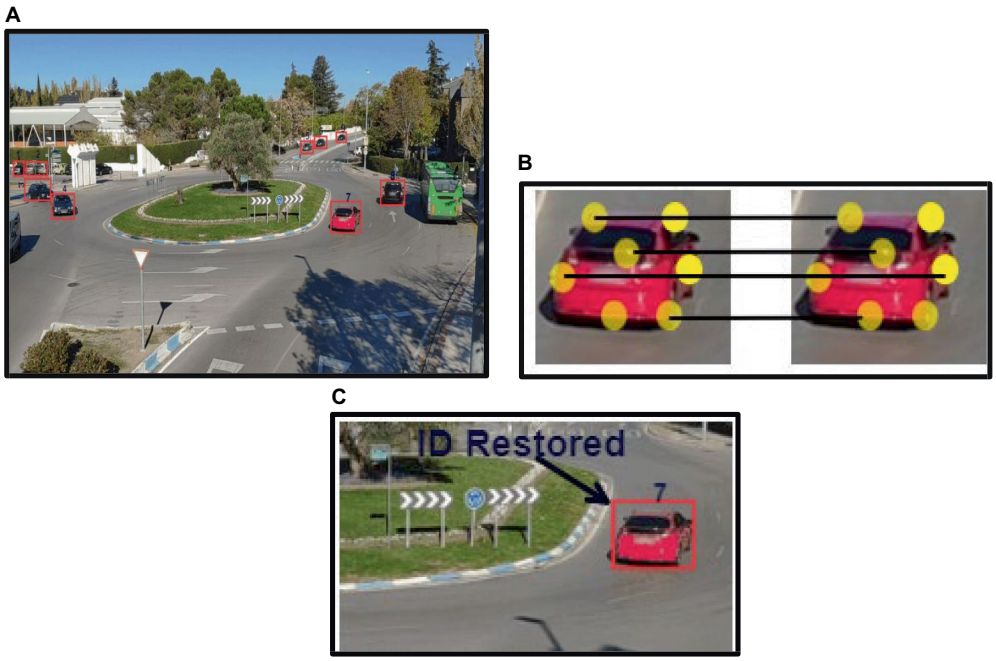


FIGURE 7
ID assignment and restoration: **(A)** ID assigned to each vehicle based on ORB features; **(B)** features matching across frames; **(C)** ID restored for the same vehicle in succeeding frame.

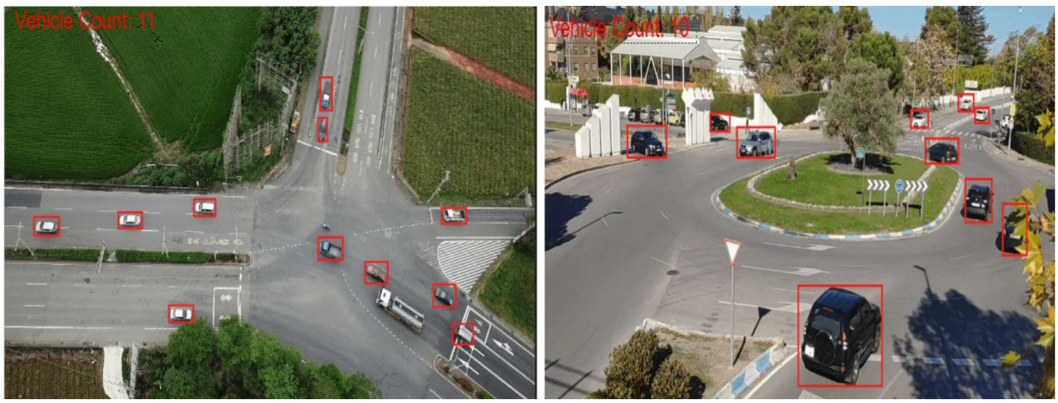


FIGURE 8
Density estimation by using vehicle count displayed at the left corner of each image.

distance metric between the Kalman state and the freshly obtained measurement, (Li et al., 2018; Sun Y. et al., 2022) the motion information is merged as described in equation:

$$k^{(1)}(i,j) = (k_j - v_i)^{\mathrm{T}} S_i^{-1} (k_j - v_i)$$

where k_j is the jth bounding box detection and (v_i, S_i) is the ith track distribution projection into space measurement. The appearance information has been computed using the smallest cosine distance, as provided by equation, between the ith and jth detections in appearance space.

$$k^{(2)}(i,j) = \min\left\{1 - t_j^{\mathrm{T}} t_k^{(i)} \mid r_k^{(i)} \in \mathcal{R}_i\right\}$$

where t_j and $t_k^{(i)}$ represent the appearance and associated appearance descriptor, respectively. The extracted appearance and motion information is combined as given in equation:

ALGORITHM 1 Trajectory estimation of tracked vehicles

Input:
: V= $\{D^1, D^2, \ldots, D^z\}$
// rectangular coordinates of length $\{x_1, x_2, y_1, y_2\}$
Output: $\{P^r\}_{r=1}^{R}$
$R = \{R^1, R^2, \ldots, R^z\}$ ←DeepSORT(V)
feature vector= []
For i = 1 to the length of R
new_feature←ORB(R^i)
If feature vector== []
feature_vector←new_feature
Else
matches← (new_feature, *feature vector)*
If matches>8
AssignID (R^i)
Else
feature_vector←new_feature
$x_1^i,\ x_2^i,\ y_1^i,\ y_2^i$ ←ExtractRectangularCoordinatesofVehicle ()

$$x_{center} \leftarrow \frac{(x_1^i + x_2^i)}{2}$$

$$y_{center} \leftarrow \frac{(y_1^i + y_2^i)}{2}$$

$R^r \leftarrow [x_{center}, y_{center}]$
end for
Return vehicle trajectories

$$c_{i,j} = \lambda k^{(1)}(i,j) + (1-\lambda)k^{(2)}(i,j)$$

where c is the corresponding weight. The appearance features are produced by a pre-trained CNN model that contains two convolution layers, six residual layers linked to a dense layer, one max pooling layer, and l2 normalization (Kumar et al., 2023; Mi et al., 2023). The DeepSORT algorithm's tracking mechanism is shown in Figure 9 (Singh et al., 2023). In Figure 10, the tracking result is shown.

3.8 Vehicle trajectory estimation

In addition to the previously computed density, we approximated the path traveled by each tracked vehicle. The trajectories taken by a vehicle may be utilized to construct vehicle detection (Adi et al., 2018; Bozcan and Kayacan, 2020). It may also be used to identify trajectory conflicts and accidents if it is further developed. The route is plotted if the vehicle is tracked (Chen and Wu, 2016; Wang et al., 2022). To approximate the trajectories, we used geometric coordinates from observed rectangular boxes. DeepSORT was used for location estimation and coordinate retrieval (Leitloff et al., 2014; Sheng et al., 2024). The center points of estimated locations, which represent individual vehicle IDs, were noted on a separate image, and then linked to construct trajectories.

The approach feeds detection coordinates into the DeepSORT tracker, which predicts vehicle placements in the following frame. Vehicle IDs are retrieved using ORB features; if the number of matches exceeds the threshold, relevant IDs are allocated, and new entries are assigned new IDs (Hou et al., 2023b). Rectangular coordinates and midpoints are used to trace vehicle routes. Algorithm 1 provides the exact processes for estimating the trajectory.

4 Experimental setup and datasets

4.1 Experimental setup

PC running x64-based Windows 11, with an Intel Core i5-12500H 2.40GHz CPU, 24GB RAM and other specifications is used to perform all the experiments. Spyder was used to acquire the results. The system employed two benchmark datasets, VEDAI and SRTID, to calculate proposed architecture's performance. In this section, concise discussion of the dataset used for vehicle identification and tracking system is done, as well as the results of several tests undertaken to examine the proposed system along with its assessment in comparison to numerous existing state-of-the-art traffic monitoring models.

4.2 Dataset description

In the subsequent subsection, we provide comprehensive and detailed descriptions of each dataset used in our study. Each dataset is thoroughly introduced, highlighting its unique characteristics, data sources, and collection methods.

4.2.1 VEDAI dataset

The VEDAI dataset (Sakla et al., 2017) is a standard point of reference for tiny target identification, specifically aerial images vehicle detection. This dataset comprises roughly 1,210 images of two distinct dimensions such as 1,024 × 1,024 pixels and 512 × 512 pixels. Both near-infrared and visible light spectra environment photos are acquired in this collection. The cars in acquired aerial shots feature incredibly tiny dimensions, lighting/shadowing shifts, various backdrops, multiple forms, scale variations, and secularities or occlusions. Moreover, it comprises nine separate kinds of automobiles, including aircraft, boats, camping cars, automobiles, pick-ups, tractors, trucks, vans, and other categories.

4.2.2 Spanish road traffic images dataset

The dataset consists of 15,070 images in.png format, followed by an equal number of files with the txt extension containing descriptions of the objects found in each image. There are 30,140 files including images and information. The images were shot at six separate places along urban and interurban highways, with motorways being deleted. The images include 155,328 identified vehicles, including automobiles (137,602) and motorbikes (17,726) (Bemposta Rosende et al., 2022).

4.2.3 VAID dataset

The VAID collection consists of six vehicle image categories: minibus, truck, sedan, bus, van, and automobile. The images were taken at a height of 90–95 meters above the ground by a drone under a variety of lighting circumstances. The photographs, which were captured at a resolution of 2,720 × 1,530 and at a frame rate of 23.98 frames per second, show the state of the roads and traffic at 10 locations in southern Taiwan, encompassing suburban, urban, and educational environments (Lin et al., 2020).

4.2.4 UAVDT dataset

UAVDT dataset: Comprising 80,000 representative frames, the UAVDT dataset (Du et al., 2018) includes UAV imagery of cars

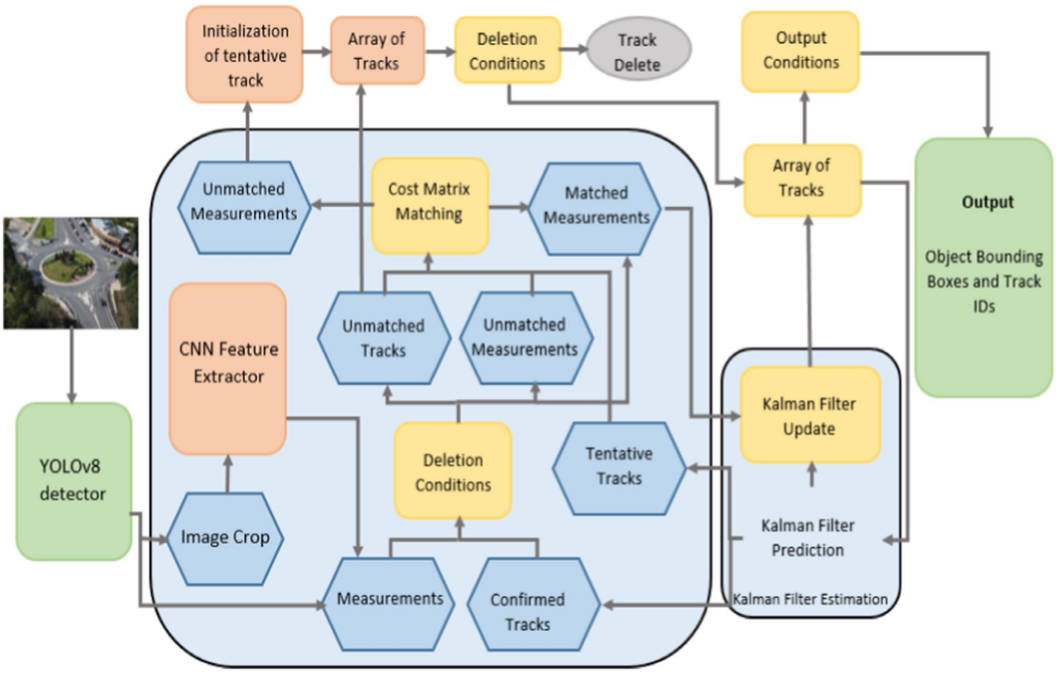


FIGURE 9
Steps of vehicle tracking using DeepSORT algorithm.

FIGURE 10
Tracking results using DeepSORT tracker across the image frames **(A)** Vehicle dectection only **(B)**. Multiple-object detection (0 = Vehicles, 1 = Bike, 2 = Pedestrians in frames).

chosen from 10-h long recordings. Bounding boxes with up to 14 different attributes (e.g., weather, flying altitude, camera view, vehicle category, occlusion, etc.) completely annotate the photos. Each of the three sets—training, val, and testing consists of 5,000, 1,658, and 3,316 images, all 1,024 × 540 pixels. The photographs from the same video have comparable backdrops, camera viewpoints, and lighting (for those recorded at the same time of day).

4.3 Experiment I: semantic segmentation accuracy

The DBSCAN and FCM algorithms were compared and assessed in terms of segmentation accuracy and computational time. DBSCAN requires training on a bespoke dataset, increasing the model's computing cost as compared to FCM. Furthermore, FCM produced superior segmentation results than DBSCAN, therefore we utilized the FCM findings for future investigation. Table 2 shows the accuracy of both segmentation strategies.

4.4 Experiment II: precision, recall, and F1 scores

The effectiveness of vehicle detection and tracking has been assessed using these evaluation metrics, namely Precision, Recall, and F1 score as calculated by using equations below:

$$\text{Precision} = \frac{\sum TP}{\sum TP + \sum FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1\,\text{Score} = \frac{2(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})}$$

Table 3 shows vehicle detection's precision, recall, and F1 scores on the segmented images, while Table 4 shows vehicle detection's precision, recall and F1 scores on the raw images. True Positive indicates how many cars are effectively identified. False Positives signify other detections besides cars, whereas False Negatives shows missing vehicles count. The findings indicate that this suggested system can accurately detect cars of varying sizes.

TABLE 2 Accuracies comparison of DBSCAN and FCM segmentation.

Datasets	Segmentations accuracy	
	DBSCAN	FCM
VEDAI	0.65	0.83
SRTID	0.68	0.79
VAID	0.62	0.72
UAVDT	0.65	0.75

In case of tracking, the number of cars successfully tracked is indicated as True Positive, whereas False Positive is the vehicles count falsely recorded, and False Negative represents untracked vehicles count. Table 4 shows the vehicle tracking method's precision, recall, and F1 scores (Figure 11).

4.5 Experiment

4.5.1 ID assignment and ID recovery

We used two new metrics to assess the ID assignment and recovery module, as shown in equations. The AID is the accurate ID rate, which is the proportion of correct ID numbers assigned to automobiles (Table 5).

$$\text{AIDRate} = \frac{\sum_{i=1}^{N} AIDs_i}{\sum_{i=1}^{N} ID_i}$$

TABLE 3 Precision, recall, and F1 Score for vehicle detection via YOLOv8 over segmented and raw images.

Datasets	Precision	Recall	F_1 score
VEDAI (segmented)	0.86	0.84	0.85
SRTID (segmented)	0.84	0.83	0.83
VAID (segmented)	0.85	0.82	0.83
UAVD (segmented)	0.81	0.82	0.81
VEDAI (raw)	0.83	0.80	0.81
SRTID (raw)	0.79	0.81	0.79
VAID (raw)	0.81	0.78	0.79
UAVDT (raw)	0.76	0.77	0.76

TABLE 4 Precision, recall, and F_1 score for vehicle tracking via DeepSORT.

Datasets	Precision	Recall	F_1 score
VEDAI	0.87	0.88	0.87
SRTID	0.83	0.82	0.82
VAID	0.88	0.85	0.86
UAVDT	0.84	0.83	0.83

TABLE 5 Precision, recall, and F_1 score for vehicle tracking via DeepSORT and ByteTrack.

Datasets	Precision	Recall	F_1 score	Tracking success rate
VEDAI	0.89	0.90	0.89	88.1%
SRTID	0.85	0.84	0.84	84.2%
VAID	0.90	0.87	0.88	87.5%
UAVDT	0.86	0.85	0.85	85.2%

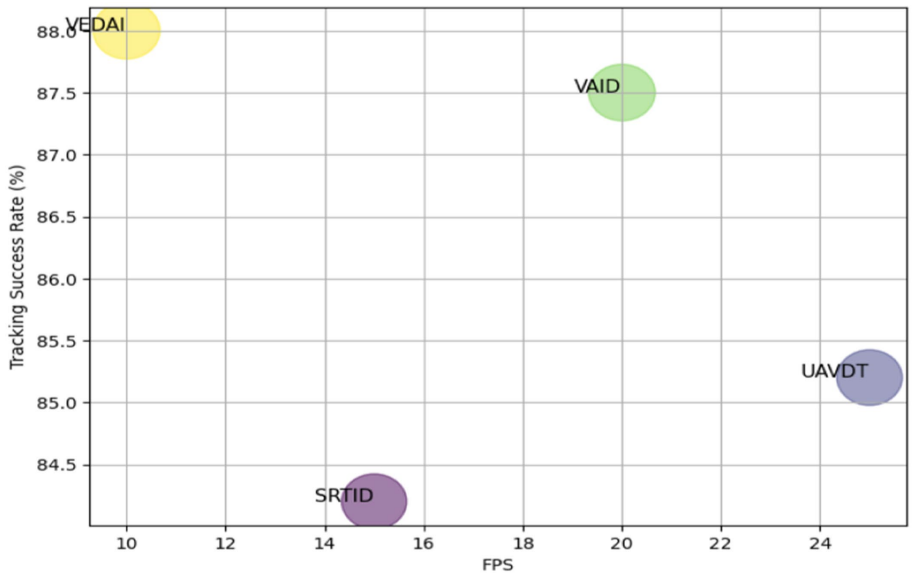


FIGURE 11
Tracking performance comparison of DeepSORT and ByteTrack across datasets.

TABLE 6 AIDRate and recovery rate for ID assignment recovery algorithm.

Datasets	AIDRate (%)	Recovery rate (%)
VEDAI	68	65
SRTID	63	59
VAID	59	55
UAVDT	65	60

where N is the total number of vehicles. $AIDs_i$ denotes the overall number of ID assignments made to the true vehicles, and ID_i denotes all of them. The Recovery Rate represents the percentage of true IDs recovered.

$$\text{Recovery Rate} = \frac{\sum_{i=1}^{N} TReCovers_i}{ReCovers}$$

where the total number of dissimilar vehicles is represented by N. $TReCovers_i$ represents the number of true recoveries and $ReCovers$ is the all-existing recoveries (Table 6).

4.6 Experiment IV: vehicle detection and tracking comparison with SOTA models

In this experiment, we have drawn a comparison of proposed model with other popular algorithms. Table 7 represents a comparison between our presented detection algorithm and other methods.

TABLE 7 Accuracy comparison of the proposed approach with SOTA vehicle detection models.

Methods	Accuracy %			
	VEDAI	SRTID	VAID	UAVDT
AVD NET (Mandal et al., 2020)	51.95	62.10	60.75	58.20
YOLOv5 (Hou S. et al., 2023)	75.54	73.20	74.30	72.10
Haar-like features (Nguyen and Tran, 2018)	77.0	65.00	64.50	62.00
D2Det (Cao et al., 2020)	73.40	56.92	68.10	64.30
R-FCN (Zhang, 2020)	68.90	73.0	69.80	70.20
SSD (Smith and Johnson, 2020)	71.00	70.30	81.0	74.50
R-FCN (Kim and Park, 2021)	72.50	70.90	75.0	73.20
NDFT (Cao et al., 2020)	63.50	62.80	64.00	52.03
YOLOv6 (Kumar and Singh, 2023)	74.07	71.20	72.80	74.07
YOLOv7 (Patel and Reddy, 2023)	76.80	74.50	75.60	72.0
Proposed method	79.4	77.7	83.1	77.2

TABLE 8 Accuracy comparison of the proposed approach with SOTA vehicle tracking models.

Methods	Accuracy %			
	VEDAI	SRTID	VAID	UAVDT
Faster R-CNN (du Terrail and Jurie, 2018)	83.50	78.0	81.0	79.5
Correlation filter tracking (Liu et al., 2019)	76.0	72.5	70.0	74.0
SIFT features (Mu et al., 2016)	72.5	75.10	73.0	71.0
HIOU (Hua and Anastasiu, 2019)	70.0	77.0	69.5	71.5
Kalman filter (Poostchi et al., 2017)	68.0	66.5	65.0	67.5
CNN (Alotaibi et al., 2020)	69.4	71.0	82.0	70.0
Affinity network (Cao et al., 2022)	73.2	74.0	71.5	74.0
MaSiamRPN (Sun et al., 2023)	82.0	79.1	83.0	84.0
Proposed method	88.6	82.2	84.6	86.1

Table 8 depicts the comparison of proposed tracking algorithm. Proposed model model performs better than other state-of-the-art methods.

5 Discussion/research limitation

For smart traffic monitoring based on aerial images, the suggested model is an efficient solution. While catering to high-definition aerial images, object detection is one of the most difficult problems. To get efficient results, we devised a technique that combines multi-label semantic segmentation with deepsort tracking. However, the suggested technique has significant limitations. First and foremost, the system has only been evaluated with RGB shots acquired during the daytime. Analyzing video or pictorial datasets in low-light conditions or at night can further confirm this proposed technique as a lot of researchers already have succeeded with such datasets. Furthermore, our segmentation and identification system have problems with partial or complete occlusions, tree-covered roadways, and similar items.

6 Conclusion

This study presents a novel approach to recognizing and tracking vehicles in aerial image sequences. Before proceeding with the detection phase, the model preprocesses aerial images to remove noise. To decrease complexity, the FCM approach is used for segmentation of all the images. The YOLOv8 algorithm is used for vehicle detection. It identifies vehicles by giving them a unique ID that contains ORB elements to aid recovery. DeepSORT tracks cars across frames and predicts their travel patterns. The suggested approach generated encouraging results across both datasets. The suggested system must be trained with additional vehicle classes. In addition, further elements may be added to increase vehicle recognition and tracking accuracy. In the future, we want to add additional features and dependable algorithms to the proposed model system to boost its efficiency and make it standard for all traffic scenarios.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found at: https://github.com/mr8bit/vedai https://www.kaggle.com/datasets/javiersanchezsoriano/traffic-images-captured-from-uavs.

Author contributions

MH: Methodology, Writing – original draft. MY: Data curation, Writing – original draft. NA: Investigation, Writing – review & editing. TS: Formal Analysis, Writing – review & editing. NAA: Project administration, Writing – review & editing. HR: Supervision, Writing – review & editing. AbA: Investigation, Writing – review & editing. AsA: Resources, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. The authors are thankful to the Deanship of Scientific Research at Najran University for funding this work under the Research Group Funding program grant code (NU/RG/SERC/13/18). This research is sup- ported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R410), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-FFR-2024-231-09."

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Adi, K., Widodo, A. P., Widodo, C. E., Pamungkas, A., and Putranto, A. B. (2018). Automatic vehicle counting using background subtraction method on gray scale images and morphology operation. *J. Phys.* 1025:012025. doi: 10.1088/1742-6596/1025/1/012025

Alotaibi, M. F., Omri, M., Abdel-Khalek, S., Khalil, E., and Mansour, R. F. (2022). Computational intelligence-based harmony search algorithm for real-time object detection and tracking in video surveillance systems. *Mathematics* 10:733. doi: 10.3390/math10050733

Alotaibi, M., Alhussein, M., Alqhtani, A., and Alghamdi, M. (2020). CNN: vehicle tracking in aerial images using a convolutional neural network. *IEEE Access* 8, 164725–164734. doi: 10.1109/ACCESS.2020.3026861

Amna, S., Jalal, A., and Kim, K.. (2020). An accurate facial expression detector using multi-landmarks selection and local transform features. In: IEEE ICACS Conference.

Angel, A., Hickman, M., Mirchandani, P., and Chandnani, D. (2003). Methods of analyzing traffic imagery collected from aerial platforms. *IEEE Trans. Intell. Transp. Syst.* 4, 99–107. doi: 10.1109/TITS.2003.821208

Aqel, S., Hmimid, A., Sabri, M. A., and Aarab, A. (2017). Road traffic: vehicle detection and classification. In Proceedings of the 2017 Intelligent Systems and Computer Vision (ISCV), Venice, pp. 17–19.

Bemposta Rosende, S., Ghisler, S., Fernández-Andrés, J., and Sánchez-Soriano, J. (2022). Dataset: traffic images captured from UAVs for use in training machine vision algorithms for traffic management. *Data* 7:53. doi: 10.3390/data7050053

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B.. (2016). Simple online and real-time tracking. In Proceedings of the International Conference Image Processing, ICIP, 2016-August, pp. 3464–3468.

Bhattacharjee, P., and Mitra, P. (2020). A survey of density-based clustering algorithms. *Front. Comput. Sci.* 15:151308. doi: 10.1007/s11704-019-9059-3

Bin Zuraimi, M. A., and Kamaru Zaman, F. H. (2021). Vehicle detection and tracking using YOLO and DeepSORT. In: 11th IEEE symposium on computer applications & industrial electronics (ISCAIE), Penang, Malaysia, 2021, pp. 23–29.

Bozcan, I., and Kayacan, E. (2020) AU-AIR: a multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance. In: 2020 IEEE ICRA, pp. 8504–8510.

Cai, D., Li, R., Hu, Z., Lu, J., Li, S., and Zhao, Y. (2024). A comprehensive overview of core modules in visual SLAM framework. *Neurocomputing* 590:127760. doi: 10.1016/j.neucom.2024.127760

Cao, J., Cholakkal, H., Anwer, R. M., Khan, F. S., Pang, Y., and Shao, L. (2020). "D2DET: towards high quality object detection and instance segmentation" in Proceedings of the IEEE/CVF Conf. On computer vision and pattern recognition (Seattle, WA), 11482–11491.

Cao, B., Li, M., Liu, X., Zhao, J., Cao, W., and Lv, Z. (2021). Many-objective deployment optimization for a drone-assisted camera network. *IEEE Trans Netw Sci Eng* 8, 2756–2764. doi: 10.1109/TNSE.2021.3057915

Cao, X., Yang, X., and Xu, W. (2022). Affinity network for multi-view object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 44, 1702–1715. doi: 10.1109/TPAMI.2021.3067218

Chen, X., and Meng, Q.. (2016). Robust vehicle tracking and detection from UAVs. In Proceedings of the 7th International Conference on Soft Computing Pattern Recognition, SoCPaR 2015, pp. 241–246.

Chen, J., Wang, Q., Cheng, H. H., Peng, W., and Xu, W. (2022). A review of vision-based traffic semantic understanding in ITSs. *IEEE Trans. Intell. Transp. Syst.* 23, 19954–19979. doi: 10.1109/TITS.2022.3182410

Chen, J., Wang, Q., Peng, W., Xu, H., Li, X., and Xu, W. (2023a). Disparity-based multiscale fusion network for transportation detection. *IEEE Trans. Intell. Transp. Syst.* 23, 18855–18863. doi: 10.1109/TITS.2022.3161977

Chen, Y., and Wu, Q.. (2016). Moving vehicle detection based on optical flow estimation of edge. In: Proceedings of the International Conference on Computing, 2016, pp. 754–758.

Chen, J., Xu, M., Xu, W., Li, D., Peng, W., and Xu, H. (2023b). A flow feedback traffic prediction based on visual quantified features. *IEEE Trans. Intell. Transp. Syst.* 24, 10067–10075. doi: 10.1109/TITS.2023.3269794

Chien, H.-J., Chuang, C.-C., Chen, C.-Y., and Klette, R. (2016). When to use what feature? SIFT, SURF, ORB, or A-KAZE features for monocular visual odometry. In: 2016 international conference on image and vision computing New Zealand (IVCNZ), New Zealand, pp. 1–6.

Chong, Q., Jindong, X., Ding, Y., and Dai, Z. (2023). A multiscale bidirectional fuzzy-driven learning network for remote sensing image segmentation. *IJRS* 44, 6860–6881. doi: 10.1080/01431161.2023.2275326

Dai, X., Xiao, Z., Jiang, H., and Lui, J. C. S. (2024). UAV-assisted task offloading in vehicular edge computing networks. *IEEE Trans. Mob. Comput.* 23, 2520–2534. doi: 10.1109/TMC.2023.3259394

Deng, Z. W., Zhao, Y. Q., Wang, B. H., Gao, W., and Kong, X. (2022). A preview driver model based on sliding-mode and fuzzy control for articulated heavy vehicle. *Meccanica* 57, 1853–1878. doi: 10.1007/s11012-022-01532-6

Di, Y., Li, R., Tian, H., Guo, J., Shi, B., Wang, Z., et al. (2023). A maneuvering target tracking based on fastIMM-extended Viterbi algorithm. *Neural Comput. Appl.* 35, 1–10. doi: 10.1007/s00521-023-09039-1

Dikbayir, H. S., and İbrahim Bülbül, H. (2020). "Deep learning based vehicle detection from aerial images" in 19th IEEE international conference on machine learning and applications (ICMLA) (Miami, FL), 956–960.

Ding, C., Li, C., Xiong, Z., Li, Z., and Liang, Q. (2024). Intelligent identification of moving trajectory of autonomous vehicle based on friction Nano-generator. *IEEE Trans. Intell. Transp. Syst.* 25, 3090–3097. doi: 10.1109/TITS.2023.3303267

Ding, Y., Zhang, W., Zhou, X., Liao, Q., Luo, Q., and Ni, L. M. (2021). FraudTrip: taxi fraudulent trip detection from corresponding trajectories. *IEEE Internet Things J.* 8, 12505–12517. doi: 10.1109/JIOT.2020.3019398

Drouyer, S., and de Franchis, C. (2019). Highway traffic monitoring on medium resolution satellite images. In IGARSS IEEE International Geoscience and Remote Sensing Symposium, pp. 1228–1231.

Du, D., Qi, Y., Yu, H., Yang, Y., Duan, K., Li, G., et al. (2018). The unmanned aerial vehicle benchmark: object detection and tracking. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 370–386.

du Terrail, J. O., and Jurie, F. (2018). Faster RER-CNN: application to the detection of vehicles in aerial images. *arXiv:1809.07628*. doi: 10.48550/arXiv.1809.07628

Gao, T., Li, K., Chen, T., Liu, M., Mei, S., Xing, K., et al. (2020). A novel UAV sensing image defogging method. *IEEE J. Select. Top. Appl. Earth Observ. Remote Sens.* 13, 2610–2625. doi: 10.1109/JSTARS.2020.2998517

Gu, Y., Hu, Z., Zhao, Y., Liao, J., and Zhang, W. (2024). MFGTN: a multi-modal fast gated transformer for identifying single trawl marine fishing vessel. *Ocean Eng.* 303:117711. doi: 10.1016/j.oceaneng.2024.117711

Hao, J., Chen, P., Chen, J., and Li, X. (2024). Multi-task federated learning-based system anomaly detection and multi-classification for microservices architecture. *Futur. Gener. Comput. Syst.* 159, 77–90. doi: 10.1016/j.future.2024.05.006

He, Y., and Li, L.. (2019). A novel multi-source vehicle detection algorithm based on deep learning. In Proceedings of the International Conference on Signal Processing, ICSP, vol. 2018, August, pp. 979–982.

Hinz, S., Lenhart, D., and Leitloff, J. (2006). "Detection and tracking of vehicles low framerate aerial image GIS integration car detection car tracking calculation of traffic parameters" in Image, International Society for Photogrammetry and Remote Sensing (ISPRS) (Rochester, NY).

Hou, S., Fan, L., Zhang, F., and Liu, B. (2023). An improved lightweight YOLOv5 for remote sensing images. In: Proceedings of the 32nd international conference on artificial neural networks, Heraklion, Greece, 26–29, pp. 77–89.

Hou, X., Xin, L., Fu, Y., Na, Z., Gao, G., Liu, Y., et al. (2023b). A self-powered biomimetic mouse whisker sensor (BMWS) aiming at terrestrial and space objects perception. *Nano Energy* 118:109034. doi: 10.1016/j.nanoen.2023.109034

Hou, X., Zhang, L., Su, Y., Gao, G., Liu, Y., Na, Z., et al. (2023a). A space crawling robotic bio-paw (SCRBP) enabled by triboelectric sensors for surface identification. *Nano Energy* 105:108013. doi: 10.1016/j.nanoen.2022.108013

Hua, S., and Anastasiu, D. C. (2019). Effective vehicle tracking algorithm for smart traffic networks. In: 2019 IEEE international conference on service-oriented system engineering (SOSE), San Francisco, CA, pp. 67–6709.

Huang, Z., Fang, H., Li, Q., Li, Z., Zhang, T., Sang, N., et al. (2018). Optical remote sensing image enhancement with weak structure preservation via spatially adaptive gamma correction. *Infrared Phys. Technol.* 94, 38–47. doi: 10.1016/j.infrared.2018.08.019

Huang, H., Meng, F., Zhou, S., Jiang, F., and Manogaran, G. (2019). Brain image segmentation based on FCM clustering algorithm and rough set. *IEEE Access* 7, 12386–12396. doi: 10.1109/ACCESS.2019.2893063

Khan, K., Rehman, S. U., Aziz, K., Fong, S., and Sarasvady, S.. (2014). DBSCAN: past, present and future. In: The fifth international conference on the applications of digital information and web technologies (ICADIWT), India, pp. 232–238.

Kim, J., and Park, S. (2021). Hybrid approach for vehicle detection in VAID using YOLOv5 and R-FCN. *J. Adv. Transport. Syst.* 28, 412–423. doi: 10.1007/s11554-021-01078-5

Kumar, R., and Singh, A. (2023). Efficient vehicle detection in UAVDT dataset using YOLOv6 and deep SORT. *IEEE Trans. Intell. Transport. Syst.* 34, 405–417. doi: 10.1109/TITS.2022.3184512

Kumar, S., Singh, S. K., Varshney, S., Singh, S., Kumar, P., Kim, B.-G., et al. (2023). Fusion of deep Sort and Yolov5 for effective vehicle detection and tracking scheme in real-time traffic management sustainable system. *Sustain. For.* 15:16869. doi: 10.3390/su152416869

Leitloff, J., Rosenbaum, D., Kurz, F., Meynberg, O., and Reinartz, P. (2014). An operational system for estimating road traffic information from aerial images. *Remote Sens.* 6, 11315–11341. doi: 10.3390/rs61111315

Li, S., Chen, J., Peng, W., Shi, X., and Bu, W. (2023). A vehicle detection method based on disparity segmentation. *Multimed. Tools Appl.* 82, 19643–19655. doi: 10.1007/s11042-023-14360-x

Li, J., Han, L., Zhang, C., Li, Q., and Liu, Z. (2023). Spherical convolution empowered viewport prediction in 360 video multicast with limited FoV feedback. *ACM Trans. Multimedia Comput. Commun. Appl.* 19, 1–23. doi: 10.1145/3511603

Li, Z., Wang, Y., Zhang, R., Ding, F., Wei, C., and Lu, J. (2024). A LiDAR-OpenStreetMap matching method for vehicle global position initialization based on boundary directional feature extraction. *IEEE Trans. Intell. Vehicles* 9, 1–13. doi: 10.1109/TIV.2024.3393229

Li, F., Zhang, S., Wu, Y., and Huang, L. (2018). Yolov3: a real-time vehicle tracking system based on deep learning. *IEEE Access* 6, 9135–9143.

Lin, C.-J., and Jhang, J.-Y. (2022). Intelligent traffic-monitoring system based on YOLO and convolutional fuzzy neural networks. *IEEE Access* 10, 14120–14133. doi: 10.1109/ACCESS.2022.3147866

Lin, H. Y., Tu, K. C., and Li, C. Y. (2020) "VAID: An aerial image dataset for vehicle detection and classification." IEEE Access, vol. 8, pp. 212209–212219.

Liu, L., Zhang, S., Zhang, L., Pan, G., and Yu, J. (2023). Multi-UUV maneuvering counter-game for dynamic target scenario based on fractional-order recurrent neural network. *IEEE Trans. Cybern.* 53, 4015–4028. doi: 10.1109/TCYB.2022.3225106

Liu, Z., Wang, X., Yu, B., Xu, X., and Yang, Y. (2019). Vehicle detection in aerial images using RetinaNet and correlation filter tracking. *J. Remote Sens. Technol.* 10, 134–149. doi: 10.1016/j.jrst.2019.04.003

Luo, G., Shao, C., Cheng, N., Zhou, H., Zhang, H., Yuan, Q., et al. (2024). EdgeCooper: network-aware cooperative LiDAR perception for enhanced vehicular awareness. *IEEE J Sel Areas Commun* 42, 207–222. doi: 10.1109/JSAC.2023.3322764

Mandal, M., Shah, M., Meena, P., Devi, S., and Vipparthi, S. K. (2020). AVDNet: a small-sized vehicle detection network for aerial visual data. *IEEE Geosci. Remote Sens. Lett.* 17, 494–498. doi: 10.1109/LGRS.2019.2923564

Mi, C., Liu, Y., Zhang, Y., Wang, J., Feng, Y., and Zhang, Z. (2023). A vision-based displacement measurement system for foundation pit. *IEEE Trans. Instrum. Meas.* 72, 1–15. doi: 10.1109/TIM.2023.3311069

Minh, K. T., Dinh, Q.-V., Nguyen, T.-D., and Nhut, T. N.. (2023). Vehicle counting on Vietnamese street. In: IEEE statistical signal processing workshop (SSP), Hanoi, Vietnam, 2023, pp. 160–164.

Mostofa, M., Ferdous, S. N., and Nasrabadi, N. M. (2020). A joint cross-modal super-resolution approach for vehicle detection in aerial imagery. *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II* 11413, 184–194.

Mu, K., Hui, F., and Zhao, X. (2016). Multiple vehicle detection and tracking in highway traffic surveillance video based on sift feature matching. *J. Inf. Process. Syst.* 12, 183–195. doi: 10.3745/JIPS.02.0040

Najiya, K. V., and Archana, M.. (2018). UAV video processing for traffic surveillance with enhanced vehicle detection. In Proceedings of the International Conference on Inventive Communication and Computational Technologies, pp. 662–668.

Nguyen, T., and Tran, P. (2018). A combined approach for vehicle detection using YOLOv2 and traditional feature-based methods. *Mach. Learn. Remote Sens.* 8, 89–100. doi: 10.1016/j.mlsr.2018.03.003

Omar, W., Oh, Y., Chung, J., and Lee, I. (2021). Aerial dataset integration for vehicle detection based on YOLOv4. *Kor. J. Remote Sens.* 37, 747–761. doi: 10.7780/kjrs.2021.37.4.6

Ozturk, M., and Cavus, E. (2021). "Vehicle detection in aerial imaginary using a miniature CNN architecture" in Proceedings of the 2021 international conference on innovations in intelligent systems and applications (INISTA) (Kocaeli), 1–6.

Patel, N., and Reddy, S. (2023). Real-time vehicle detection in UAVDT using YOLOv7 and SORT. *J. Aerial Robot.* 16, 120–135. doi: 10.1002/rob.22082

Peng, J. J., Chen, X. G., Wang, X. K., Wang, J. Q., Long, Q. Q., and Yin, L. J. (2023). Picture fuzzy decision-making theories and methodologies: a systematic review. *Int. J. Syst. Sci.* 54, 2663–2675. doi: 10.1080/00207721.2023.2241961

Poostchi, M., Palaniappan, K., and Seetharaman, G.., (2017). Spatial pyramid context-aware moving vehicle detection and tracking in urban aerial imagery. In: 2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS), Lecce, pp. 1–6.

Qu, Z., Liu, X., and Zheng, M. (2022). Temporal-spatial quantum graph convolutional neural network based on Schrödinger approach for traffic congestion prediction. *IEEE Trans. Intell. Transp. Syst.* 24, 8677–8686. doi: 10.1109/TITS.2022.3203791

Rafique, A. A., al-Rasheed, A., Ksibi, A., Ayadi, M., Jalal, A., Alnowaiser, K., et al. (2023). Smart traffic monitoring through pyramid pooling vehicle detection and filter-based tracking on aerial images. *IEEE Access* 11, 2993–3007. doi: 10.1109/ACCESS.2023.3234281

Rehman, S. N., and Hussain, M. A. (2018). Fuzzy C-means algorithm-based satellite image segmentation. *Indones. J. Electr. Eng. Comput. Sci.* 9, 332–334. doi: 10.11591/ijeecs.v9.i2.pp332-334

Ren, Y., Lan, Z., Liu, L., and Yu, H. (2024). EMSIN: enhanced multi-stream interaction network for vehicle trajectory prediction. *IEEE Trans. Fuzzy Syst.* 32, 1–15. doi: 10.1109/TFUZZ.2024.3360946

Rong, Y., Xu, Z., Liu, J., Liu, H., Ding, J., Liu, X., et al. (2022). Du-bus: a realtime bus waiting time estimation system based on multi-source data. *IEEE Trans. Intell. Transp. Syst.* 23, 24524–24539. doi: 10.1109/TITS.2022.3210170

Sakla, W., Konjevod, G., and Mundhenk, T. N. (2017). Deep multi-modal vehicle detection in aerial ISR imagery. In: IEEE winter conference on applications of computer vision (WACV), Santa Rosa, CA, pp. 916–923.

Schreuder, M., Hoogendoorn, S. P., van Zulyen, H. J., Gorte, B., and Vosselman, G. (2003). Traffic data collection from aerial imagery. *IEEE Trans. Intell. Transp.* 1, 779–784. doi: 10.1109/ITSC.2003.1252056

Sheng, H., Wang, S., Chen, H., Yang, D., Huang, Y., Shen, J., et al. (2024). Discriminative feature learning with co-occurrence attention network for vehicle ReID. *IEEE Trans. Circuits Syst. Video Technol.* 34, 3510–3522. doi: 10.1109/TCSVT.2023.3326375

Shi, Y., Xi, J., Hu, D., Cai, Z., and Xu, K. (2023). RayMVSNet++: learning ray-based 1D implicit fields for accurate multi-view stereo. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 1–17. doi: 10.1109/TPAMI.2023.3296163

Singh, I. S., Wijegunawardana, I. D., Samarakoon, S. M. B. P., Muthugala, M. A. V. J., and Elara, M. R. (2023). Vision-based dirt distribution mapping using deep learning. *Sci. Rep.* 13:12741. doi: 10.1038/s41598-023-38538-3

Smith, K., and Johnson, L. (2020). Vehicle detection and tracking in VAID dataset using SSD and deep SORT. *Adv. Comput. Vis.* 12, 350–362. doi: 10.3390/s20154296

Song, F., Liu, Y., Shen, D., Li, L., and Tan, J. (2022). Learning control for motion coordination in water scanners: toward gain adaptation. *IEEE Trans. Ind. Electron.* 69, 13428–13438. doi: 10.1109/TIE.2022.3142428

Sun, R., Dai, Y., and Cheng, Q. (2023). An adaptive weighting strategy for multisensor integrated navigation in urban areas. *IEEE Internet Things J.* 10, 12777–12786. doi: 10.1109/JIOT.2023.3256008

Sun, G., Sheng, L., Luo, L., and Yu, H. (2022). Game theoretic approach for multipriority data transmission in 5G vehicular networks. *IEEE Trans. Intell. Transp. Syst.* 23, 24672–24685. doi: 10.1109/TITS.2022.3198046

Sun, G., Song, L., Yu, H., Chang, V., Du, X., and Guizani, M. (2019). V2V routing in a VANET based on the autoregressive integrated moving average model. *IEEE Trans. Veh. Technol.* 68, 908–922. doi: 10.1109/TVT.2018.2884525

Sun, G., Zhang, Y., Liao, D., Yu, H., Du, X., and Guizani, M. (2018). Bus-trajectory-based street-centric routing for message delivery in urban vehicular ad hoc networks. *IEEE Trans. Veh. Technol.* 67, 7550–7563. doi: 10.1109/TVT.2018.2828651

Sun, G., Zhang, Y., Yu, H., Du, X., and Guizani, M. (2020). Intersection fog-based distributed routing for V2V communication in urban vehicular ad hoc networks. *IEEE Trans. Intell. Transp. Syst.* 21, 2409–2426. doi: 10.1109/TITS.2019.2918255

Sun, Y., Zhao, Y., and Wang, S. (2022). Multiple traffic target tracking with spatial-temporal affinity network. *Comput. Intell. Neurosci* 2022:9693767. doi: 10.1155/2022/9693767

Tang, Q., Qu, S., Zhang, C., Tu, Z., and Cao, Y. (2024). Effects of impulse on prescribed-time synchronization of switching complex networks. *Neural Netw.* 174:106248. doi: 10.1016/j.neunet.2024.106248

Teutsch, M., Krüger, W., and Beyerer, J. (2017). Moving object detection in top-view aerial videos improved by image stacking. *Opt. Eng.* 56:083102. doi: 10.1117/1.OE.56.8.083102

Tian, J., Wang, B., Guo, R., Wang, Z., Cao, K., and Wang, X. (2022). Adversarial attacks and defenses for deep-learning-based unmanned aerial vehicles. *IEEE Internet Things J.* 9, 22399–22409. doi: 10.1109/JIOT.2021.3111024

Wang, G., Chen, Y., An, P., Hong, H., Hu, J., and Huang, T. (2023). UAV-YOLOv8: a small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios. *Sensors* 23:7190. doi: 10.3390/s23167190

Wang, R., Gu, Q., Lu, S., Tian, J., Yin, Z., Yin, L., et al. (2024). FI-NPI: exploring optimal control in parallel platform systems. *Electronics* 13:1168. doi: 10.3390/electronics13071168

Wang, S., Sheng, H., Yang, D., Zhang, Y., Wu, Y., and Wang, S. (2022). Extendable multiple nodes recurrent tracking framework with RTU++. *IEEE Trans. Image Process.* 31, 5257–5271. doi: 10.1109/TIP.2022.3192706

Weng, S. K., Kuo, C. M., and Tu, S. K. (2006). Video object tracking using adaptive Kalman filter. *J. Vis. Commun. Image Represent.* 17, 1190–1208. doi: 10.1016/j.jvcir.2006.03.004

Wu, T., and Dong, Y. (2023). YOLO-SE: improved YOLOv8 for remote sensing object detection and recognition. *Appl. Sci.* 13:12977. doi: 10.3390/app132412977

Wu, Z., Zhu, H., He, L., Zhao, Q., Shi, J., and Wu, W. (2023). Real-time stereo matching with high accuracy via spatial attention-guided Upsampling. *Appl. Intell.* 53, 24253–24274. doi: 10.1007/s10489-023-04646-w

Wu, W., Zhu, H., Yu, S., and Shi, J. (2019). Stereo matching with fusing adaptive support weights. *IEEE Access* 7, 61960–61974. doi: 10.1109/ACCESS.2019.2916035

Xiao, Z., Shu, J., Jiang, H., Min, G., Chen, H., and Han, Z. (2023). Overcoming occlusions: perception task-oriented information sharing in connected and autonomous vehicles. *IEEE Netw.* 37, 224–229. doi: 10.1109/MNET.018.2300125

Xiao, Z., Shu, J., Jiang, H., Min, G., Liang, J., and Iyengar, A. (2024). Toward collaborative occlusion-free perception in connected autonomous vehicles. *IEEE Trans. Mob. Comput.* 23, 4918–4929. doi: 10.1109/TMC.2023.3298643

Xu, H., Cao, Y., Lu, Q., and Yang, Q. (2020). "Performance comparison of small object detection algorithms of UAV based aerial images" in 2020 19th international symposium on distributed computing and applications for business engineering and science (DCABES) (Xuzhou), 16–19.

Xu, X., Liu, W., and Yu, L. (2022). Trajectory prediction for heterogeneous traffic-agents using knowledge correction data-driven model. *Inf. Sci.* 608, 375–391. doi: 10.1016/j.ins.2022.06.073

Xuemin, Z., Haitao, D., Zenggang, X., Ying, R., Yanchao, L., Yuan, L., et al. (2024). Self-organizing key security management algorithm in socially aware networking. *J. Signal Process. Syst.* 96, 369–383. doi: 10.1007/s11265-024-01918-7

Yang, D., Cui, Z., Sheng, H., Chen, R., Cong, R., Wang, S., et al. (2023). An occlusion and noise-aware stereo framework based on light field imaging for robust disparity estimation. *IEEE Trans. Comput.* 73, 764–777. doi: 10.1109/TC.2023.3343098

Yang, M., Han, W., Song, Y., Wang, Y., and Yang, S. (2024). Data-model fusion driven intelligent rapid response design of underwater gliders. *Adv. Eng. Inform.* 61:102569. doi: 10.1016/j.aei.2024.102569

Yao, Y., Zhao, B., Zhao, J., Shu, F., Wu, Y., and Cheng, X. (2024). Anti-jamming technique for IRS aided JRC system in Mobile vehicular networks. *IEEE Trans. Intell. Transp. Syst.* 25, 1–11. doi: 10.1109/TITS.2024.3384038

Yi, H., Liu, B., Zhao, B., and Liu, E. (2024). Small object detection algorithm based on improved YOLOv8 for remote sensing. *IEEE J. Select. Top. Appl. Earth Observ. Remote Sens.* 17, 1734–1747. doi: 10.1109/JSTARS.2023.3339235

Yin, Y., Guo, Y., Su, Q., and Wang, Z. (2022). Task allocation of multiple unmanned aerial vehicles based on deep transfer reinforcement learning. *Drones* 6:215. doi: 10.3390/drones6080215

Zhang, X. (2020). Advanced vehicle detection in aerial images using YOLOv3 and R-FCN. *J. Real-Time Image Process.* 17, 77–88. doi: 10.1007/s11554-019-00843-4

Zhang, Y., Li, S., Wang, S., Wang, X., and Duan, H. (2023). Distributed bearing-based formation maneuver control of fixed-wing UAVs by finite-time orientation estimation. *Aerosp. Sci. Technol.* 136:108241. doi: 10.1016/j.ast.2023.108241

Zhang, X., and Zhu, X. (2019). Vehicle detection in aerial infrared images via an improved Yolov3 network. In: Proceedings of the 2019 IEEE 4th international conference on signal and image processing (ICSIP), Wuxi, China, pp. 372–376.

Zhao, X., Fang, Y., Min, H., Wu, X., Wang, W., and Teixeira, R. (2024). Potential sources of sensor data anomalies for autonomous vehicles: An overview from road vehicle safety perspective. *Expert Syst. Appl.* 236:121358. doi: 10.1016/j.eswa.2023.121358

Zhao, L., Xu, H., Qu, S., Wei, Z., and Liu, Y. (2024). Joint trajectory and communication design for UAV-assisted symbiotic radio networks. *IEEE Trans. Veh. Technol.* 73, 8367–8378. doi: 10.1109/TVT.2024.3356587

Zheng, W., Lu, S., Yang, Y., Yin, Z., Yin, L., and Ali, H. (2024). Lightweight transformer image feature extraction network. *PeerJ Comput. Sci.* 10:e1755. doi: 10.7717/peerj-cs.1755

Zhu, H., Xu, D., Huang, Y., Jin, Z., Ding, W., Tong, J., et al. (2024). Graph structure enhanced pre-training language model for knowledge graph completion. *IEEE Trans. Emerg. Top. Comput. Intell.* 8, 2697–2708. doi: 10.1109/TETCI.2024.3372442

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 19 August 2024
DOI 10.3389/fnbot.2024.1451924

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Boyu Ma,
Harbin Institute of Technology, China
Haoen Huang,
Huazhong University of Science and Technology, China

*CORRESPONDENCE
Haitao Fu
✉ fht@jlau.edu.cn

RECEIVED 20 June 2024
ACCEPTED 22 July 2024
PUBLISHED 19 August 2024

CITATION
Li J, Su J, Yu W, Mao X, Liu Z and Fu H (2024) Recurrent neural network for trajectory tracking control of manipulator with unknown mass matrix. *Front. Neurorobot.* 18:1451924. doi: 10.3389/fnbot.2024.1451924

Recurrent neural network for trajectory tracking control of manipulator with unknown mass matrix

Jian Li, Junming Su, Weilin Yu, Xuping Mao, Zipeng Liu and Haitao Fu*

College of Information Technology, Jilin Agricultural University, Changchun, China

Real-world robotic operations often face uncertainties that can impede accurate control of manipulators. This study proposes a recurrent neural network (RNN) combining kinematic and dynamic models to address this issue. Assuming an unknown mass matrix, the proposed method enables effective trajectory tracking for manipulators. In detail, a kinematic controller is designed to determine the desired joint acceleration for a given task with error feedback. Subsequently, integrated with the kinematics controller, the RNN is proposed to combine the robot's dynamic model and a mass matrix estimator. This integration allows the manipulator system to handle uncertainties and synchronously achieve trajectory tracking effectively. Theoretical analysis demonstrates the learning and control capabilities of the RNN. Simulative experiments conducted on a Franka Emika Panda manipulator, and comparisons validate the effectiveness and superiority of the proposed method.

KEYWORDS

recurrent neural network (RNN), trajectory tracking, manipulator control, dynamic model, unknown mass matrix

1 Introduction

With the rapid development of modern robot research and development technology, manipulators have permeated various aspects of human life, such as space explorations (Ma et al., 2023a) and smart factories (Abate et al., 2022). Its fundamental functionality lies in trajectory tracking, where specific tasks are accomplished by executing predefined end-effector trajectories (Jin et al., 2024b). This involves the control of robot kinematics and dynamics (Liao et al., 2022; Lian et al., 2024; Sun et al., 2024). To exert control over the robot, desired joint attributes should be obtained according to the task trajectory and converted into the corresponding joint torques (Müller et al., 2023). Numerous algorithms, such as pseudoinverse methods (Guo et al., 2018; Sun et al., 2023a) and model predictive control method (Jin et al., 2023), have been designed to achieve precise control of the manipulator. However, these algorithms rely on accurate robot models and struggle to control the robot effectively when its parameters change. In practical applications, it is common for robot model parameters to vary, especially when robots are modified to perform different tasks in diverse application scenarios (Xiao et al., 2022; Xie and Jin, 2024). Reliable model-free control methods need to be designed to enable effective control of robots after the parameter changes.

In recent decades, there are many emerging algorithms to address the control issues of manipulators (Liao et al., 2024; Yan et al., 2024), which are considered from the velocity level (Zhang et al., 2019; Sun et al., 2022), acceleration level (Wen and Xie, 2024), or torque level (Hua et al., 2023). For instance, to eliminate the joint-angle drift and prevent excessive joint velocity, a velocity-level bi-criteria optimization scheme is provided for coordinated path tracking of manipulators, focusing on the velocity aspect (Xiao et al., 2017). Additionally, a data-driven acceleration-level scheme is introduced to address control continuity and stability issues for manipulator (Wen and Xie, 2024). However, most of these studies focus solely on kinematics, neglecting dynamic factors (Tang and Zhang, 2022). Robot kinematics and dynamics are two fundamental domains within the field of robotics. Robot kinematics focuses on the study of the motion capabilities of robots in space, encompassing aspects such as joint angles, positions, velocities, and accelerations, without considering the effects of forces (Xie et al., 2023). In addition, robot dynamics is concerned with the impact of forces and torques on the motion of the manipulator, including the interactions between the robot and its environment. In the control of robotic manipulators, considering dynamic factors can help precisely predict the actual motion trajectory of the manipulator under various load and motion conditions, thereby improving the overall motion accuracy (Sun et al., 2023b; Xiao et al., 2023). It can also compensate for the oscillation and coupling effects in joint motion, making the movement of the manipulator smooth and stable. Furthermore, the dynamics-based studies aid in selecting the optimal drive scheme, reducing energy consumption, and enhancing energy utilization efficiency. However, manipulators frequently encounter issues with dynamic uncertainties due to the diversity of robotic grippers and uncertainties in load (Bruder et al., 2021; Liu et al., 2024b). Specifically, surgical manipulators may be equipped with different end-effectors to meet various task requirements, implying changes in dynamic parameters (Liu et al., 2024b). Moreover, in robotic-grasping tasks, unknown loads also lead to variations in robot dynamic parameters (Bruder et al., 2021). Dynamic uncertainties significantly impede the accurate control of manipulators, highlighting its research significance.

Recurrent neural networks (RNNs) have emerged as effective robot control algorithms in recent years (Liao et al., 2023; Ma et al., 2023b; Jin et al., 2024a). RNN is utilized to establish a scheme for addressing the coordination problem for multirobot systems (Cao et al., 2023; Liu et al., 2024a). In addition, RNN can mitigate uncertainties in the robot systems by enabling online learning of robot parameters (Xie et al., 2022). However, further research is needed to explore the integration of synchronous dynamic parameter learning with the kinematic model to achieve accurate trajectory tracking (Tang et al., 2024). To this aim, this study assumes the presence of deviations in the robot dynamic model and proposes an RNN for the model-free control of manipulators. Specifically, relevant control algorithms are designed at both the kinematic and dynamic levels, and an estimator of the mass matrix is proposed to compensate for the uncertainty of the dynamic model. Further verifications are carried out on a Franka Emika Panda manipulator to perform a trajectory-tracking task, taking into account dynamic uncertainties. In addition, compared with the existing methods, the superiorities of the proposed RNN lie in the following two aspects:

- Compared with kinematics-based methods (Guo et al., 2018; Jin et al., 2023, 2024b), the proposed RNN bridges the robot kinematics and robot dynamics models through joint acceleration signals, considering the motion feature and the dynamic behavior of manipulators.
- Compared with dynamics-based methods (Shojaei et al., 2021; Zong and Emami, 2021), the proposed RNN addresses the dynamic uncertainty problem by estimating the mass matrix online and realizes synchronous trajectory tracking.

Through the introduction of the above basic content, the specific research of this study is organized as follows. Section 2 explains the kinematic relationship between the joint angle of the manipulator and the end-effector. In Section 3, a corresponding RNN is designed. Subsequently, the learning and control ability of the proposed RNN are analyzed theoretically in Section 4. Finally, simulations and comparisons are carried out in Section 5.

2 Kinematic controller

The forward kinematics of the manipulator describes the mapping relationship between the joint angle and the end-effector position, described as $f(\boldsymbol{q}) = \boldsymbol{r}$, where $\boldsymbol{q} \in \mathbb{R}^a$ is the joint angle, $\boldsymbol{r} \in \mathbb{R}^b$ denotes the position of the end-effector, and $f(\cdot)$ stands for the non-linear mapping. Furthermore, the time derivative of the forward kinematics is derived as

$$\boldsymbol{L}\dot{\boldsymbol{q}} = \dot{\boldsymbol{r}}, \tag{1}$$

where $\boldsymbol{L} = \partial f(\boldsymbol{q})/\partial \boldsymbol{q} \in \mathbb{R}^{b\times a}$ is the Jacobian matrix, $\dot{\boldsymbol{q}}$ denotes the joint velocity, and $\dot{\boldsymbol{r}}$ is the velocity of the end-effector. Concerning the joint acceleration level, taking the time derivative of Equation 1 leads to

$$\dot{\boldsymbol{L}}\dot{\boldsymbol{q}} + \boldsymbol{L}\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{r}}, \tag{2}$$

where $\dot{\boldsymbol{L}}$ is the time derivative of $\boldsymbol{L}$, $\ddot{\boldsymbol{q}}$ represents the joint acceleration, and $\ddot{\boldsymbol{r}}$ denotes the acceleration of the end-effector. Building upon Equation 2, the desired joint acceleration can be obtained by the following kinematic controller:

$$\ddot{\boldsymbol{q}} = \boldsymbol{L}^{\dagger}(\ddot{\boldsymbol{r}}_{\mathrm{d}} - \dot{\boldsymbol{L}}\dot{\boldsymbol{q}} - \boldsymbol{\nu}), \tag{3}$$

where $\boldsymbol{\nu} = \beta(\dot{\boldsymbol{r}} - \dot{\boldsymbol{r}}_{\mathrm{d}}) + \zeta(\boldsymbol{r} - \boldsymbol{r}_{\mathrm{d}})$ denotes the error feedback term; $\boldsymbol{r}_{\mathrm{d}}$, $\dot{\boldsymbol{r}}_{\mathrm{d}}$, and $\ddot{\boldsymbol{r}}_{\mathrm{d}}$ are the desired position, velocity, and acceleration of the trajectory tracking task; superscript † denotes the pseudoinverse operation of a matrix with $\boldsymbol{L}^{\dagger} = \boldsymbol{L}^{\mathrm{T}}(\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}})^{-1}$; and $\beta > 0$ and $\alpha > 0$ are convergence coefficients. On the one hand, kinematic controller (Equation 3) utilizes the minimization function of the pseudoinverse operation to obtain the desired joint acceleration (Wen and Xie, 2024). On the other hand, it takes the desired trajectory tracking task as input and incorporates feedback of the tracking error, leading to improved trajectory tracking performance. In addition, kinematic controller

(Equation 3) can apply traditional methods to avoid the singularity issues, such as the damped least squares method (Xie et al., 2024). Specifically, a damped term can be added in the computation of the pseudoinverse. The specific calculation formula is $\boldsymbol{L}^{\mathrm{T}}(\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}+\mu\boldsymbol{I})^{-1}$, where $\mu > 0$ denotes a tiny parameter and $\boldsymbol{I}$ denotes the identity matrix. By doing so, the infinite values caused by zero eigenvalues in the pseudoinverse operation can be avoided.

3 Recurrent neural network design

Robot dynamics refers to the mathematical description of the relationship between joint torques, dynamic parameters, and joint motions in a robotic system. Specifically, the dynamic model of a manipulator can be written as

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}), \tag{4}$$

where $\boldsymbol{\tau} \in \mathbb{R}^a$ represents the joint torque, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{a\times a}$ is the mass matrix, $\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) \in \mathbb{R}^a$ is the Coriolis and centrifugal vector, and $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^a$ denotes the gravity vector. Generally, traditional methods, such as Guo et al. (2018), are capable of performing accurate dynamic control by relying on precise dynamic models (Equation 4). However, in real-world applications, it is common for manipulators to undergo modifications to perform various tasks, resulting in changes in their dynamic parameters. Given the assumption that the change occurs in the inertia matrix, we design an estimated inertia matrix $\bar{\boldsymbol{M}} \in \mathbb{R}^{a\times a}$ to effectively mitigate dynamic uncertainties. As a result, the following state equation is established:

$$\bar{\boldsymbol{\tau}} = \bar{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}), \tag{5}$$

where $\bar{\boldsymbol{\tau}} \in \mathbb{R}^a$ is the corresponding joint torque. When the estimated inertia matrix converges to the actual one, it indicates that the dynamic uncertainty issue is solved. To this aim, an estimation equation is presented as follows:

$$\dot{\bar{\boldsymbol{M}}} = \alpha(\boldsymbol{\tau} - \bar{\boldsymbol{\tau}})\ddot{\boldsymbol{q}}^{\dagger}, \tag{6}$$

where $\dot{\bar{\boldsymbol{M}}}$ determines the evolution direction of $\bar{\boldsymbol{M}}$, and $\alpha > 0$ stands for the convergence coefficient. In Equation 6, $\boldsymbol{\tau}$ is measured in real time. Combining Equations 3, 5, 6, an RNN is designed as follows:

$$\dot{\bar{\boldsymbol{M}}} = \alpha(\boldsymbol{\tau} - \bar{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}))\ddot{\boldsymbol{q}}^{\dagger}, \tag{7a}$$

$$\boldsymbol{\tau}_{\mathrm{out}} = \bar{\boldsymbol{M}}(\boldsymbol{q})(\boldsymbol{L}^{\dagger}(\ddot{\boldsymbol{r}}_{\mathrm{d}} - \dot{\boldsymbol{L}}\dot{\boldsymbol{q}}) - \boldsymbol{v}) + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}), \tag{7b}$$

where $\boldsymbol{\tau}_{\mathrm{out}}$ is the output signal and $\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})$. In addition, a control flow chart of RNN (Equation 7) is shown in Figure 1. Notably, the joint acceleration generated by Equation 3 in a kinematic manner serves as the input for Equation 7b. Furthermore, Equation 7a utilizes measurement data $\boldsymbol{\tau}$ to estimate the mass matrix, which, in turn, facilitates the precise control of Equation 7b. In this context, RNN (Equation 7) demonstrates its capability to learn the mass matrix and achieve synchronous trajectory tracking via the joint torque. The parameters in RNN (Equation 7) include α, β, and ζ, which are determined through trial and error methods.

In RNN (Equation 7), we first apply kinematic controller (Equation 3) to output the joint acceleration corresponding to the trajectory task. This process belongs to the inverse kinematics solution. Subsequently, we further obtain the joint torque by calculating the obtained joint acceleration. This process belongs to the inverse dynamics of solution. Finally, the output joint torque can directly control the manipulator to perform the given task. In this control mode, robot kinematics and dynamics are combined together with joint acceleration to form a bridge.

4 Theoretical analysis

The following theorem provides a verification of the learning and control capabilities of the proposed RNN (Equation 7).

Theorems: Assuming a sufficiently large value of α, the estimated error $\boldsymbol{M} - \bar{\boldsymbol{M}}$ generated by Equation 7a is global convergent to a zero matrix. Based on the estimated mass matrix, Equation 7b enables accurate trajectory-tracking control of the manipulator with an unknown mass matrix.

Proof: By incorporating Equations 2, 3, we can rewrite Equation 7b as $\dot{\bar{\boldsymbol{M}}} = \alpha(\boldsymbol{M} - \bar{\boldsymbol{M}})\ddot{\boldsymbol{q}}\ddot{\boldsymbol{q}}^{\dagger}$. Multiplying both sides of the equation by $\ddot{\boldsymbol{q}}$ yields $\dot{\bar{\boldsymbol{M}}}\ddot{\boldsymbol{q}} = \alpha(\boldsymbol{M} - \bar{\boldsymbol{M}})\ddot{\boldsymbol{q}}$. Then, it follows that $\dot{\bar{\boldsymbol{M}}} = \alpha(\boldsymbol{M} - \bar{\boldsymbol{M}})$. Define an estimated error as $\boldsymbol{e} = (\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)$ with $\boldsymbol{M}_i$ and $\bar{\boldsymbol{M}}_i$ being the i-th column of $\boldsymbol{M}$ and $\bar{\boldsymbol{M}}$ ($i = 1,\cdots,a$), respectively. Set a Lyapunov function as $\boldsymbol{V} = (\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)^{\mathrm{T}}(\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)$, and then, its time derivative is calculated as follows:

$$\begin{aligned}\dot{\boldsymbol{V}} &= (\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)^{\mathrm{T}}\dot{\boldsymbol{M}}_i - \alpha(\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)^{\mathrm{T}}(\boldsymbol{M}_i - \bar{\boldsymbol{M}}_i)\\ &= \boldsymbol{e}^{\mathrm{T}}\dot{\boldsymbol{M}}_i - \alpha\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}\\ &\le ||\boldsymbol{e}||_2||\dot{\boldsymbol{M}}_i||_2 - \alpha||\boldsymbol{e}||_2^2\\ &= ||\boldsymbol{e}||_2(||\dot{\boldsymbol{M}}_i||_2 - \alpha||\boldsymbol{e}||_2),\end{aligned} \tag{8}$$

with $||\cdot||_2$ being the Euclidean norm of a vector. The above equation leads to three different situations as follows:

- Situation i: $||\boldsymbol{e}||_2 > ||\dot{\boldsymbol{M}}_i||_2/\alpha$. This situation contributes to $\dot{\boldsymbol{V}} < 0$ and $\boldsymbol{V} > 0$, which implies that $||\boldsymbol{e}||_2$ is convergent until $||\boldsymbol{e}||_2 = ||\dot{\boldsymbol{M}}_i||_2$.
- Situation ii: $||\boldsymbol{e}||_2 = ||\dot{\boldsymbol{M}}_i||_2/\alpha$. It leads to $\dot{\boldsymbol{V}} \le 0$. This suggests that $||\boldsymbol{e}||_2$ converges to zero or maintains at $||\boldsymbol{e}||_2 = ||\dot{\boldsymbol{M}}_i||_2$.
- Situation iii: $||\boldsymbol{e}||_2 < ||\dot{\boldsymbol{M}}_i||_2/\alpha$. We deduce that $\dot{\boldsymbol{V}} > 0$ or $\dot{\boldsymbol{V}} \le 0$. Subsequently, it can be inferred that $||\boldsymbol{e}||_2$ continues to increase until it reaches Situation ii or it remains unchanged or convergent.

Considering the above three situations, it can be obtained that $||\boldsymbol{e}||_2 \le ||\dot{\boldsymbol{M}}_i||_2/\alpha$ when $t \to \infty$. Provided a sufficiently large value of α, we have that $||\boldsymbol{e}||_2$ reaches zero when $t \to \infty$. In conclusion, the estimated error $\boldsymbol{M} - \bar{\boldsymbol{M}}$ generated by Equation 7b globally converges to a zero matrix. Hence, applying LaSalle's invariance principle (K.Khalil, 2001), we replace $\bar{\boldsymbol{M}}$ with $\boldsymbol{M}$ in Equation 7b and deduce

$$\boldsymbol{\tau}_{\mathrm{out}} = \boldsymbol{M}(\boldsymbol{q})(\boldsymbol{L}^{\dagger}(\ddot{\boldsymbol{r}}_{\mathrm{d}} - \dot{\boldsymbol{L}}\dot{\boldsymbol{q}}) - \boldsymbol{v}) + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}). \tag{9}$$

Therefore, Equation 7b enables dynamic control of the manipulator depending on the desired joint acceleration in kinematic controller (Equation 3).

The desired joint acceleration allows the manipulator to precisely follow a given trajectory, which is proven through

10.3389/fnbot.2024.1451924

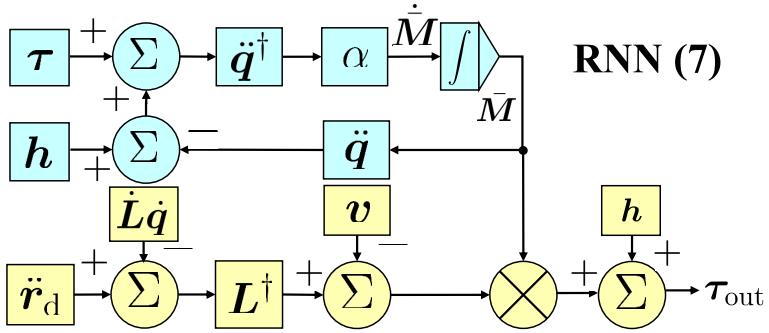


FIGURE 1
A control flow chart of RNN (Equation 7).

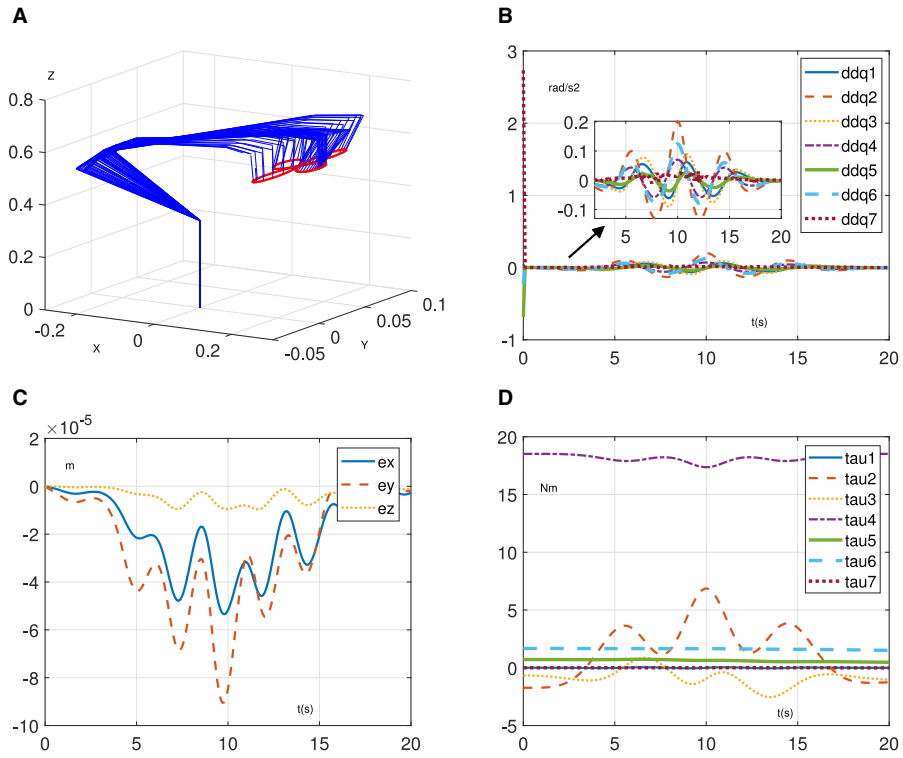


FIGURE 2
Simulative results of RNN (Equation 7) for trajectory-tracking task on Franka Emika Panda manipulator. **(A)** Motion process. **(B)** Joint acceleration. **(C)** Position error. **(D)** Joint torque.

the following proof. Primarily, (Equation 3) can be equivalently converted into

$$\boldsymbol{L}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{L}}\dot{\boldsymbol{q}} - \ddot{\boldsymbol{r}}_{\mathrm{d}} = \ddot{\boldsymbol{r}} - \ddot{\boldsymbol{r}}_{\mathrm{d}} = -\beta(\dot{\boldsymbol{r}} - \dot{\boldsymbol{r}}_{\mathrm{d}}) - \zeta(\boldsymbol{r} - \boldsymbol{r}_{\mathrm{d}}). \tag{10}$$

Assuming the position error as $\boldsymbol{u} = \boldsymbol{r} - \boldsymbol{r}_{\mathrm{d}}$, the above equation is reorganized as $\ddot{\boldsymbol{u}} + \beta\dot{\boldsymbol{u}} + \zeta\boldsymbol{u} = 0$, which belongs to a second-order differential equation system. The roots of the corresponding characteristic equation are $s_1 = (-\beta + \sqrt{\beta^2 - 4\zeta})/2$ and $s_2 = (-\beta - \sqrt{\beta^2 - 4\zeta})/2$. Furthermore, According to Equations 8–10, we can analyze that the convergence of this system can be categorized into the three cases (Jin et al., 2017).

- Case i: When $\beta^2 - 4\zeta > 0$, we obtain that $s_1 < 0$ and $s_2 < 0$ are real numbers with $s_1 \neq s_2$. Then, the solution satisfies $\boldsymbol{u}(t) = \boldsymbol{c}_1\exp(s_1 t) + \boldsymbol{c}_2\exp(s_2 t)$ with $\boldsymbol{c}_1 \in \mathbb{R}^b$ and $\boldsymbol{c}_1 \in \mathbb{R}^b$ being coefficient vectors determined by the initial state of the system.
- Case ii: When $\beta^2 - 4\zeta = 0$, the system has two equivalent characteristic roots with $s_1 = s_2 < 0$. Therefore, the solution can be deduced as $\boldsymbol{u}(t) = (\boldsymbol{c}_1 + \boldsymbol{c}_2)\exp(s_1 t)$.
- Case iii: When $\beta^2 - 4\zeta < 0$, $s_1 = z + iy$ and $s_2 = z - iy$ are conjugate complex numbers with $z < 0$. As a result, the solution can be deduced as $\boldsymbol{u}(t) = \exp(zt)(\boldsymbol{c}_1\cos yt + \boldsymbol{c}_2\sin yt)$.

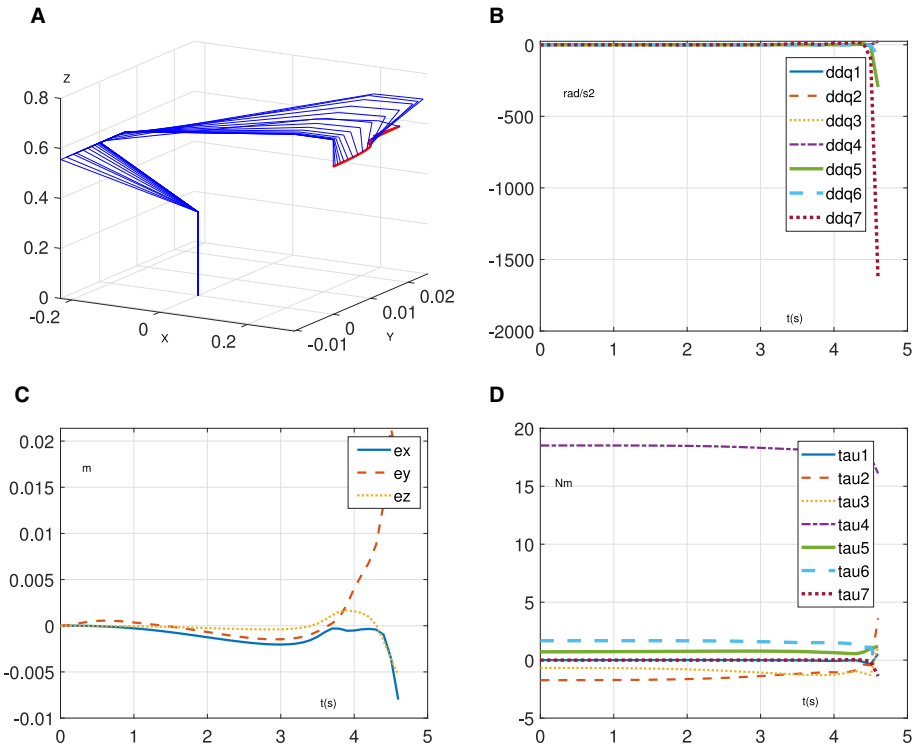


FIGURE 3
Simulative results of Equation 7b without Equation 7a for trajectory-tracking task on Franka Emika Panda manipulator. **(A)** Motion process. **(B)** Joint acceleration. **(C)** Position error. **(D)** Joint torque.

These cases demonstrate that the position error $\boldsymbol{u}$ generated by Equation 3 exponentially converges to a zero vector from any initial states. In other words, it is concluded that Equation 7b enables trajectory tracking control of the manipulator with the unknown mass matrix. The proof is thus completed.

5 Simulative results and comparisons

This section provides simulation experiments to demonstrate the learning and control performance of RNN (Equation 7). Specifically, we test it on a 7-degree-of-freedom manipulator called Franka Emika Panda (Liu and Shang, 2024) to task a four-leaf clover path with $\alpha = 10^4$, $\beta = 1$, and $\zeta = 5$. In addition, we assume that the mass matrix is unknown and design a random noise matrix with elements <0.5 to represent its uncertainties. The related results are shown in Figures 2, 3. Figure 2A demonstrates the effectiveness of the proposed method in enabling the manipulator to accomplish trajectory-tracking tasks, even in the presence of an unknown mass matrix. In addition, the initial joint acceleration in Figure 2B is relatively large due to the initial mass matrix error and becomes smooth and normal. Furthermore, the position error keeps the order of 10^{-5} m in Figure 2C. Similarly, in Figure 2D, it can be observed that the joint torque exhibits reasonable variations. However, when the estimation Equation 7b is not considered, achieving the trajectory tracking task based on Equation 7b becomes challenging due to the presence of the unknown mass matrix. As shown in Figure 3A, the manipulator driven by Equation 7a cannot complete the tracking task. Evidently, the joint acceleration became uncontrollable at ~4.5 s, as shown in Figure 3B, and the manipulator system is no longer operational. Furthermore, the position error exhibits divergence in Figure 3C. Similarly, the joint torque in Figure 3D is out of control at 4.5 s. Through the above results, the learning and control ability of the proposed method are verified.

To further demonstrate the feasibility of the proposed method, we additionally apply the proposed RNN (Equation 7) to control the Franka Emika Panda manipulator performing a Lissajous trajectory-tracking task. It is noteworthy that the parameters involved are identical to the previous simulation, except for the trajectory-tracking task. The specific results are presented in Figure 4. Specifically, Figures 4A, B demonstrate that the manipulator successfully executes the given trajectory tracking task, taking into account dynamics uncertainties. Furthermore, the positional error, as shown in Figure 4C, is maintained at the order of 10^{-5} m. Additionally, the joint acceleration exhibits normal variations, as shown in Figure 4D. In addition, Figures 4E, F illustrate that the proposed method is capable of compensating for the dynamics uncertainties, with tiny estimated errors of the joint torque. The aforementioned results indicate the effectiveness of the proposed RNN (Equation 7).

In addition, the advantages of the proposed RNN are shown in Table 1, compared with the existing methods. One notable advantage of the proposed RNN (Equation 7) is its simultaneous consideration of both the kinematic and dynamic models. This approach enables the realization

10.3389/fnbot.2024.1451924

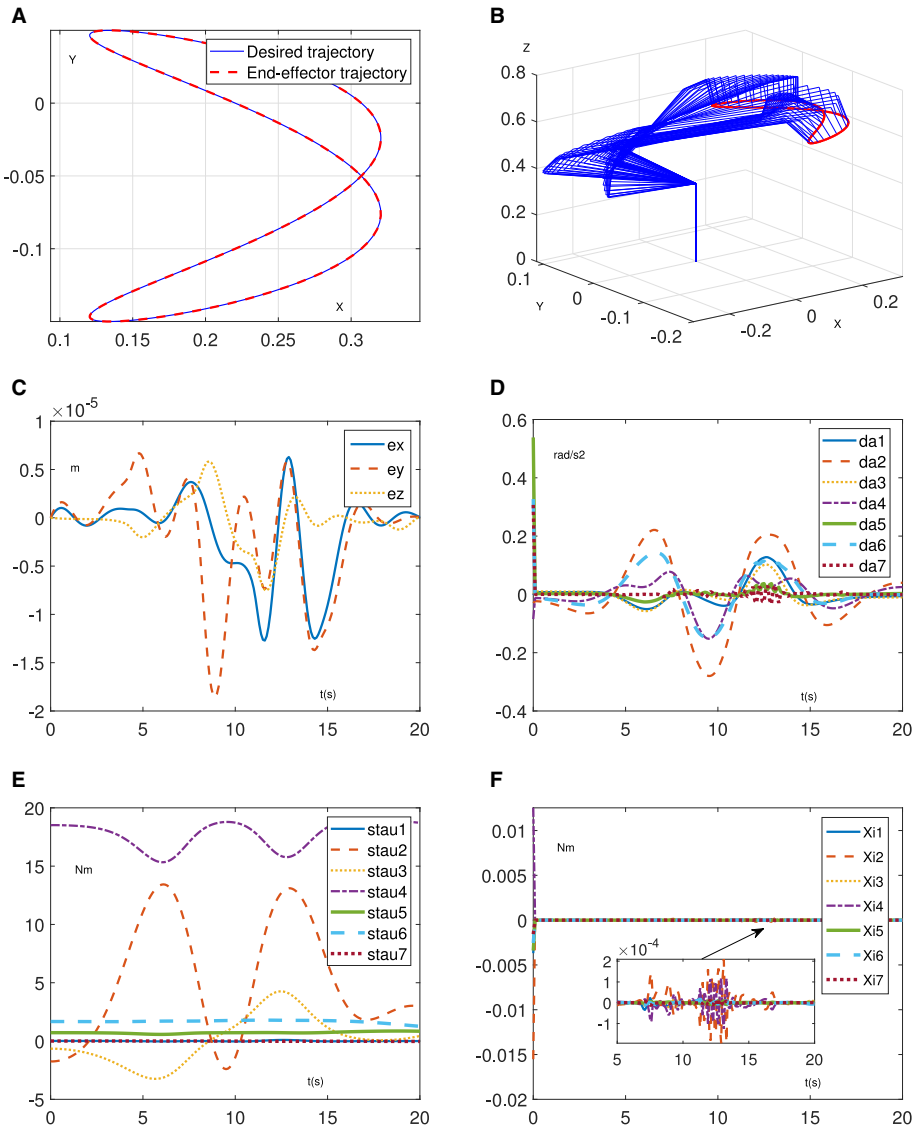


FIGURE 4
Simulative results of RNN (Equation 7) for trajectory-tracking task on Franka Emika Panda manipulator. **(A)** Desired trajectory and end-effector trajectory. **(B)** Motion process of manipulator. **(C)** Position error. **(D)** Joint acceleration. **(E)** Estimated joint torque. **(F)** Joint torque error with $\xi = \tau - \tilde{\tau}$.

TABLE 1 Comparisons among different methods for controlling manipulator.

Different methods	Dynamic control	Unknown issue	Trajectory tracking	Mass matrix Online estimation
RNN (Equation 7)	Yes	Yes	Yes	Yes
Guo et al. (2018)	No	No	Yes	No
Jin et al. (2023)	No	No	Yes	No
Ma et al. (2023b)	No	No	Yes	No
Jin et al. (2017)	No	No	Yes	No
Liu and Shang (2024)	No	No	Yes	No
Shojaei et al. (2021)	Yes	Yes	No	No
Zong and Emami (2021)	Yes	No	No	No

of online estimation of the mass matrix and synchronous trajectory tracking.

6 Conclusion

In this study, we have proposed a recurrent neural network (RNN) to address the challenges of trajectory tracking in manipulator systems with unknown mass matrices. The key idea of our proposed RNN is to establish a connection between the kinematics and dynamics models using joint acceleration signals, considering the motion characteristics and dynamic behavior of manipulators. Primarily, it has incorporated a kinematic controller to generate the desired joint acceleration based on the given task. On this basis, the robot dynamics model and a mass matrix estimator have been designed and integrated into the RNN to enable trajectory tracking in the presence of an unknown mass matrix. Subsequently, theoretical analysis has demonstrated the learning and control capabilities of the RNN. Through simulation experiments and comparisons, we have validated the effectiveness and superiority of the proposed RNN for trajectory tracking control of the manipulator with unknown mass matrix.

In addition to the robot's mass matrix, other dynamic parameters of the manipulator, such as the gravity vector, may also change. In addition, joint constraints help to improve the safety of robot operation. Therefore, future research will focus on estimating multiple dynamic parameters and considering multiple levels of joint constraints.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

JL: Funding acquisition, Investigation, Methodology, Project administration, Writing – review & editing, Supervision. JS: Data curation, Writing – original draft. HF: Methodology, Supervision, Writing – review & editing. WY: Methodology, Writing – review & editing. XM: Methodology, Writing – review & editing. ZL: Data curation, Methodology, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This research was funded by the Changchun Science and Technology Development Program, grant number: 21ZGN26 and by the Jilin Province Science and Technology Development Program, grant number: 20230508026RC.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Abate, A. F., Cimmino, L., Cuomo, I., Nardo, M. D., and Murino, T. (2022). On the impact of multimodal and multisensor biometrics in smart factories. *IEEE Trans. Industr. Informat.* 18, 9092–9100. doi: 10.1109/TII.2022.3178376

Bruder, D., Fu, X., Gillespie, R. B., Remy, C. D., and Vasudevan, R. (2021). Koopman-based control of a soft continuum manipulator under variable loading conditions. *IEEE Robot. Automat. Lett.* 6, 6852–6859. doi: 10.1109/LRA.2021.3095268

Cao, X., Peng, C., Zheng, Y., Li, S., Ha, T. T., Shutyaev, V., et al. (2023). Neural networks for portfolio analysis in high-frequency trading. *IEEE Trans. Neural Netw. Learn. Syst.* 2023:3311169. doi: 10.1109/TNNLS.2023.3311169

Guo, D., Xu, F., and Yan, L. (2018). New pseudoinverse-based path-planning scheme with PID characteristic for redundant robot manipulators in the presence of noise. *IEEE Trans. Contr. Syst. Technol.* 26, 2008–2019. doi: 10.1109/TCST.2017.2756029

Hua, C., Cao, X., Liao, B., and Li, S. (2023). Advances on intelligent algorithms for scientific computing: an overview. *Front. Neurorobot.* 17, 1–21. doi: 10.3389/fnbot.2023.1190977

Jin, L., Liu, L., Wang, X., Shang, M., and Wang, F. Y. (2024a). Physical-informed neural network for MPC-based trajectory tracking of vehicles with noise considered. *IEEE Trans. Intell. Vehicl.* 9, 4493–4503. doi: 10.1109/TIV.2024.3358229

Jin, L., Zhang, F., Liu, M., and Xu, S. S. D. (2023). Finite-time model predictive tracking control of position and orientation for redundant manipulators. *IEEE Trans. Industr. Electr.* 70, 6017–6026. doi: 10.1109/TIE.2022.3196372

Jin, L., Zhang, Y., Li, S., and Zhang, Y. (2017). Noise-tolerant ZNN models for solving time-varying zero-finding problems: a control-theoretic approach. *IEEE Trans. Automat. Contr.* 62, 992–997. doi: 10.1109/TAC.2016.2566880

Jin, L., Zhao, J., Chen, L., and Li, S. (2024b). Collective neural dynamics for sparse motion planning of redundant manipulators without hessian matrix inversion. *IEEE Trans. Neural Netw. Learn. Syst.* 2024:3363241. doi: 10.1109/TNNLS.2024.3363241

K.Khalil, H. (2001). *Nonlinear Systems, 3rd Edn*. Englewood Cliffs, NJ: Prentice Hall.

Lian, Y., Xiao, X., Zhang, J., Jin, L., Yu, J., and Sun, Z. (2024). Neural dynamics for cooperative motion control of omnidirectional mobile manipulators in the presence of noises: a distributed approach. *IEEE/CAA J. Automat. Sin.* 11, 1605–1620. doi: 10.1109/JAS.2024.124425

Liao, B., Hua, C., Xu, Q., Cao, X., and Li, S. (2023). A predefined-time and anti-noise varying-parameter ZNN model for solving time-varying complex stein equations. *Neurocomputing* 526, 158–168. doi: 10.1016/j.neucom.2023.01.008

Liao, B., Hua, C., Xu, Q., Cao, X., and Li, S. (2024). Inter-robot management via neighboring robot sensing and measurement using a zeroing neural dynamics approach. *Expert Syst. Appl.* 244:122938. doi: 10.1016/j.eswa.2023.122938

Liao, B., Wang, Y., Li, J., Guo, D., and He, Y. (2022). Harmonic noise-tolerant ZNN for dynamic matrix pseudoinversion and its application to robot manipulator. *Front. Neurorobot.* 16, 1–13. doi: 10.3389/fnbot.2022.928636

Liu, M., Li, Y., Chen, Y., Qi, Y., and Jin, L. (2024a). A distributed competitive and collaborative coordination for multirobot systems. *IEEE Trans. Mob. Comput.* 2024:3397242. doi: 10.1109/TMC.2024.3397242

Liu, M., Liu, K., Zhu, P., Zhang, G., Ma, X., and Shang, M. (2024b). Data-driven remote center of cyclic motion (RC^2M) control for redundant robots with rod-shaped end-effector. *IEEE Trans. Industr. Informat*. 20, 6772–6780. doi: 10.1109/TII.2024.3353930

Liu, M., and Shang, M. (2024). Orientation tracking incorporated multicriteria control for redundant manipulators with dynamic neural network. *IEEE Trans. Industr. Electr*. 71, 3801–3810. doi: 10.1109/TIE.2023.3273253

Ma, B., Jiang, Z., Liu, Y., and Xie, Z. (2023a). Advances in space robots for on-orbit servicing: a comprehensive review. *Adv. Intell. Syst*. 5, 1–21. doi: 10.1002/aisy.202200397

Ma, B., Xie, Z., Zhan, B., Jiang, Z., Liu, Y., and Liu, H. (2023b). Actual shape-based obstacle avoidance synthesized by velocity—acceleration minimization for redundant manipulators: an optimization perspective. *IEEE Trans. Syst. Man Cybernet*. 53, 6460–6474. doi: 10.1109/TSMC.2023.3283266

Müller, A., Kumar, S., and Kordik, T. (2023). A recursive lie-group formulation for the second-order time derivatives of the inverse dynamics of parallel kinematic manipulators. *IEEE Robot. Automat. Lett*. 8, 3804–3811. doi: 10.1109/LRA.2023.3267005

Shojaei, K., Kazemy, A., and Chatraei, A. (2021). An observer-based neural adaptive PID^2 controller for robot manipulators including motor dynamics with a prescribed performance. *IEEE/ASME Trans. Mechatr*. 26, 1689–1699. doi: 10.1109/TMECH.2020.3028968

Sun, Z., Tang, S., Jin, L., Zhang, J., and Yu, J. (2023a). Nonconvex activation noise-suppressing neural network for time-varying quadratic programming: application to omnidirectional mobile manipulator. *IEEE Trans. Industr. Informat*. 19, 10786–10798. doi: 10.1109/TII.2023.3241683

Sun, Z., Tang, S., Zhang, J., and Yu, J. (2023b). Nonconvex noise-tolerant neural model for repetitive motion of omnidirectional mobile manipulators. *IEEE/CAA J. Automat. Sin*. 10, 1766–1768. doi: 10.1109/JAS.2023.123273

Sun, Z., Wang, G., Jin, L., Cheng, C., Zhang, B., and Yu, J. (2022). Noise-suppressing zeroing neural network for online solving time-varying matrix square roots problems: a control-theoretic approach. *Expert Syst. Appl*. 192:116272. doi: 10.1016/j.eswa.2021.116272

Sun, Z., Xu, C., Gu, J., Zhao, L., and Hu, Y. (2024). Design, modeling and optimal control of a novel compliant actuator. *Contr. Eng. Pract*. 148:105967. doi: 10.1016/j.conengprac.2024.105967

Tang, Z., and Zhang, Y. (2022). Refined self-motion scheme with zero initial velocities and time-varying physical limits via zhang neurodynamics equivalency. *Front. Neurorobot*. 16, 1–15. doi: 10.3389/fnbot.2022.945346

Tang, Z., Zhang, Y., and Ming, L. (2024). Novel snap-layer MMPC scheme via neural dynamics equivalency and solver for redundant robot arms with five-layer physical limits. *IEEE Trans. Neural Netw. Learn. Syst*. 2024:3351674. doi: 10.1109/TNNLS.2024.3351674

Wen, L., and Xie, Z. (2024). A data-driven acceleration-level scheme for image-based visual servoing of manipulators with unknown structure. *Front. Neurorobot*. 18, 1–14. doi: 10.3389/fnbot.2024.1380430

Xiao, L., He, Y., Dai, J., Liu, X., Liao, B., and Tan, H. (2022). A variable-parameter noise-tolerant zeroing neural network for time-variant matrix inversion with guaranteed robustness. *IEEE Trans. Neural Netw. Learn. Syst*. 33, 1535–1545. doi: 10.1109/TNNLS.2020.3042761

Xiao, L., He, Y., Wang, Y., Dai, J., Wang, R., and Tang, W. (2023). A segmented variable-parameter ZNN for dynamic quadratic minimization with improved convergence and robustness. *IEEE Trans. Neural Netw. Learn. Syst*. 34, 2413–2424. doi: 10.1109/TNNLS.2021.3106640

Xiao, L., Zhang, Y., Liao, B., Zhang, Z., and Jin, L. (2017). A velocity-level Bi-criteria optimization scheme for coordinated path tracking of dual robot manipulators using recurrent neural network. *Front. Neurorobot*. 11, 1–7. doi: 10.3389/fnbot.2017.00047

Xie, Z., and Jin, L. (2024). A fuzzy neural controller for model-free control of redundant manipulators with unknown kinematic parameters. *IEEE Trans. Fuzzy Syst*. 32, 1589–1601. doi: 10.1109/TFUZZ.2023.3328545

Xie, Z., Jin, L., Luo, X., Hu, B., and Li, S. (2022). An acceleration-level data-driven repetitive motion planning scheme for kinematic control of robots with unknown structure. *IEEE Trans. Syst. Man Cybernet*. 52, 5679–5691. doi: 10.1109/TSMC.2021.3129794

Xie, Z., Li, S., and Jin, L. (2023). A Bi-criteria kinematic strategy for motion/force control of robotic manipulator. *IEEE Trans. Automat. Sci. Eng*. 2023:3313564. doi: 10.1109/TASE.2023.3313564

Xie, Z., Zheng, Y., and Jin, L. (2024). A data-driven image-based visual servoing scheme for redundant manipulators with unknown structure and singularity solution. *IEEE Trans. Syst. Man Cybernet*. 2024:3420882. doi: 10.1109/TSMC.2024.3420882

Yan, J., Liu, M., and Jin, L. (2024). Cerebellum-inspired model predictive control for redundant manipulators with unknown structure information. *IEEE Trans. Cogn. Dev. Syst*. 16, 1198–1210. doi: 10.1109/TCDS.2023.3340179

Zhang, Y., Li, S., Kadry, S., and Liao, B. (2019). Recurrent neural network for kinematic control of redundant manipulators with periodic input disturbance and physical constraints. *IEEE Trans. Cybernet*. 49, 4194–4205. doi: 10.1109/TCYB.2018.2859751

Zong, L., and Emami, M. R. (2021). Control verifications of space manipulators using ground platforms. *IEEE Trans. Aerospace Electr. Syst*. 57, 341–354. doi: 10.1109/TAES.2020.3016877

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 27 September 2024
DOI 10.3389/fnbot.2024.1448482

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Mario Versaci,
Mediterranea University of Reggio Calabria, Italy
Puchen Zhu,
The Chinese University of Hong Kong, China

*CORRESPONDENCE
Junwei Yin
✉ yjwzx@djtu.edu.cn

RECEIVED 13 June 2024
ACCEPTED 02 September 2024
PUBLISHED 27 September 2024

CITATION
Ma F, Wang H, E M, Sha Z, Wang X, Cui Y and Yin J (2024) Fast reconstruction of milling temperature field based on CNN-GRU machine learning models.
Front. Neurorobot. 18:1448482.
doi: 10.3389/fnbot.2024.1448482

Fast reconstruction of milling temperature field based on CNN-GRU machine learning models

Fengyuan Ma[1], Haoyu Wang[1], Mingfeng E[1], Zhongjin Sha[2], Xingshu Wang[1], Yunxian Cui[1] and Junwei Yin[1]*

[1]School of Mechanical Engineering, Dalian Jiaotong University, Dalian, China, [2]Angang Heavy Machinery Co., Ltd, Anshan, China

With the development of intelligent manufacturing technology, robots have become more widespread in the field of milling processing. When milling difficult-to-machine alloy materials, the localized high temperature and large temperature gradient at the front face of the tool lead to shortened tool life and poor machining quality. The existing temperature field reconstruction methods have many assumptions, large arithmetic volume and long solution time. In this paper, an inverse heat conduction problem solution model based on Gated Convolutional Recurrent Neural Network (CNN-GRU) is proposed for reconstructing the temperature field of the tool during milling. In order to ensure the speed and accuracy of the reconstruction, we propose to utilize the inverse heat conduction problem solution model constructed by knowledge distillation (KD) and compression acceleration, which achieves a significant reduction of the training time with a small loss of optimality and ensures the accuracy and efficiency of the prediction model. With different levels of random noise added to the model input data, CNN-GRU + KD is noise-resistant and still shows good robustness and stability under noisy data. The temperature field reconstruction of the milling tool is carried out for three different working conditions, and the curve fitting excellence under the three conditions is 0.97 at the highest, and the root mean square error is 1.43°C at the minimum, respectively, and the experimental results show that the model is feasible and effective in carrying out the temperature field reconstruction of the milling tool and is of great significance in improving the accuracy of the milling machining robot.

KEYWORDS

temperature field reconstruction, gated convolutional neural networks, knowledge distillation, inverse heat transfer, milling

1 Introduction

In the era of Industry 4.0, China's manufacturing industry is undergoing a profound transformation, and the use of robotics is becoming increasingly important in intelligent manufacturing. Intelligent manufacturing relies on multifunctional sensors to perceive the production environment (Cheng et al., 2016; Javaid et al., 2021). Production equipment autonomously learns through sensor-based and data-driven methods. This enables adaptive machining in changing environments. Ultimately, intelligent control achieves the desired outcomes (Lee et al., 2015). As one of the important machining methods in the manufacturing

industry, milling processing has a broad prospect for the use of robots. In the use of robots, the most important issue is the processing quality and processing accuracy. In milling machining, localized high temperatures and strong time-varying temperature gradients are mainly concentrated at the boundary of the tool heat transfer system, i.e., the cutting region. Localized high temperatures impact tool life and can stimulate the chemical activity of the material being removed. This leads to material oxidation, rapid corrosion, and adhesion and diffusion between the material and the tool. Consequently, these effects degrade the machining accuracy and quality of the workpiece (Korkmaz and Gupta, 2024). In addition, the high temperature of the cutting area will also cause localized thermal deformation of the tool tip, which is one of the main reasons for the reduction of machining accuracy. Therefore, the localized high temperature in the cutting area and the strong time-varying temperature gradient will lead to the shortening of tool life, the reduction of workpiece machining quality, and the reduction of machining efficiency.

Due to the interference of cutting fluid and chips, existing sensing technology cannot directly measure the temperature field inside the cutting area (Alammari et al., 2024). Sensors can only be placed near the tool-chip contact area to obtain limited temperature data outside the cutting zone. Metal cutting is a thermodynamic coupling process with significant changes in material elastic–plastic deformation and contact area friction. These changes cause strong non-uniformity in the tool temperature field over time and space. A single or a few measurement points cannot accurately describe the actual processing conditions. Therefore, studying tool temperature field reconstruction during milling is crucial for extending tool life and improving machining accuracy.

Currently, there is still some difficulty in accurately measuring the temperature field online for the cutting region, and physical methods based on infrared thermography, artificial thermocouples and embedded thermocouples (Cichosz et al., 2023; Leonidas et al., 2022; Longbottom and Lanham, 2005) can only measure a limited number of *in-situ* temperatures near the cutting region or the approximate temperature field close to the location of the cutting region. In recent years, computational reconstruction methods for modeling the temperature field of tools have gained widespread attention. These methods bypass physical limitations to obtain temperature data at any location of interest. Current modeling techniques are mainly categorized into analytical modeling, numerical simulation based on cutting mechanisms, and inverse heat conduction modeling, which combines physical measurements with model-solving methods. The inverse heat conduction problem (IHCP) is part of the "mathematical physics inverse problem" field. A positive problem in physics research can be described by mathematical equations, where given equations and parameters, the output can be determined from a known input. Early studies simplified tool models and the cutting process, often treating the tool as a one- or two-dimensional model and the cutting process as steady-state. In these cases, analytical methods were combined with IHCP to directly compute mathematical expressions for the relationship between unknown quantities and measured values (Murio, 1981).

Nowadays, more and more researchers are focusing on reducing the complex three-dimensional structure of the tool with transient cutting process (Oommen and Srinivasan, 2022), for example, Some scholars (Liang et al., 2013) proposed a three-dimensional inverse heat transfer model based on an improved conjugate gradient method, which can quantitatively calculate the temperature of the tool chip contact area in dry turning. Some other researchers (Carvalho et al., 2006) used the golden section iterative method to solve the inverse heat conduction problem, and used the finite volume method to construct a three-dimensional model of the turning tool, which takes into account the thermal properties of the material as affected by temperature as well as the convective heat transfer losses to realize the temperature field reconstruction calculation.

In recent years, with the advancement of artificial intelligence algorithms and machine learning technology, artificial neural network models based on data relations have been widely used in inverse thermal problem solving. For example, the application of algorithms such as physical information neural network (PINN; Qian et al., 2023), nonlinear autoregressive exogenous input neural network (NARX; Chen and Pan, 2023), convolutional neural network (CNN; Kim and Lee, 2020), and multidomain physical information neural network (M-PINN; Zhang et al., 2022), etc., has made a certain contribution to the solution of the inverse heat conduction problem. Researchers (Zhang and Wang, 2024) have used deep neural networks to characterize and approximate partial differential equations (PDEs) in the forward problem style. They proposed an optimization algorithm that uses sequence-to-sequence (Seq2Seq) stacking with the gated recurrent unit (GRU) model. It improves the solving of these equations by stacking GRU modules to capture their evolution over time. It also has strong generalization ability.

There is still a wide range of prospects for the fusion of artificial neural network models. For example, CNN struggles to capture temporal features, while GRU struggles to capture spatial features. Combining CNN and GRU might allow their strengths to complement each other, enabling the CNN-GRU model to effectively capture spatio-temporal features, thereby improving the model's accuracy and generalization performance.

This paper proposes a CNN-GRU based milling tool heat transfer model with knowledge distillation compression acceleration. The model reconstructs the milling tool temperature field under three different working conditions. A self-built milling temperature data acquisition system collects real-time temperature data from multiple points on the back face of the milling cutter. This system uses a temperature measurement tool embedded with a thin-film thermocouple array and a multi-channel signal acquisition device. By analyzing the relationship between machining parameters, the temperature at four measurement points on the milling tool, and the temperature in the cutting area, we use machining parameters and multi-point temperatures as input features. The temperature boundary conditions in the cutting area serve as prediction labels. The GRU is introduced to the convolutional neural network (CNN) to extract multi-dimensional feature information, aiming to improve reconstruction accuracy and efficiency. We then apply a knowledge distillation strategy to compress and accelerate the CNN-GRU model. This approach reduces computation time while maintaining high prediction performance and accuracy, ensuring efficient temperature field reconstruction.

The rest of this study is organized as follows: section 2 reviews the work related to this study, section 3 describes the proposed method in detail, section 4 reports the experimental results and analysis, and section 5 concludes this study.

The contributions of this paper are as follows:

1 CNN-GRU-based solution model for inverse heat transfer problem: a solution model for inverse heat transfer problem based on convolutional gated recurrent network (CNN-GRU) to predict the temperature boundary conditions in the cutting region of the tool is proposed. The model can well utilize the machining parameters in milling processing, the characteristics of the multi-point temperature of the milling tool back face, thus significantly improving the accuracy and efficiency of the milling temperature field reconstruction.
2 KD compression-accelerated model for solving inverse heat conduction problem: the constructed CNN-GRU model is compressed and accelerated using the knowledge distillation strategy. Compared with the model without KD acceleration, the model can substantially accelerate the training time with the least loss of goodness-of-fit, and has strong noise immunity.
3 A transient heat conduction model of milling tool is constructed, and the temperature field reconstruction of milling tool is carried out for three different working conditions, and the tests under the three working conditions are carried out in order to check its application ability in the reconstruction of milling temperature field.

2 Related work

Currently, deep learning techniques have been successfully applied to real-world scenarios, solving challenging problems like predicting the lifetime of relays and batteries. Constructing prediction models with artificial neural networks has broad applications in solving inverse heat transfer problems (Cortés et al., 2007; Wang et al., 2023; Kamyab et al., 2022). Additionally, methods for compressing and accelerating deep learning models have enhanced the efficiency and applicability of these techniques in real-time temperature field reconstruction. Integrating these advanced algorithms significantly improves the precision and speed of temperature field predictions during milling. This makes them indispensable for optimizing machining operations.

2.1 Shallow artificial neural network approach

Shallow artificial neural network methods were among the first techniques applied to the solution of inverse heat transfer problems. These methods utilize a simple hierarchical structure for data processing and prediction by simulating the way neurons in the brain work. Despite the simplicity of their structure, shallow neural networks have demonstrated their effectiveness and feasibility in solving specific problems, such as in the area of predicting lifespan.

Combining Back Propagation Neural Networks (BPNNs) with time-series data analysis methods has been utilized to predict the remaining life of cooling fans (Lixin et al., 2016). Based on the time series data analysis of historical data information to obtain the future trend of the data, the prediction error is adjusted using BPNN to ensure the accuracy of the prediction results. A single BPNN will face the problem of weight local optimization, i.e., overfitting, during training, and in recent years a large number of scholars have combined BPNN with other machine learning methods to improve the model accuracy.

Radial Basis Function Networks (RBFNs) are widely used in various fields due to their advantages of having outputs independent of initial weights and shorter training times. A gray RBFN-based prediction model (Li et al., 2009) for life and reliability of constant stress accelerated life testing has been developed and compared with traditional single Backpropagation Neural Networks (BPNNs). Experimental results demonstrate that the accuracy of the gray RBFN model surpasses that of the BPNN.

The shallow artificial neural network model has a high dependence on large-scale data, and the shallow model is prone to overfitting phenomenon during the training process, especially when the training data is small or the data dimension is high. In practice, the sensitivity of shallow artificial neural networks to data quality and noise may lead to a decrease in the robustness of the model.

2.2 Deep artificial neural network methods

With the improvement of computational power and the development of deep learning technology, deep artificial neural network methods have demonstrated powerful performance in solving complex problems. Deep neural networks are able to better capture complex patterns and higher-order features in the data through multilayer nonlinear transformations, which significantly improves the predictive ability of the model.

A deep learning method combining sparse stacked self-encoders (Stacked Sparse AEs, SSAEs) with Backpropagation Neural Networks (BPNNs) has been proposed (He et al., 2021). This method uses tool temperature measurements from temperature sensors to predict tool wear. When compared to BPNN and SVM models that rely on manually extracted time-frequency domain features, this approach demonstrates high prediction accuracy and stability.

A time window method for obtaining samples and a multivariate equipment life prediction method based on deep Convolutional Neural Networks (CNNs) have been proposed (Li et al., 2018), focusing on feature extraction. To avoid filtering out effective information by the pooling layer, the pooling layer was removed when constructing the network model. Additionally, a deep CNN method for bearing residual life prediction has been introduced (Ren et al., 2018), which combines spectral principal energy vectors into a feature map. This method extracts one-dimensional vectors and inputs them into the deep learning model through a multilayer CNN structure, demonstrating that its prediction accuracy meets the required standards.

Furthermore, the problem of predicting the remaining life of batteries using deep learning has been explored (Zhang Y. et al., 2018). Long Short-Term Memory (LSTM) networks are used to learn the long-term dependencies between the capacity degradation of lithium-ion batteries. LSTM employs backward error propagation for adaptive optimization and uses the dropout regularization technique to address the overfitting problem. This method exhibits better learning and generalization abilities compared to support vector machines and traditional recurrent neural networks.

The deep artificial neural network method has high accuracy for prediction problems such as lifetime prediction, but there is still a lot of room for improvement in efficiency, and there are some limitations

in the application of inverse heat conduction solving problems and temperature field reconstruction.

2.3 Deep learning model compression and acceleration method

Neural network pruning is an important method to achieve network model compression and acceleration, and its working principle is mainly to cut off the weights and model branches that are not important when the neural network is working, to get a small model, from achieving the compression and acceleration of the model.

The ThiNet pruning method (Luo et al., 2017) differs from traditional pruning methods by treating network pruning as a reconstruction optimization problem. This approach determines the pruning strategy for the convolutional kernel of the current layer based on statistical information computed from the reconstruction differences between the inputs and outputs of the subsequent layer.

In addition to network pruning, other lightweight network design methods have been developed. Group point-by-point convolution (Zhang X. et al., 2018) performs grouped convolution operations to reduce the computational loss associated with point-by-point convolution operations. To enable grouped convolution to capture features computed by other groups, a mixing operation is introduced to reintegrate features from different groups, allowing the new group to contain features from other groups as well.

Automatic machine learning algorithms (AutoML) have also been widely used in lightweight neural network design. Some researchers (He et al., 2018) proposed AutoML for Model Compression (AMC), which utilizes reinforcement learning to efficiently sample the design space and learn compression strategies with better compression ratios to maintain model performance while reducing human intervention in the model. and maintain model performance while reducing human intervention in the model.

Due to the large model capacity difference between the teacher model and the student model, which leads to a "generation gap" between the student model and the teacher model, Wang Y. et al. (2018) pioneered a teacher-assistant-assisted knowledge distillation method, which utilizes the discriminator of the generative adversarial network as the teacher-assistant. They regarded the student model as a generator, and guided by the discriminator, the student model generated a feature distribution similar to that of the teacher's model, thus assisting the student model in learning. Some researchers (Cui et al., 2017) proposed a novel mutual distillation method, which allowed two groups of untrained student models to start learning and solve the task together, i.e., the teacher and the student models were trained and updated at the same time.

According to the above findings, knowledge distillation, a deep learning model compression and acceleration strategy, has been widely applied and developed, but little research has been reported on the application of knowledge distillation techniques in the field of heat conduction inverse problem solving.

2.4 Temperature field reconstruction

Temperature field reconstruction is a key step in solving inverse heat transfer problems, through which accurate reconstruction of the temperature field can lead to a better understanding of the heat transfer process and improve the thermal performance of materials and devices. In recent years, temperature field reconstruction techniques combining advanced algorithms and neural network methods have made significant progress.

An enhanced Bayesian backpropagation neural network based on Kalman filtering has been proposed (Deng and Hwang, 2007), applying the Kalman filtering algorithm to improve the weak generalization ability of the backpropagation algorithm in approximating nonlinear functions. This enhancement improves the performance of the Bayesian backpropagation network in solving the inverse heat conduction problem, and it has been compared with backpropagation networks optimized using other mature algorithms, such as GMB and LMB.

In another study, the volumetric heat capacity function of solid materials with temperature has been solved using a backpropagation neural network combined with a radial basis function neural network based on full-history information (Czél et al., 2013). Some researchers (Wang H. et al., 2018) proposed a heat flux estimation algorithm based on a linear artificial neural network for identifying a finite shock response under a linear dynamic system.

In conclusion, temperature field reconstruction plays an important role in the solution of inverse heat conduction problems. By introducing neural networks and other intelligent algorithms, researchers have made many breakthroughs in improving the reconstruction accuracy and computational efficiency. These methods not only enrich the means of solving inverse problems theoretically, but also demonstrate a strong potential in practical applications, providing new ideas for the solution of complex heat conduction problems.

3 Methods

3.1 Acquisition of data sets

In metal cutting, the temperature of the tool is mainly affected by the integrated heat source of the three deformation zones, in which the heat is mainly transferred to the tool through the cutting area, and

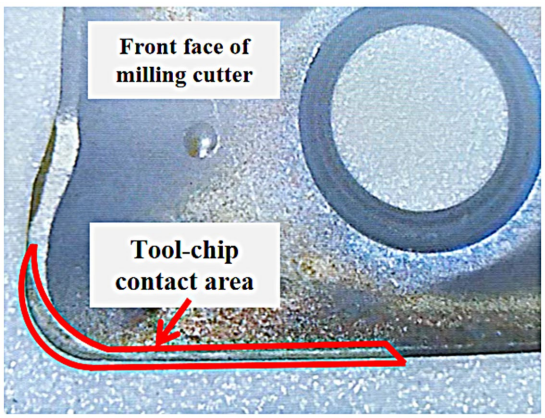


FIGURE 1
Image of the cutting area of the tool with radial depth of cut (ae) of 0.2 mm and axial depth of cut (ap) of 8 mm.

the cutting area of the tool can be regarded as the boundary of the tool heat conduction system. The cutting region of the tool generally includes: the tool-chip contact region and the tool-worker contact region, when the tool back angle is large, the cutting time is short and the cutting speed is small, and the back face of the tool does not undergo intense wear, the tool-worker contact region can be regarded as a part of the tool-chip contact region (Jaspers et al., 1998), and at this time, the tool-chip contact region is the tool's cutting region. The front face of the tool can be photographed using an electron microscope, and the wear area of the main cutting edge attachment is the tool-chip contact area. Figure 1 shows the image of the cutting area of the tool with radial depth of cut (ae) of 0.2 mm and axial depth of cut (ap) of 8 mm.

In this test, a temperature measuring tool embedded with a thin-film thermocouple (TFTC) developed by this group was used for end milling Inconel 718 nickel-based high-temperature alloy workpiece, and according to the requirements of the test, the size of the workpiece was designed to be 50 mm × 20 mm × 10 mm. In the design of the test for end milling Inconel 718, the comprehensive consideration of the theory of heat transfer of metal cutting was taken into account, and the spindle speed (r/min), feed rate (mm/min), and radial milling depth (mm), which have an important influence on milling temperature, were taken as test variable factors. The spindle speed (r/min), feed rate (mm/min), and radial milling depth (mm), which have an important influence on the milling temperature, are taken as the test variable factors. After determining the test variables, a full factorial design of experiments (DOE) was used to ensure that all levels of each test variable were tested at least once. Figure 2 shows a physical diagram of a transient milling multi-point temperature measurement toolholder.

The end milling test was conducted using the constructed test platform and test program, and the temperature data corresponding to the four temperature measurement points of the milling tool were recorded and saved. According to the actual processing requirements, when reconstructing the temperature field of the milling process tool, only the temperature field reconstruction of the tool during the cutting process needs to be considered, without the need to reconstruct the temperature field of the retracting process after the completion of machining, so this paper in the subsequent processing of data in the process of selecting the cut in to the cut out of the retracting tool before the start of the temperature reduction moment to be recorded. In the end milling process, the tool is often accompanied by violent vibration, so the collected temperature data will have a certain noise level, and the data need to be filtered.

FIGURE 2
Milling multi-point temperature measurement toolholder.

The inverse heat conduction problem of the tool heat conduction model refers to the fact that one of the parameters in the control equations, initial conditions, thermophysical parameters, and all boundary conditions of the tool heat conduction is in a missing state, and the unknown parameters need to be solved in reverse by measuring the physical signals by other methods. The inverse heat conduction problem in this paper belongs to the first type of margin estimation inverse problem, where the temperature on the boundary of the tool heat conduction system is estimated from the temperature sensor measurement results. The temperature on the boundary of the tool heat transfer system cannot be measured by physical methods or the measurement accuracy is poor, and is generally obtained using simulation methods. By constructing a local numerical simulation model of the milling process, the Inconel 718 end milling simulation model is operated and set up in complete control of the full factorial test parameters and machining time, and the simulation model is adjusted and corrected by the results of the actual sensor measurements and comparison of the chip morphology. The test simulation was completed using a cutting model with the required accuracy to obtain temperature data on the boundary of the tool heat transfer system.

The simulation model is adjusted and calibrated according to the test results, and after the accuracy meets the requirements, the average temperature of the cutting area of the tool is derived from the cloud diagram of the simulation results, and the other 26 sets of temperature curves can also be obtained by the simulation model, which provides the data sample set of the training model for the subsequent inverse heat conduction problem solving.

3.2 Construction of gated convolutional recurrent network model

The traditional one-dimensional CNN may ignore the time series features in the input data, resulting in the loss of some important time series information, in order to solve this problem, GRU can be introduced on the basis of one-dimensional CNN to simultaneously extract the multi-dimensional feature information as well as the temporal characteristics of the time series. Gated Convolutional Recurrent Neural Network (CNN-GRU) is a kind of neural network that combines the features of both CNN and GRU models, and is usually used to process time-series data, text, speech, and video, etc. The workflow of CNN-GRU is firstly, the input data undergoes a series of Convolution and pooling operations to extract the spatial dimension information in the data, and then the local features after the convolution operation are input into the GRU for sequence modeling, the GRU will dynamically update the hidden state according to the feature sequences of the input data, obtaining the long-term dependencies in the input data, and perform the

task of output prediction according to the hidden state of the GRU network, and the final output is performed through a fully connected layer. Using CNN-GRU as a model for solving the inverse heat transfer problem can directly establish the nonlinear data relationship between the machining parameters, the temperature of the back face of the milling cutter and the temperature of the cutting area. The CNN effectively captures the spatial information and combines with the GRU network to model the long-term dependence in the sequence to realize the rapid solution of the nonlinear inverse heat transfer problem.

The prediction process based on the CNN-GRU model is shown in Figure 3 as follows:

(1) Preprocess the original dataset with data normalization and dataset division;
(2) Construct the CNN-GRU model;
(3) Use the validation set to verify the model accuracy and save the model with the required accuracy;
(4) Test the CNN-GRU model with the test set to obtain the final temperature prediction results on the cutting area of the tool.

As the original data set milling temperature and machining parameters and other types of data have different scales and value ranges, which makes some features weight update process will be affected by the larger and ignore some other features, normalization can eliminate this effect so that all features have the same scale. In addition, in this case, the difference in the scale of the features will affect the training and convergence of the model, if there is a large difference in the scale of the features, then the step size of the update in the gradient descent process may be affected by the difference in the size of the gradient, which will lead to a slower convergence speed. By normalization, the direction of gradient descent can be made consistent, accelerating the convergence speed of the model. There are many methods of normalization such as Min-Max Scaling, Z-score Standardization, Softmax Normalization etc. According to the data type choose the Min-Max Scaling method for data normalization, which is a common normalization method to scale the data to between [−1,1], the formula is shown as Equation (1):

$$x^* = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{1}$$

where x^* represents the normalized data, x_i represents the observed value at moment i, and x_{min} and x_{max} are the minimum and maximum values in the data, respectively.

The predictions of the model on the test set are restored after the model training using inverse normalization, which is formulated as Equation (2):

$$x = x^*\left(x_{\max} - x_{\min}\right) + x_{\min} \tag{2}$$

where x inverse normalized value, x^* normalized value of the prediction result, and, x_{min}, x_{max} are the minimum and maximum values in the data, respectively.

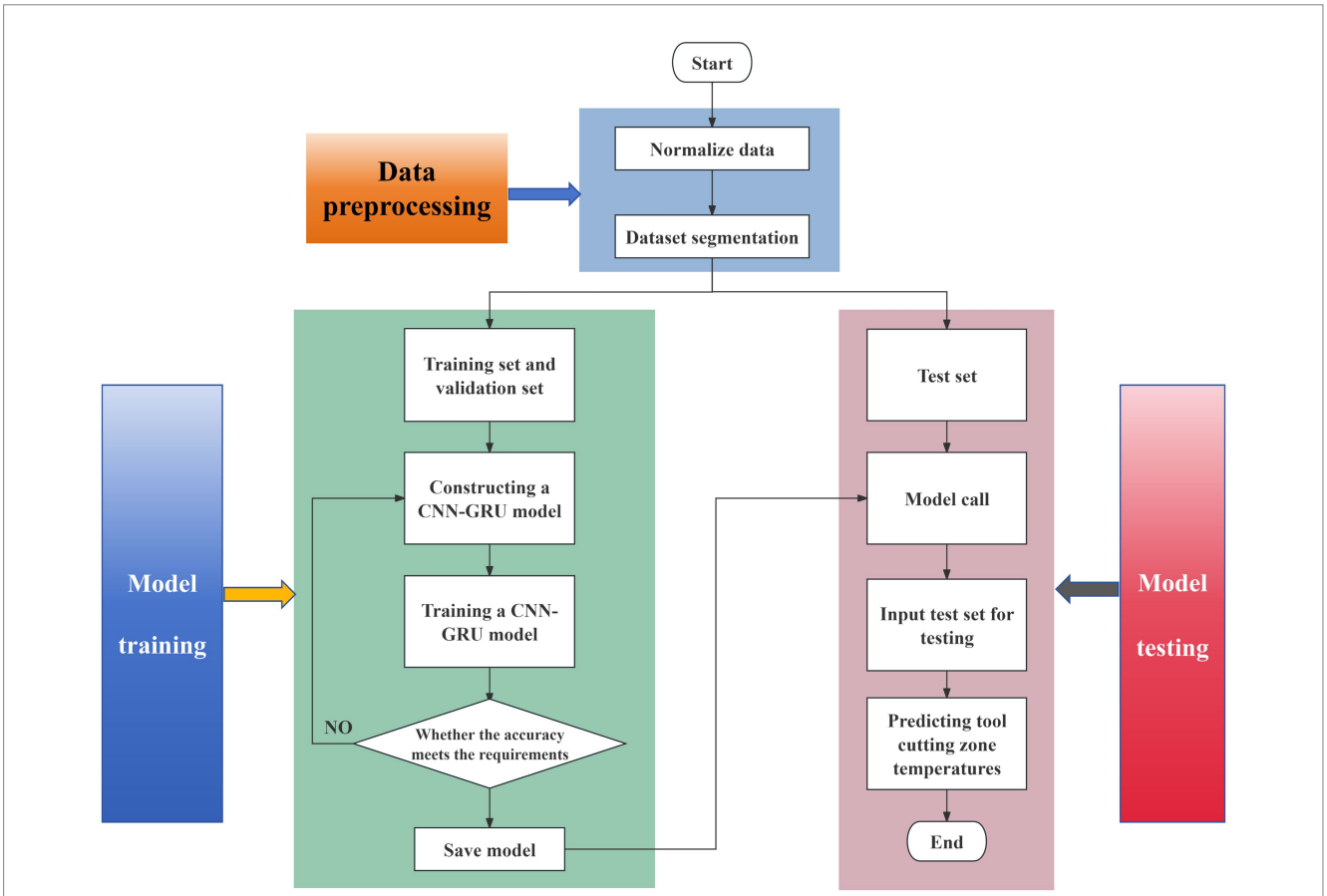


FIGURE 3
Prediction process based on the CNN-GRU model.

To determine the model structure, this study employs Mean Squared Error (MSE) and R-squared (R^2) as evaluation metrics. The formulas for these metrics are as Equations (3, 4):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{3}$$

where n represents the total number of samples, i denotes the current sample, $\hat{y}_i$ is the predicted value for the ith sample, and y_i is the true value for the ith sample. A smaller MSE indicates that the model's predictions are closer to the true values, signifying better model performance.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{4}$$

Where n represents the total number of samples, i denotes the current sample, $\hat{y}_i$ is the predicted value for the ith sample, y_i is the true value for the ith sample, $\bar{y}_i$ represents the mean of the true values y_i. The range of R^2 is [0,1], with a higher R^2 indicating better model performance.

In the inverse heat transfer problem solving model based on deep learning, effective data samples are the key to develop the model to accurately predict the boundary temperature conditions in the cutting region, among the 27 sets of full factorial test samples, the 24th set of test data is extracted as the test set data, and 80% of the remaining data is treated as the training set, and 20% is treated as the validation set. The machining parameters and the temperature at multiple points on the back face of the milling cutter are chosen as input features, and the temperature boundary conditions on the milling cutter cutting region are used as prediction labels. The compiled language for the neural network is Python 3.7, the model is built using the PyTorch deep learning framework, the operating system is 64-bit Windows 10, and the GPU is an NVIDIA GTX 1050Ti graphics card.

Among them, the parameters of the model are shown in Table 1, and the overall structure of the CNN-GRU model built in this paper is shown in Figure 4.

In order to confirm the validity and accuracy of the models, this paper compares the constructed CNN-GRU models with CNN, GRU, and LSTM networks, using MSE as the Loss function and R^2 as the evaluation index of model error. All models use the dataset delineated in the previous section, and the parameter details of each model are shown in Table 2, and the training Epoch and batch size of all models are kept the same in order to ensure the scientific nature of model comparison.

The Loss function curves for the training process of each model are shown in Figure 5A. The figure illustrates that, for the

TABLE 1 CNN-GRU model parameters.

Model parameters	Numerical values
CNN Layers	2
CNN convolutional kernel size	4
Number of Convolutional Kernels	First layer 132, second layer 164
Number of GRU layers	2
Number of GRU neural units	First layer 50, second layer 50
Total number of neurons in the model	117,157

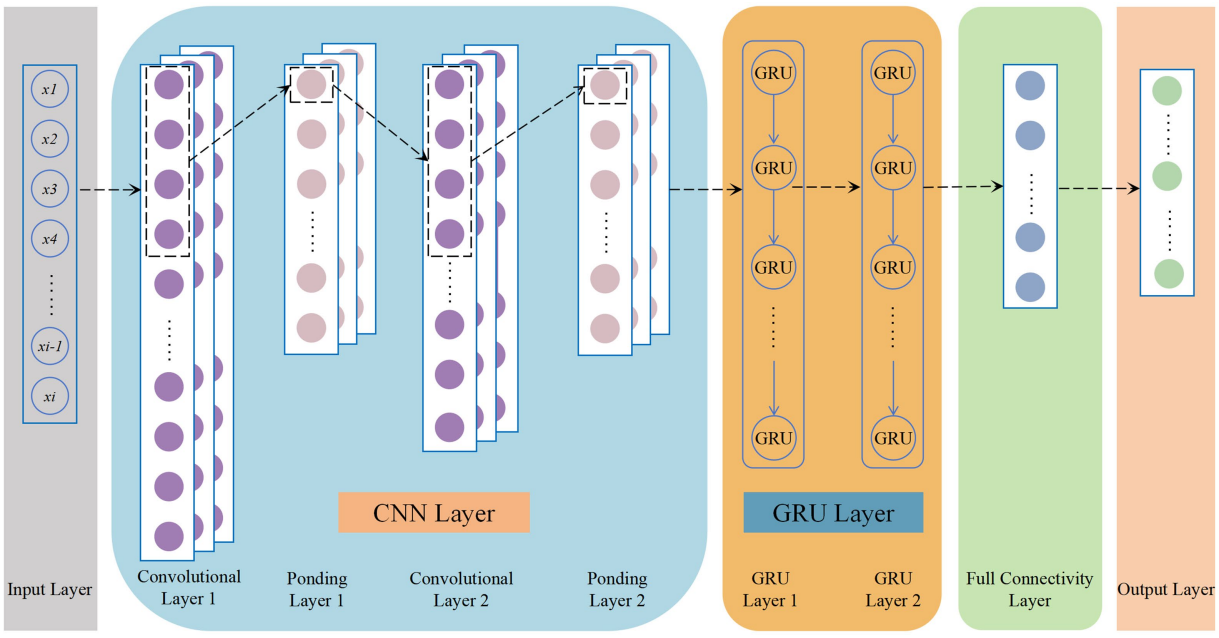


FIGURE 4
General structure of CNN-GRU model.

same number of training iterations (150), the CNN model stabilizes its Loss value at approximately the 120th iteration, making it the slowest to converge among the models. In contrast, the LSTM and GRU models stabilize around the 90th iteration. Notably, the CNN-GRU model exhibits a smooth trend and stabilizes as early as the 45th iteration, demonstrating the fastest convergence speed among all the models.

During 150 training sessions, the final LOSS value of the CNN-GRU model is 2.57×10^{-3}, the final LOSS value of the CNN model is 6.37×10^{-3}, the final LOSS value of the LSTM model is 5.3×10^{-3}, and the final LOSS value of the GRU model is 6.82×10^{-3}. In comparison, the CNN-GRU model exhibits better learning ability and fitting effect, and the evaluation indexes of each model are shown in Table 3.

The fitting curves of each model on the test set are shown in Figure 5B, from which it can be seen that the prediction curves of the CNN-GRU model are the closest to the real value, and compared with other models, it can predict the temperature trend on the cutting area of the tool more efficiently, especially in the position of the peaks and valleys of the best fitting, which further verifies that the prediction results of the CNN-GRU model are more in line with the practical requirements.

TABLE 2 Parameter details for each model.

Network model	Epoch	Batch size	Total number of neurons in the model	Single training time(s)
LSTM	150	50	65,389	3.72
GRU	150	50	53,283	1.67
CNN	150	50	78,633	4.03
CNN-GRU	150	50	117,157	6.8

3.3 Temperature boundary condition estimation model based on knowledge distillation with gated convolutional recurrent networks

Knowledge distillation is an instructor-student training structure that typically utilizes a student model with a simpler network structure to learn the knowledge provided by an instructor model that has been trained with a more complex network structure; this approach trades a slight performance loss for faster computation and smaller model parameters. Knowledge distillation works by training the student model with both the predictions of the teacher model (soft labeling) and the real data (hard labeling), and calculating the weighted total loss of the student model on both the soft and hard labels, essentially "migrating" the knowledge learned by the teacher model to the student model. The structure of the knowledge distillation strategy used in this paper is shown in Figure 6.

The specific knowledge distillation strategy process is as follows:

(1) The raw data that has been preprocessed is input to both the teacher model and the student model, the teacher model is the CNN-GRU model constructed in the previous section, and the student model is a small model with a single CNN layer and a single GRU layer.
(2) The output of the teacher model is softened using the Softmax function with temperature coefficient T. The processed labels are used as soft labels.
(3) Use the same Softmax function with temperature coefficient T to soften the results of the student model output, and process the labels of the student output and the soft labels of the teacher model output in the previous step through the distillation loss function $LOSS_{soft}$ to obtain the distillation loss function between the student model and the teacher model.
(4) Process the unsoftened student model output labels with the real hard labels through the student loss function $LOSS_{hard}$ to get the student loss.

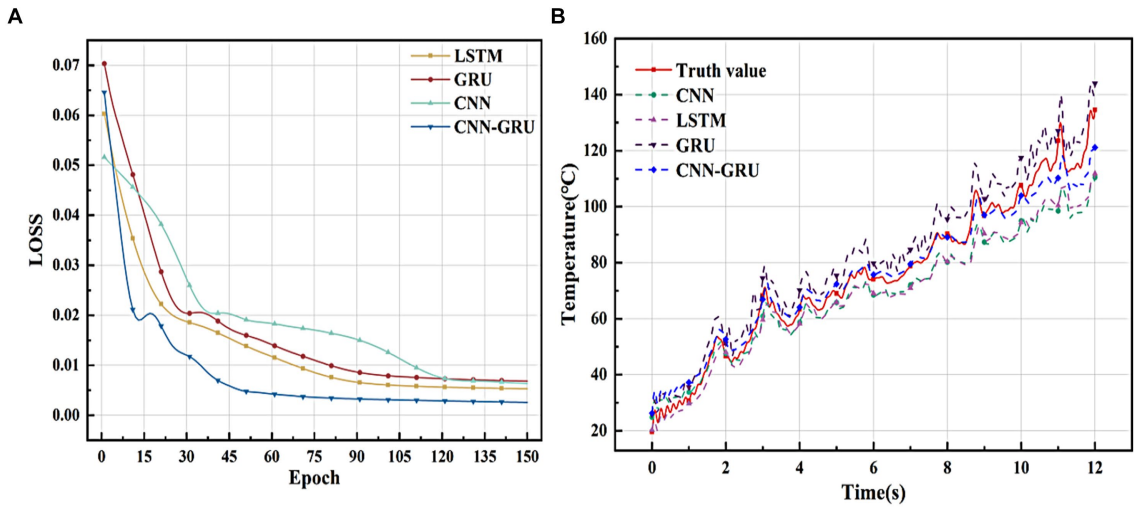


FIGURE 5
Performance comparison of different models. **(A)** The Loss function curves for each model training process. **(B)** Fit curves for each model on the test set.

TABLE 3 Evaluation indicators for each model.

Model	MSE	R^2
CNN	6.37×10^{-3}	0.89
LSTM	5.3×10^{-3}	0.93
GRU	6.82×10^{-3}	0.88
CNN-GRU	2.57×10^{-3}	0.98

(5) The distillation loss and the student loss are weighted to obtain the total loss, and the gradient of each parameter is updated in the backpropagation process.

The following are the calculation formulas involved in the knowledge distillation operation process:

Knowledge distillation soft labeling calculation formula as Equation (5):

$$q_i = \frac{\exp\left(z_i / T\right)}{\sum_{j=0}^{k} \exp\left(z_j / T\right)} \tag{5}$$

where T is the distillation temperature coefficient, used to control the "hardness" of the soft label. When T is larger, the soft label distribution area is uniform, more softened, when T is smaller, the soft label distribution closer to the hard label.

Distillation loss of the loss function LOSSsoft formula is as Equation (6):

$$LOSS_{soft} = \sum_{i=0}^{k} - p_i\left(u_i, T\right) \log\left(p_i\left(z_i, T\right)\right) \tag{6}$$

where k is the total number of samples, $p_i(u_i,T)$ is the ith output of the teacher model at temperature coefficient T, and $p_i(z_i,T)$ is the ith output of the student model at temperature coefficient T.

The loss function LOSShard for student loss is formulated as Equation (7):

$$LOSS_{hard} = \sum_{i=0}^{k} - y_i \log\left(p_i\left(z_i, 1\right)\right) \tag{7}$$

where $y_i7$is a vector of hard labels representing the class i output of the unsoftened student model.

The total loss of knowledge distillation can be expressed as Equation (8):

$$LOSS_{total} = \lambda LOSS_{soft} + \left(1 - \lambda\right) LOSS_{hard} \tag{8}$$

where λ are hyperparameters, which are fixed constants that can be empirically tuned to the reference or dynamically adjusted.

Based on the above knowledge distillation strategy for model optimization design of the constructed CNN-GRU teacher model, the first step is to construct a simple CNN-GRU student model, and with reference to the structure of the teacher model with 2 layers of CNN layers plus 2 layers of GRU layers, the student model structure is designed as a 1-layer CNN layer plus 1 layer of GRU layer structure. In order to determine the optimal student model total neuron number, the student models with total neuron number of 10, 20, 30, 40, 50, 60, 70, 80, and 90% of the teacher's model were designed, and the gradient descent training was performed on each model using the same training, validation, and test sets, and the training Epoch and batch sizes were consistent with those of the teacher's model. The learning ability and single-step training time of each student model not trained by the knowledge distillation strategy are first compared to the true values, and the comparison of the prediction results of each percentage of student models is shown in Figure 7A.

As can be seen from the figure, the goodness of fit of the student model gradually increases with the increase of the total number of neurons before the knowledge guidance of the teacher's model, and the goodness of fit tends to stabilize when the ratio of the student model to the teacher's model is 60%, which indicates that the closer the student model is to the teacher's model, the better the ability to learn the data, however, due to the structural limitation of the student model, the simple model structure is not enough to accurately reflect the complex nonlinear relationship between the input data and the output data. However, due to the structural limitations of the student model, the simple model structure is not enough to accurately reflect the complex nonlinear relationship between the input data and the output data, although the goodness of fit of the student model to the teacher's model still fluctuates slightly after the ratio of the student model to the teacher's model is more than 60%, but the overall learning ability does not improve much. The single-step training time consumed by the student model also becomes more with the increase of the total number of neurons, and the rate of change of the single-step training time consumed increases when the ratio of the student model to the teacher's model is 70%, which demonstrates that the closer the number of neurons of the student model is to that of the teacher's model, the slower the model's inference is, and the more hardware resources it occupies.

Then, using the knowledge distillation strategy, the teacher model trained in the previous section is used to "guide the training" of the above student models of different sizes, so as to transfer the knowledge learned from the teacher model to the student model. In order to avoid random errors, the distillation temperature coefficient T is set to [1,10], T takes an integer, the total loss weighting factor λ is set to [0.1,0.9], λ retains one decimal place, and the distillation effect of the model under the parameter combinations of T and λ is compared one by one, and it is finally determined that $T = 7$, $\lambda = 0.8$. The comparison of the prediction results of various proportions of the students' models after the distillation is shown in Figure 7B.

As can be seen from the figure, the student models (CNN-GRU + KD) guided by the teacher's model all have a better improvement in the goodness-of-fit, and the R^2 shows a smooth trend and stabilizes around 0.96 when the percentage of the student model to the teacher's model is 60%, which results in a longer single-step training time than that of the original model after the distillation before the original model is longer, when the percentage is more than 80%, the single-step training time of the model is close to that of the teacher model. Therefore, considering the goodness of fit of the student model and the single-step training time, the total number of neurons of the student model is determined to be 60% of the teacher model, and the network parameters of the student model are shown in Table 4. A comparison of the performance

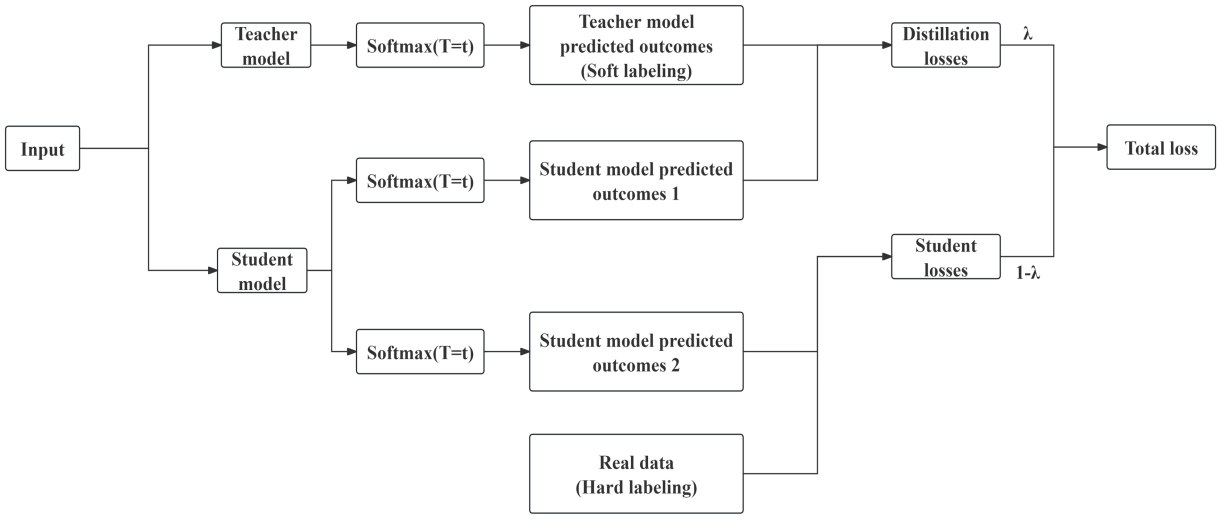


FIGURE 6
Structure of knowledge distillation strategy.

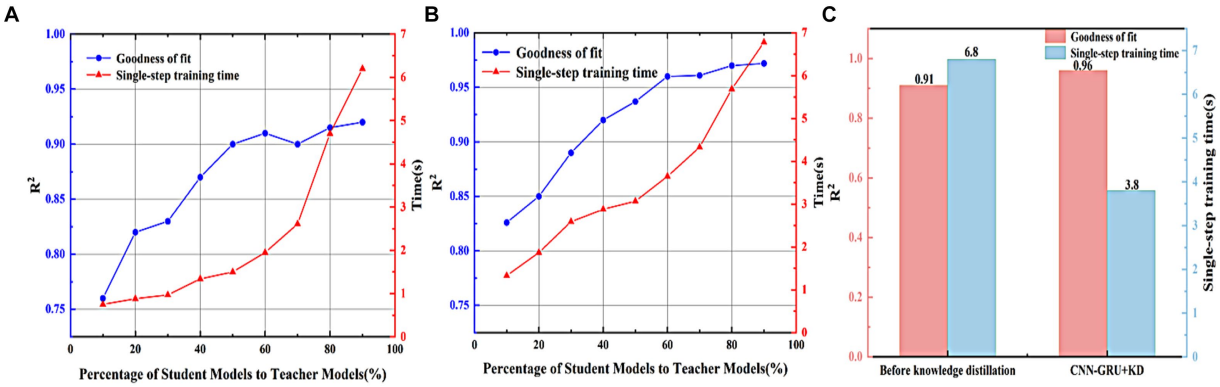


FIGURE 7
Comparison of model performance before and after adding knowledge distillation. **(A)** Comparison of student model training results for each scale. **(B)** Comparison of student model predictions for each scale after knowledge distillation. **(C)** Comparison of model performance improvement before and after acceleration of knowledge distillation strategy.

TABLE 4 Network parameters of the student model.

Model parameter	Numerical values
CNN Layers	1
CNN convolutional kernel size	4
Number of convolution kernels	148
GRU layers	1
Number of GRU neural units	90
Total number of neurons in the model	70,170

improvement of the model before and after acceleration by the knowledge distillation strategy is shown in Figure 7C.

From Figure 7C, it can be seen that the CNN-GRU+KD model accelerated based on the knowledge distillation strategy has an accuracy of 0.96, which compares with the teacher's model although there is some performance loss (loss of 2%), but it improves the prediction accuracy by 5% over the student's model of 0.91, and the training time of the CNN-GRU+KD as usual reduces by 44.1% compared with the teacher's model, which proves that the acceleration of the knowledge distillation strategy is feasibility of the compression model.

4 Experimental result and analysis

4.1 Model noise resistance test

In 3.3, the model accuracy and computation time have been discussed, and the results show that he CNN-GRU+KD model achieves significant time acceleration with minimal loss in accuracy compared to the teacher model, making it more suitable

for the rapid provision of necessary data for fast temperature field reconstruction.

In addition to model accuracy and computation time, the model's noise immunity to data noise is especially critical in the solution of the inverse heat conduction problem, because the inverse heat conduction problem is essentially an "unsettled" problem, which is more sensitive to the noise of the signal, and therefore the model's noise immunity needs to be tested. The CNN-GRU + KD after knowledge distillation is tested with the teacher model and the student model on the dataset with noise level $\sigma = 0K$, noise level $\sigma = 10K$, and noise level $\sigma = 10K$, respectively, and the results are shown in Figures 8A–I. Table 5 shows the MSE computed by the student model, the teacher model, and the CNN-GRU + KD Model on the test set under the noise level of these three sets of data.

As can be seen from the learning effects of the above three models at different noise levels, the student model with fewer model parameters, simple structure, and no knowledge distillation training performs the worst as the noise level increases, and has been severely distorted at noise level $\sigma = 10K$. The teacher model, on the other hand, can still learn the trend of real data in noisy data due to its complex model structure and more model parameters, and shows better noise immunity, which can be attributed to the dimensionality reduction of the noisy data by multilayer convolutional pooling, thus compressing the random noise information. Thus the CNN-GRU + KD trained based on the teacher model learns the better noise immunity of the teacher model and shows better robustness compared to the student model, and the stability of the CNN-GRU + KD is still satisfactory even in the case of high noise level.

4.2 Simulation reconstruction

In the milling process, the heat in the milling cutter is mainly transferred in three ways: the mutual transfer of heat between the tool-workpiece-chips in the cutting area, the convective heat transfer between the tool-stem-air, and the thermal radiation heat transfer in the high

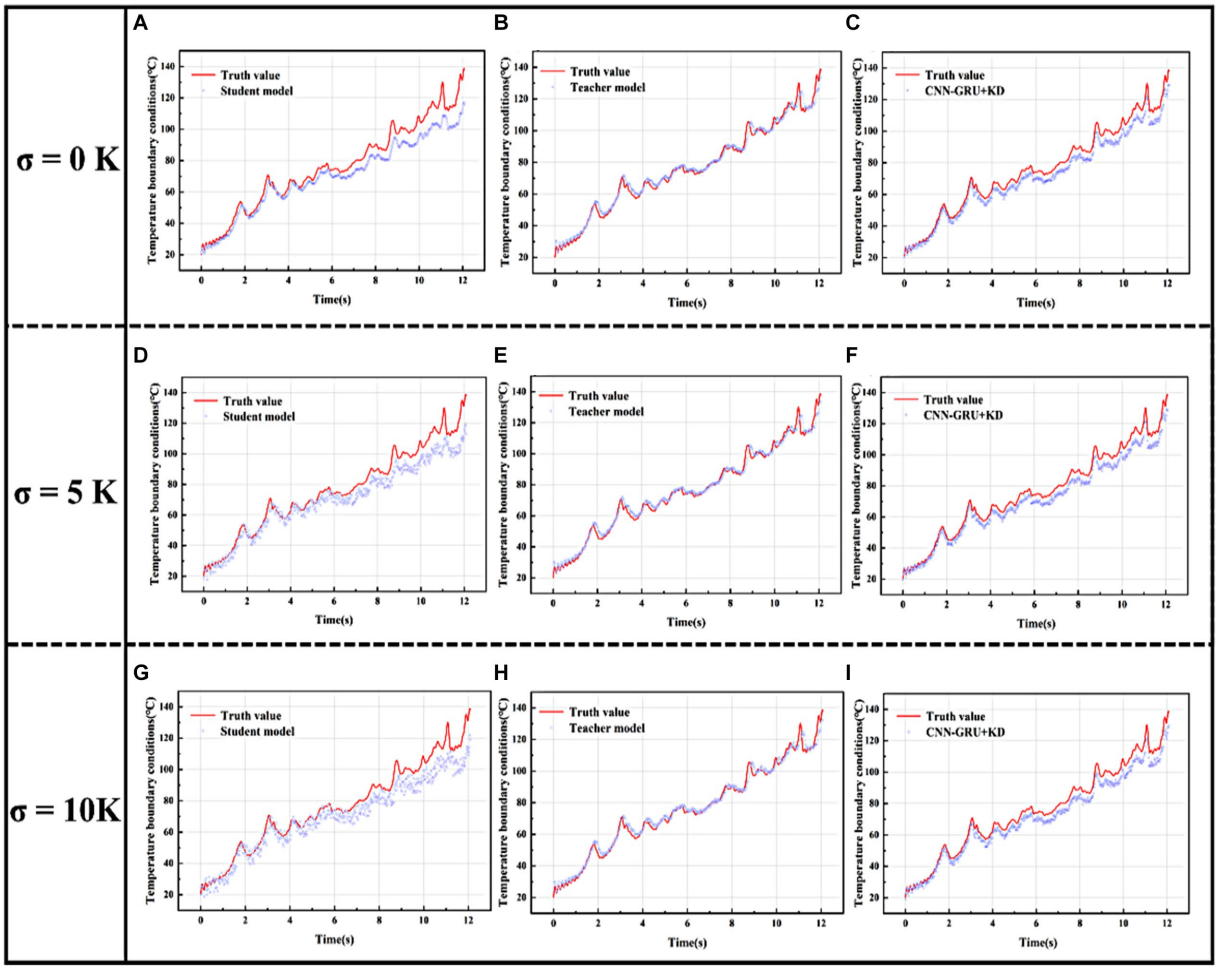


FIGURE 8
Test set fitting curves for the three models at different noise levels. **(A)** Noise level $\sigma = 0K$ student model fit curve on test set. **(B)** Noise level $\sigma = 0K$ teacher model fit curve on test set. **(C)** Noise level $\sigma = 0K$ CNN-GRU + KD model fit curve on test set. **(D)** Noise level $\sigma = 5K$ student model fit curve on test set. **(E)** Noise level $\sigma = 5K$ teacher model fit curve on test set. **(F)** Noise level $\sigma = 5K$ CNN-GRU + KD model fit curve on test set. **(G)** Noise level $\sigma = 10K$ student model fit curve on test set. **(H)** Noise level $\sigma = 10K$ teacher model fit curve on test set. **(I)** Noise level $\sigma = 10K$ CNN-GRU + KD model fit curve on test set.

TABLE 5 MSE of the three models at different data noise levels.

Noise level	Model		CNN-GRU + KD
	Student model	Teacher model	
$\sigma = 0K$	6.42×10^{-3}	2.57×10^{-3}	2.85×10^{-3}
$\sigma = 5K$	7.89×10^{-3}	3.02×10^{-3}	3.52×10^{-3}
$\sigma = 10K$	1.69×10^{-2}	3.95×10^{-3}	4.47×10^{-3}

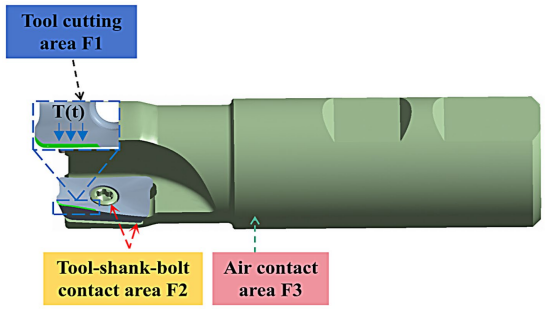


FIGURE 9
Three-dimensional schematic diagram of the tool-shaft-bolt cutting system.

temperature region, because the radiation area of the high temperature region in the process of metal cutting is generally very small, so the thermal radiation heat transfer is generally negligible. Therefore, before constructing the transient heat transfer model of double-edged milling cutter, if only simplified two-dimensional or three-dimensional modeling for the milling cutter will greatly affect the accuracy of the heat transfer model, should be three-dimensional modeling of the cutting system as a whole, including inserts, shanks, bolts, etc. This paper in accordance with the actual tool as a whole to establish a three-dimensional geometric model shown in Figure 9, and in accordance with Figure 1 in the cutting area of the tool cutting region image of the model tool cutting area location for detailed division.

The tool cutting area is defined as F1, the tool-stem-bolt contact area is defined as F2, and the contact area between the cutting system and the air is defined as F3. The heat flow inside the system during the whole milling process can be understood as follows: the cutting heat enters into the overall cutting system from the tool cutting area, and then transfers to the tool stem and bolt through the tool-stem-bolt contact area, and the heat dissipation is realized by convective heat transfer with the surrounding air in the contact area of the cutting system and the air. The cutting system in contact with the air through the convective heat transfer with the surrounding air to achieve heat dissipation, this process is a complex, three-dimensional, transient heat transfer process, the control equation of heat transfer can be expressed as Equation (9):

$$\frac{\partial}{\partial x}\left(k_t \frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k_t \frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k_t \frac{\partial T}{\partial z}\right)=\rho c_t \frac{\partial T}{\partial t}, t>0 \tag{9}$$

where T represents the transient temperature at the internal point of the tool heat transfer model, (x,y,z,t) are the spatial and temporal variables of the tool heat transfer model, and k_t and c_t represent the specific heat capacity and thermal conductivity of the material and ρ the material density.

Based on the proposed CNN-GRU + KD model, the actual measured temperature of the back face of the milling cutter and the machining parameters are inputted into the model to predict the temperature boundary value on the cutting area of the milling cutter, and the estimated temperature boundary conditions are inputted into the transient heat transfer model of the milling cutter to realize the reconstruction of the tool temperature field during the milling process. In this paper, the tool temperature field reconstruction is carried out for different rotational speeds under the feed rate $f = 0.075$ mm/z per tooth, radial depth of cut (ae) of 0.2 mm, and the milling mode is dry cutting and reverse milling. According to the temperature boundary conditions prediction results and double-edged milling cutter transient heat conduction model combination of the milling process tool temperature field reconstruction, respectively, take 4 s, 8 s, 12 s to draw the temperature field of the pre-milling, mid-term, the end of the milling period, the results are shown in Figures 10A–I.

In order to verify the accuracy of the milling process tool temperature field reconstruction model, the milling cutter transient heat transfer finite element calculation results corresponding to the location of the thin-film thermocouple sensor temperature measurement point of the unit temperature reconstruction curve exported and compared with the sensor's actual measurement of the milling temperature curve, the results of the reconstruction of the temperature and the measured temperature comparison results in different working conditions are shown in Figures 10J–L.

As can be seen from Figures 10A–I, in the pre-milling process, the tool cutting area produces a large amount of cutting heat, which will spread around to form a temperature gradient, with the cutting process heat generation and dissipation to reach an equilibrium state, the tool near the cutting area near the temperature distribution gradually tends to stabilize, while the tool distal space on the tool is still part of the region's temperature field distribution in the change, because this part of the is farther from the cutting region, the heat conduction is slower, and it takes longer to reach the thermal steady state. In addition, it can be seen that there is a large temperature gradient at the tip of the milling cutter, the temperature gradient distribution on the front face is very uneven, and the high temperature region is mainly concentrated in the vicinity of the cutting area and the temperature increases with time.

From Figures 10J–L, it can be seen that the reconstructed temperature curve of the milling process fits well with the temperature curves of the four temperature measurement points. Subject to the actual measurement temperature of the sensor, the fit between the reconstructed temperature curves of the four temperature measurement points at the locations of the above three conditions and the actual measurement temperature curves of the sensor is as high as 0.97, and the fit is as low as 0.92, with the fit above 0.9, which proves that there is a good fitting relationship between the reconstructed temperature curve There is a good fitting relationship between the reconstructed temperature curve and the actual measured temperature curve of the sensor. The Root Mean Square Error (RMSE) was chosen as the evaluation index for calculating the error between the reconstructed temperature profile and the actual measured temperature profile, and the formula of RMSE is as Equation (10):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{n}} \tag{10}$$

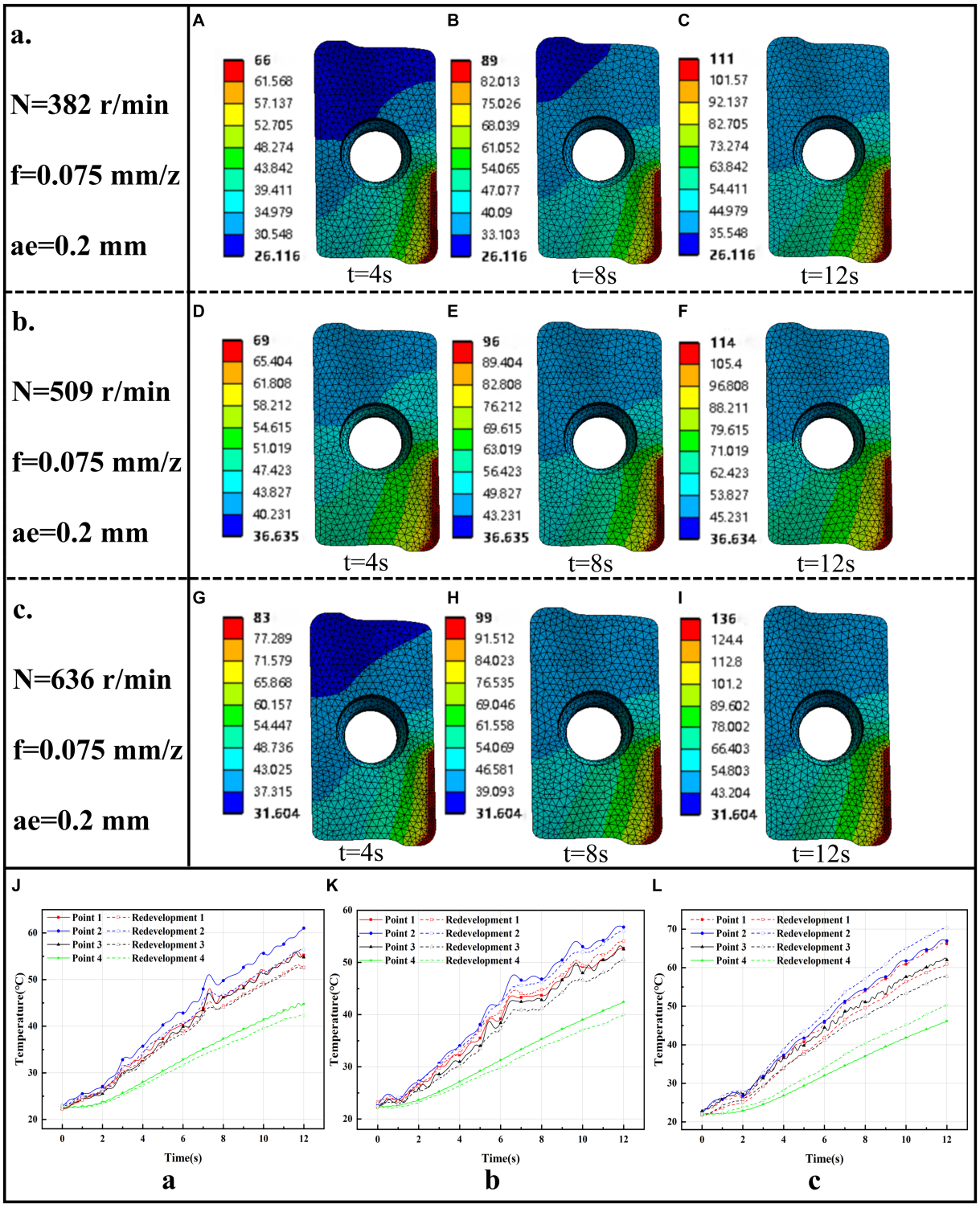


FIGURE 10
Temperature field reconstruction results for three different operating conditions. a. Tool Temperature Field Reconstruction Results for Case N = 382 r/min, f = 0.075 mm/z, ae = 0.2 mm. **(A)** t = 4 s. **(B)** t = 8 s. **(C)** t = 12 s. b. Tool Temperature Field Reconstruction Results for Case N = 509 r/min, f = 0.075 mm/z, ae = 0.2 mm. **(D)** t = 4 s. **(E)** t = 8 s. **(F)** t = 12 s. c. Tool Temperature Field Reconstruction Results for Case N = 636 r/min, f = 0.075 mm/z, ae = 0.2 mm. **(G)** t = 4 s. **(H)** t = 8 s. **(I)** t = 12 s. **(J)** N = 382 r/min, f = 0.075 mm/z, ae = 0.2 mm Reconstruction temperature vs. measured temperature. **(K)** N = 509 r/min, f = 0.075 mm/z, ae = 0.2 mm Reconstruction temperature vs. measured temperature. **(L)** N = 636 r/min, f = 0.075 mm/z, ae = 0.2 mm Reconstruction temperature vs. measured temperature.

where n represents the data points, $\sigma = 10K$ represents the measured temperature value of the sensor at time i moment, and y_i represents the modeled temperature result at time i moment.

The RMSE between the reconstructed temperature curve at the temperature measurement point when the spindle speed N=509 r/min and the actual temperature curve measured by the sensor is 1.85, 1.98,

1.62, and 1.43°C; the RMSE between the reconstructed temperature curve at the temperature measurement point when the spindle speed $N = 636$ r/min and the actual temperature curve measured by the sensor is 4.51, 2.89, 3.68, and 3.56°C, which is in the acceptable range. The above calculations prove the accuracy and feasibility of the tool temperature field reconstruction method for milling process based on the inverse heat conduction problem used in this paper.

5 Conclusion

In this paper, a solution model for the inverse heat transfer problem based on a convolutional gated recurrent network to predict the temperature boundary conditions in the cutting region of the tool is proposed. And the CNN-GRU model that will be built is compressed and accelerated using the knowledge distillation strategy, and the big model that will be built is considered as the teacher model, and the small CNN-GRU is designed, which is considered as the student model. The goodness-of-fit of the CNN-GRU + KD model trained by the teacher model guidance is 0.96, and the single-step training time of the model is reduced by 44.1% compared to the teacher model. Compared to than the student model without teacher model guidance, the accuracy of the CNN-GRU + KD model increased by 5% and the mean square error decreased by 55%. By adding different levels of random noise to the model input data, the CNN-GRU + KD model learns the noise-resistant ability of the teacher model and still shows good robustness and stability under noisy data.

The transient heat transfer model of the tool is constructed, and all the surface areas of the model are divided into three major areas, namely, the tool cutting area F1, the tool-shank-bolt contact area F2, and the contact area between the cutting system and the air F3, according to the actual situation, and the boundary conditions on each area are defined according to the theory of heat transfer. Based on the CNN-GRU + KD model predicted temperature boundary conditions in the tool cutting region, combined with the tool transient heat conduction model, the temperature field of the milling cutter was reconstructed for three different working conditions, and the reconstructed temperature curves of the milling process at the location of the temperature measurement points and the temperature curves of the sensors were calculated for the goodness-of-fit and the root-mean-square error, and the goodness-of-fit of curves in the three working conditions was the highest of 0.97. The minimum root mean square error is 1.43°C, which are in the acceptable range, and the reconstruction of tool temperature field in milling process is realized.

Based on the results of this paper, future research could further enhance the CNN-GRU+KD model for predicting temperature boundary conditions in the tool cutting region. One promising direction is to integrating intuitionistic fuzzy approaches, as demonstrated by Versaci and La Foresta (2024) in energy management. Fuzzy systems could more effectively handle the uncertainty and noise inherent in temperature data. By integrating fuzzy rules based on operator experience, the CNN-GRU model could become more versatile and adaptable to various milling scenarios, thereby improving the robustness and adaptability of the CNN-GRU+KD model under different operating conditions. Combining intuitionistic fuzzy approaches with the CNN-GRU+KD model could lead to even greater performance improvements, particularly in situations where real-time decision-making and noise immunity are critical.

Data availability statement

The data that support the finding of this study are available from the corresponding author upon reasonable request.

Author contributions

FM: Writing – original draft, Investigation, Methodology, Writing – review & editing. HW: Conceptualization, Writing – original draft, Visualization. ME: Validation, Writing – review & editing. XW: Investigation, Writing – original draft. YC: Funding acquisition, Writing – review & editing. JY: Funding acquisition, Supervision, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by National Natural Science Foundation of China (Nos. 52275407, 52175379, 51905071, 52072056, and 52305449), the Applied Basic Research Program (Youth Project) of Liaoning Province (2023JH2/101600060), the Project of Department of Education of Liaoning Province, China (Nos. LJKZ0473, LJKZ0483, and LJKQZ20222336), the Doctoral Research Foundation of Liaoning Province (2019-BS-043), and the Key Laboratory of Precision and Special Processing of Ministry of Education, and the Dalian University of Technology (JMTZ201902).

Conflict of interest

ZS was employed by Angang Heavy Machinery Co., Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Alammari, Y., Weng, J., Saelzer, J., and Biermann, D. (2024). Transient Temperature at Tool–Chip Interface during Initial Period of Chip Formation in Orthogonal Cutting of Inconel 718. *Materials* 17:2232. doi: 10.3390/ma17102232

Carvalho, S. R. E., Silva, S. L., Machado, A. R., and Guimaraes, G. (2006). Temperature determination at the chip–tool interface using an inverse thermal model considering the tool and tool holder. *J. Mater. Process. Technol.* 179, 97–104. doi: 10.1016/j.jmatprotec.2006.03.086

Chen, C., and Pan, Z. (2023). A neural network-based method for real-time inversion of nonlinear heat transfer problems. *Energies* 16:7819. doi: 10.3390/en16237819

Cheng, G J, Liu, L T, Qiang, X J, and Liu, Ye (2016). "Industry 4.0 development and application of intelligent manufacturing." in 2016 international conference on information system and artificial intelligence (ISAI). IEEE. pp. 407–410. doi: 10.1109/ISAI.2016.0092

Cichosz, P., Karolczak, P., and Waszczuk, K. (2023). Review of cutting temperature measurement methods. *Materials* 16:6365. doi: 10.3390/ma16196365

Cortés, O, Urquiza, G, Hernandez, J A, and Cruz, Marco A. (2007) "Artificial neural networks for inverse heat transfer problems." in IEEE Electronics, Robotics and Automotive Mechanics Conference, 2007. pp. 198–201. doi: 10.1109/CERMA.2007.4367685

Cui, J, Kingsbury, B, Ramabhadran, B, Saon, George, Sercu, Tom, Audhkhasi, Kartik, et al. (2017). "Knowledge distillation across ensembles of multilingual models for low-resource languages." in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE.* pp. 4825–4829. doi: 10.1109/icassp.2017.7953073

Czél, B., Woodbury, K. A., and Gróf, G. (2013). Inverse identification of temperature-dependent volumetric heat capacity by neural networks. *Int. J. Thermophys.* 34, 284–305. doi: 10.1007/s10765-013-1410-6

Deng, S., and Hwang, Y. (2007). Solution of inverse heat conduction problems using Kalman filter-enhanced Bayesian back propagation neural network data fusion. *Int. J. Heat Mass Transf.* 50, 2089–2100. doi: 10.1016/j.ijheatmasstransfer.2006.11.019

He, Y, Lin, J, Liu, Z, Wang, Hanrui, Li, Li-Jia, and Han, Song (2018). "Amc: Automl for model compression and acceleration on mobile devices." in *Proceedings of the European conference on computer vision (ECCV). 2018.* pp. 784–800. doi: 10.1007/978-3-030-01234-2_48

He, Z., Shi, T., Xuan, J., and Li, T. (2021). Research on tool wear predicti-on based on temperature signals and deep learning. *Wear* 478-479:203902. doi: 10.1016/j.wear.2021.203902

Jaspers, S. P. F. C., Dautzenberg, J. H., and Taminiau, D. A. (1998). Temperature measurement in orthogonal metal cutting. *Int. J. Adv. Manuf. Technol.* 14, 7–12. doi: 10.1007/BF01179411

Javaid, M., Haleem, A., Singh, R. P., Rab, S., and Suman, R. (2021). Significance of sensors for industry 4.0: roles, capabilities, and applications. *SensorsInt.* 2:100110. doi: 10.1016/j.sintl.2021.100110

Kamyab, S., Azimifar, Z., Sabzi, R., and Fieguth, P. (2022). Deep learning methods for inverse problems. *PeerJ Comp. Sci.* 8:e951. doi: 10.7717/peerj-cs.951

Kim, J., and Lee, C. (2020). Prediction of turbulent heat transfer using convolutional neural networks. *J. Fluid Mech.* 882:A18. doi: 10.1017/jfm.2019.814

Korkmaz, M. E., and Gupta, M. K. (2024). A state of the art on cryogenic cooling and its applications in the machining of difficult-to-machine alloys. *Materials* 17:2057. doi: 10.3390/ma17092057

Lee, K. M., Cheng, K., Ding, H., Kazmer, D. O., Lin, W., Luo, R. C., et al. (2015). Guest editorial introduction to the focused section on mechatronics for intelligent manufacturing. *IEEE/ASM-E Trans. Mech.* 20, 1001–1004. doi: 10.1109/TMECH.2015.2422214

Leonidas, E., Ayvar-Soberanis, S., Laalej, H., Fitzpatrick, S., and Willmott, J. R. (2022). A comparative review of thermocouple and infrared radiation temperature measurement methods during the machining of metals. *Sensors* 22:4693. doi: 10.3390/s22134693

Li, X., Ding, Q., and Sun, J. Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab. Eng. Syst. Saf.* 172, 1–11. doi: 10.1016/j.ress.2017.11.021

Li, S, Li, X, and Jiang, T. (2009) "A prediction method of life and reliability for CSAL-T using Grey RBF neural networks" in *2009 16th International Conference on Industrial Engineering and Engineering Management. IEEE.* pp. 699–703. doi: 10.1109/ICIEEM.2009.5344500

Liang, L., Xu, H., and Ke, Z. (2013). An improved three-dimensional invers-e heat conduction procedure to determine the tool-chip interface temperature in dry turning. *Int. J. Therm. Sci.* 64, 152–161. doi: 10.1016/j.ijthermalsci.2012.08.012

Lixin, W, Zhenhuan, W, Yudong, F, and Guoan, Yang (2016). "Remaining life predictions of fan based on time series analysis and BP neural networks." in *2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, 2016.* pp. 607–611. doi: 10.1109/ITNEC.2016.7560432

Longbottom, J. M., and Lanham, J. D. (2005). Cutting temperature measurement while machining–a review. *Aircr. Eng. Aerosp. Technol.* 77, 122–130. doi: 10.1108/00022660510585956

Luo, J H, Wu, J, and Lin, W. (2017). "Thinet: a filter level pruning method for deep neural network compression" in *Proceedings of the IEEE International Conference On Computer Vision. 2017.* pp. 5058–5066. doi: 10.1109/ICCV.2017.541

Murio, D. A. (1981). The mollification method and the numerical solution of an inverse heat conduction problem. *SIAM J. Sci. Stat. Comput.* 2, 17–34. doi: 10.1137/0902003

Oommen, V., and Srinivasan, B. (2022). Solving inverse heat transfer problems without surrogate models: a fast, data-sparse, physics informed neural network approach. *J. Comput. Inf. Sci. Eng.* 22:041012. doi: 10.1115/1.4053800

Qian, W., Hui, X., Wang, B., Zhang, Z., Lin, Y., and Yang, S. (2023). Physics-informed neural network for inverse heat conduction problem. *Heat Trans. Res.* 54, 65–76. doi: 10.1615/HeatTransRes.2022042173

Ren, L., Sun, Y., Wang, H., and Zhang, L. (2018). Prediction of bearing remaining useful life with deep convolution neural network. *IEEE Access* 6, 13041–13049. doi: 10.1109/ACCESS.2018.2804930

Versaci, M., and La Foresta, F. (2024). Fuzzy approach for managing renewable energy flows for DC-microgrid with composite PV-WT generators and energy storage system. *Energies*, 17:402. doi: 10.3390/en17020402

Wang, Y, Xu, C, Xu, C, and Tao, Dacheng (2018). "Adversarial learning of portable student networks." in *Proceedings of the AAAI conference on artificial intelligence. 2018*, 32(1): 4260–4267. doi: 10.1609/aaai.v32i1.11667

Wang, Z., Yang, L., Lin, H., Zhao, G., Liu, Z., and Song, X. (2023). Distributed deep learning optimization of heat equation inverse problem solvers. *IEEE Trans. Comp. Aided Design Integ. Circ. Syst.* 42, 4831–4843. doi: 10.1109/TCAD.2023.3296370

Wang, H., Yang, Q., Zhu, X., Zhou, P., and Yang, K. (2018). Inverse estimation of heat flux using linear artificial neural networks. *Int. J. Therm. Sci.* 132, 478–485. doi: 10.1016/j.ijthermalsci.2018.04.034

Zhang, B., Wu, G., Gu, Y., Wang, X., and Wang, F. (2022). Multi-domain physics-informed neural network for solving forward and inverse problems of steady-state heat conduction in multilayer media. *Phys. Fluids* 34:116116. doi: 10.1063/5.0116038

Zhang, Y., Xiong, R., He, H., and Pecht, M. G. (2018). Long short-term m-emory recurrent neural network for remaining useful life prediction of lithium-i-on batteries. *IEEE Trans. Veh. Technol.* 67, 5695–5705. doi: 10.1109/TVT.2018.2805189

Zhang, Z., and Wang, Q. (2024). Sequence-to-sequence stacked gate recurrent unit networks for approximating the forward problem of partial differential equations. *IEEE Access*, 12, 61795–61809. doi: 10.1109/ACCESS.2024.3395517

Zhang, X, Zhou, X, Lin, M, and Sun, Jian (2018). "Shufflenet: an extremely efficient convolutional neural network for mobile devices." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.* pp. 6848–6856. doi: 10.1109/cvpr.2018.00716

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 08 November 2024
DOI 10.3389/fnbot.2024.1484038

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Xiaoyin Zheng,
XMotors.ai, United States
Huayu Zhang,
The Chinese University of Hong Kong, China

*CORRESPONDENCE
Byeong-il Lee
✉ bilee@pknu.ac.kr
Myunggi Yi
✉ myunggi@pknu.ac.kr

RECEIVED 21 August 2024
ACCEPTED 28 October 2024
PUBLISHED 08 November 2024

CITATION
Malekroodi HS, Seo S-D, Choi J, Na C-S, Lee B-i and Yi M (2024) Real-time location of acupuncture points based on anatomical landmarks and pose estimation models. *Front. Neurorobot.* 18:1484038. doi: 10.3389/fnbot.2024.1484038

Real-time location of acupuncture points based on anatomical landmarks and pose estimation models

Hadi Sedigh Malekroodi[1], Seon-Deok Seo[2], Jinseong Choi[2], Chang-Soo Na[3], Byeong-il Lee[1,4,5]* and Myunggi Yi[1,2,5]*

[1]Industry 4.0 Convergence Bionics Engineering, Pukyong National University, Busan, Republic of Korea, [2]Major of Biomedical Engineering, Division of Smart Healthcare, Pukyong National University, Busan, Republic of Korea, [3]College of Korean Medicine, Dongshin University, Naju, Republic of Korea, [4]Major of Human Bioconvergence, Division of Smart Healthcare, Pukyong National University, Busan, Republic of Korea, [5]Digital of Healthcare Research Center, Institute of Information Technology and Convergence, Pukyong National University, Busan, Republic of Korea

Introduction: Precise identification of acupuncture points (acupoints) is essential for effective treatment, but manual location by untrained individuals can often lack accuracy and consistency. This study proposes two approaches that use artificial intelligence (AI) specifically computer vision to automatically and accurately identify acupoints on the face and hand in real-time, enhancing both precision and accessibility in acupuncture practices.

Methods: The first approach applies a real-time landmark detection system to locate 38 specific acupoints on the face and hand by translating anatomical landmarks from image data into acupoint coordinates. The second approach uses a convolutional neural network (CNN) specifically optimized for pose estimation to detect five key acupoints on the arm and hand (LI11, LI10, TE5, TE3, LI4), drawing on constrained medical imaging data for training. To validate these methods, we compared the predicted acupoint locations with those annotated by experts.

Results: Both approaches demonstrated high accuracy, with mean localization errors of less than 5 mm when compared to expert annotations. The landmark detection system successfully mapped multiple acupoints across the face and hand even in complex imaging scenarios. The data-driven approach accurately detected five arm and hand acupoints with a mean Average Precision (mAP) of 0.99 at OKS 50%.

Discussion: These AI-driven methods establish a solid foundation for the automated localization of acupoints, enhancing both self-guided and professional acupuncture practices. By enabling precise, real-time localization of acupoints, these technologies could improve the accuracy of treatments, facilitate self-training, and increase the accessibility of acupuncture. Future developments could expand these models to include additional acupoints and incorporate them into intuitive applications for broader use.

KEYWORDS

deep learning, acupuncture, traditional medicine, computer vision, pose estimation

1 Introduction

Acupuncture is an ancient medical technique is a practice with roots extending back thousands of years. It involves the precise insertion of thin, sterile needles into specific points on the body known as acupoints (Formenti et al., 2023; Hou et al., 2015). These acupoints lie along meridians, or pathways, that are believed to facilitate the flow of vital energy, known as

qi or chi, throughout the body. By stimulating these acupoints, acupuncture aims to balance qi flow and promote healing. These points are not visible to the naked eye but are identified based on anatomical landmarks, palpation (feeling for subtle depressions or sensitivities), and traditional knowledge passed down through generations of practitioners (Tegiacchi, 2021).

In traditional medicine, acupuncture is used to treat various conditions including chronic pain, nausea, allergies, anxiety, depression, infertility, and more (Formenti et al., 2023; Yang et al., 2011). It is thought to work by releasing natural painkillers called endorphins, regulating blood flow, stimulating nerves and connective tissue, altering brain chemistry, and affecting hormone release (Wang et al., 2022; Vanderploeg and Yi, 2009). There are hundreds of acupoints located throughout the body, each associated with specific meridians and therapeutic effects (Zhang B. et al., 2022). For example, acupoint LI4 (Hegu), located between the thumb and index finger, is commonly used to relieve headaches and toothaches (Lu and Lu, 2008). Some practitioners even suggest its potential benefits for managing symptoms associated with Parkinson's disease (Park et al., 2023). Once dismissed by Western medicine, acupuncture has gained more mainstream acceptance in recent decades. In 1997, the National Institutes of Health found acupuncture to be effective for nausea and other conditions. Since then, clinical trials have demonstrated its efficacy for various health issues (Mayer, 2000; Mao and Khanna, 2012). Today, acupuncture is practiced worldwide including in Western countries. It is one of the most widely used forms of alternative and complementary medicine (Yang et al., 2011; Wang et al., 2022).

Traditionally, acupuncturists locate acupuncture points by feeling for specific landmarks on the body, such as bony protrusions or muscle lines. However, manual identification depends heavily on the experience of the acupuncturist and can suffer from inaccuracy, and can be time-consuming. Technology may be able to improve acupoint localization. Artificial intelligence (AI) can be used to revolutionize the practice of acupuncture. One of the most promising applications of AI in acupuncture is the use of computer vision to locate acupuncture points (Wang et al., 2022; Sun et al., 2020; Zhang M. et al., 2022). Computer vision techniques like pose estimation provide an attractive solution by automating acupoint localization in a standardized way. Pose estimation is an important computer vision task that involves detecting key points on the human body and understanding their positions and orientations. It has a wide range of applications such as human-computer interaction, augmented reality, action recognition, and motion capture (Sulong and Randles, 2023).

Recent studies have increasingly focused on leveraging computer vision techniques to automate the identification and localization of acupoints, recognizing the complexity of acupoint anatomy and the subtlety of acupoint landmarks. Deep learning approaches, particularly convolutional neural networks (CNNs), have emerged as promising tools for acupoint recognition due to their powerful feature extraction capabilities. Researchers have explored various architectures, including U-Net, cascaded networks, and improved high-resolution networks (HRNet), to enhance detection accuracy (Sun et al., 2020; Sun et al., 2022; Chan et al., 2021; Li et al., 2024; Yuan et al., 2024). In a recent study, Liu et al. (2023) introduced an improved Keypoint RCNN network was designed for back acupoint localization. By incorporating a posterior median line positioning method, the accuracy improved to 90.12%. Another significant development is the integration of anatomical measurements and proportional bone measurement methods with deep learning models to improve acupoint localization (Zhang M. et al., 2022; Chan et al., 2021). This approach combines traditional acupuncture knowledge with modern computer vision techniques.

Researchers have also explored the application of augmented reality (AR) and mixed reality (MR) technologies to visualize and localize acupoints in real-time, with systems like FaceAtlasAR and HoloLens 2-based applications showing promise. These technologies offer real-time tracking and visualization capabilities, potentially improving the practical application of automated acupoint detection systems in clinical settings (Zhang M. et al., 2022; Chen et al., 2021; Chen et al., 2017). For instance, Yang et al. (2021) developed tools like the SMART Table, which integrates 3D and AR technologies to improve acupoint education, training, and evaluation. This interactive system is designed to support both educational purposes and clinical competency assessments, showing promise in enhancing skills related to acupuncture and musculoskeletal treatments. Despite the limited number of studies in this area, several limitations persist in the current research despite recent advancements. These issues include limited datasets and accuracy problems in certain body areas. Many studies focus on a small number of acupoints or specific body regions (Sun et al., 2020; Chan et al., 2021), which restricts the applicability of their methods to comprehensive acupoint recognition.

The primary objective of this study is to develop a real-time acupuncture point detection system using state-of-the-art pose estimation models. While previous works like Sun et al. (2022) have shown promising results, our approach offers several key innovations. We explore and compare two distinct computer vision techniques: utilizing a real-time landmark detection framework to map acupoint locations based on classical proportional measurement methods, and fine-tuning a state-of-the-art pose estimation model on a custom dataset to directly predict acupoint coordinates. Our system is designed to detect a comprehensive set of acupoints, not limited small number as in previous studies. Furthermore, we develop an integrated application that enables real-time visualization of predicted acupuncture points on a webcam feed, showcasing their potential for assistive technologies in acupuncture treatment. Through this research, we aim to address several key questions: How does the accuracy of acupoint localization using a landmark-based approach compare to that of a deep learning-based pose estimation model? To what extent can these computer vision techniques be applied in real-time for practical acupuncture assistance? What are the limitations and potential improvements for each approach in the context of acupoint localization? By addressing these questions, our work aims to bridge the gap between traditional manual methods and

Abbreviations: AI, Artificial intelligence; FPN, Feature pyramid network; PAN, Path aggregation network; CNN, Convolutional neural network; COCO, Common objects in context; mAP, Mean average precision; OKS, Object keypoint similarity; TP, True positives; FP, False positives; FN, False negatives; TN, True negatives; AP, Average precision; SGD, Stochastic gradient descent; CIoU, Complete intersection over union; DFL, Distribution focal loss; BCE, Binary cross entropy; KIoU, Keypoint intersection over union; FPS, Frames per second; CV, Conception vessel; BL, Bladder; GB, Gallbladder; GV, Governing vessel; LI, Large intestine; LU, Lung; PC, Pericardium; SI, Small intestine; ST, Stomach; TE, Triple energizer; WHO, World Health Organization; RGB, Red, green, blue (color model); GPU, Graphics processing unit; IoU, Intersection over union; KBCE, Keypoint objectness binary cross entropy.

automated computer-assisted approaches, providing acupuncturists with efficient tool to enhance their practice. This research has the potential to significantly impact acupuncture practice by improving accuracy and consistency in acupoint localization, providing a more comprehensive detection system, and offering real-time assistance to practitioners.

2 Materials and methods

2.1 Landmark detection and proportional mapping approach

The MediaPipe framework (Lugaresi et al., 2019), developed by Google, has garnered considerable attention in the computer vision community due to its versatility and efficiency in building real-time applications. Initially designed for hand and face tracking, MediaPipe has expanded its capabilities to cater to a wide range of pose estimation and human body tracking tasks (Figure 1). The framework's ability to leverage deep learning models, coupled with its lightweight design, makes it an attractive choice for developing applications that require real-time performance on resource-constrained devices (Lugaresi et al., 2019). This attribute was the primary motivation for employing this framework in the present study. However, an important limitation is that MediaPipe does not provide access to the model architectures and parameters. So users cannot train the models from scratch on their own datasets. In the context of acupuncture point detection, by harnessing the framework's capabilities, it becomes possible to develop a real-time system that can efficiently identify acupuncture points on the human body, thereby enhancing the precision and effectiveness of acupuncture treatments. In this approach we landmark coordinate data generated by the MediaPipe framework to calculate proportional acupoint locations based on formulas guided by traditional acupuncture literature (World Health Organization, 2008; Focks, 2008; National University of Korean Medicine, Graduate School of Korean Medicine, Meridian and Acupoint Studies Textbook Compilation Committee, 2020).

2.1.1 Acupoint selection

A total of 38 acupoints were selected for localization including 18 acupoints on the hands and 20 acupoints on the face (Table 1). These acupoints were selected based on their common usage in clinical practice for a variety of conditions. The Supplementary Table S1, provides a summary of these acupoints included in the study, along with anatomical locations and key clinical usages.

2.1.2 Method

In order to identify the locations of over 38 acupoints, we utilized a combination of published literature regarding acupoint locations (World Health Organization, 2008; Focks, 2008), principles of oriental medicine, and the MediaPipe framework (v0.10.1).

The process involved first compiling a list of acupoint locations on the hands and face by referencing established acupuncture literature and standards (Supplementary Table S1). Then, each frame of the input video was captured through OpenCV computer vision library (v4.7.0.72) and converted to RGB format. The RGB frames were input into MediaPipe Face Mesh and Hand pipelines to acquire facial and hand landmark coordinates. These 468 facial and 21 hand landmarks per hand were used to mathematically estimate locations of key acupoints based on anatomical proportionality. Small dots were drawn on the original frames at the calculated acupoint locations using OpenCV drawing functions, representing acupoints. Finally, the output frame with overlayed acupoint dots was displayed to the user in real-time via OpenCV, allowing viewing of the acupoint tracking in the live video stream (Figure 2).

In more detail, the landmark detection framework provides the *X*, *Y*, and *Z* coordinates for each of the estimated anatomical landmarks. These 3D landmark points were used to mathematically calculate the locations of associated acupuncture points. Although MediaPipe predicts 21 hand landmarks (Figure 1a), accuracy constraints were encountered in projecting acupoints across different hand postures based on the literature guidelines. To overcome this, the hand postures were divided into four categories—front, inside, outside, and back views (Figure 3). To determine which posture the hand was in, three specific landmarks on the palm plane were selected (as shown in

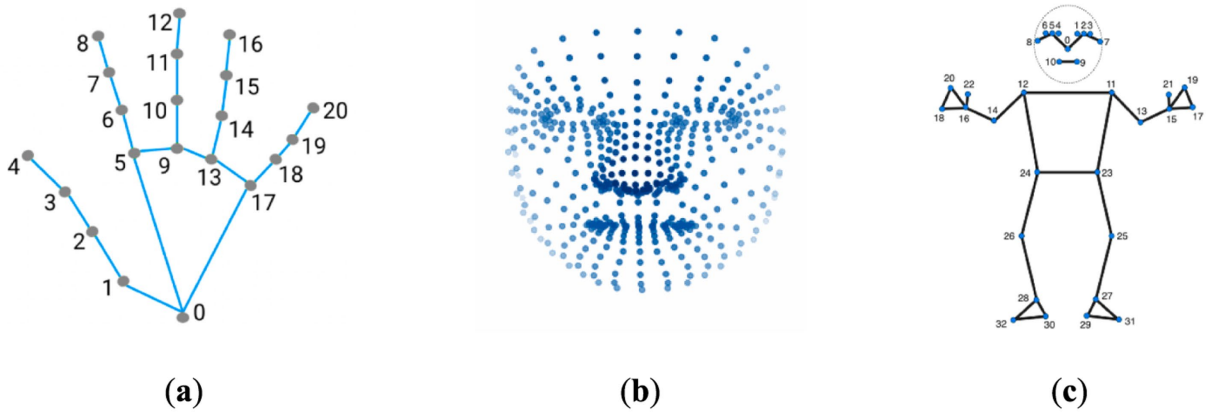


FIGURE 1
Keypoint localization examples using MediaPipe framework. **(a)** The MediaPipe Hand solution localizing 21 hand-knuckle coordinates within detected hand regions. **(b)** The MediaPipe Facemesh solution localizing 468 facial landmarks. **(c)** The MediaPipe Pose solution localizing 33 body landmarks. The figure demonstrates the capabilities of MediaPipe for anatomical keypoint localization across hands, faces, and bodies through the use of machine learning models tailored to each area.

TABLE 1 Acupuncture points selected for detection utilizing landmark detection framework.

Acupoint	Full name	Meridian	Acupoint	Full name	Meridian
Face[a]			Hand[a]		
CV-24	Chengjiang	Conception vessel	HT-7	Shenmen	Heart
BL-1	Jingming	Bladder	HT-8	Shaofu	Heart
BL-2	Cuanzhu	Bladder	HT-9	Shaochong	Heart
GB-1	Tongziliao	Gallbladder	LI-1	Shangyang	Large intestine
GB-2	Tinghui	Gallbladder	LI-2	Erjian	Large intestine
GB-14	Yangbai	Gallbladder	LI-3	Sanjian	Large intestine
GV-25	Suliao	Governing vessel	LI-4	Hegu	Large intestine
GV-26	Shuigou	Governing vessel	LU-11	Shaoshang	Lung
GV-27	Duiduan	Governing vessel	LU-9	Taiyuan	Lung
LI-19	Kouheliao	Large intestine	PC-9	Zhongchong	Pericardium
LI-20	Yingxiang	Large intestine	SI-1	Shaoze	Small intestine
SI-18	Quanliao	Small intestine	SI-2	Qiangu	Small intestine
ST-1	Chengqi	Stomach	SI-3	Houxi	Small intestine
ST-2	Sibai	Stomach	SI-4	Wangu	Small intestine
ST-3	Juliao	Stomach	TE-1	Guanchong	Triple energizer
ST-4	Dicang	Stomach	TE-2	Yemen	Triple energizer
ST-5	Daying	Stomach	TE-3	Zhongzhu	Triple energizer
ST-6	Jiache	Stomach	TE-4	Yangchi	Triple energizer
ST-7	Xiaguan	Stomach			
TE-23	Sizhukong	Triple energizer			

Provides the standard name and abbreviation for each acupuncture point.
[a]More detail provided in the Supplementary Table S1.

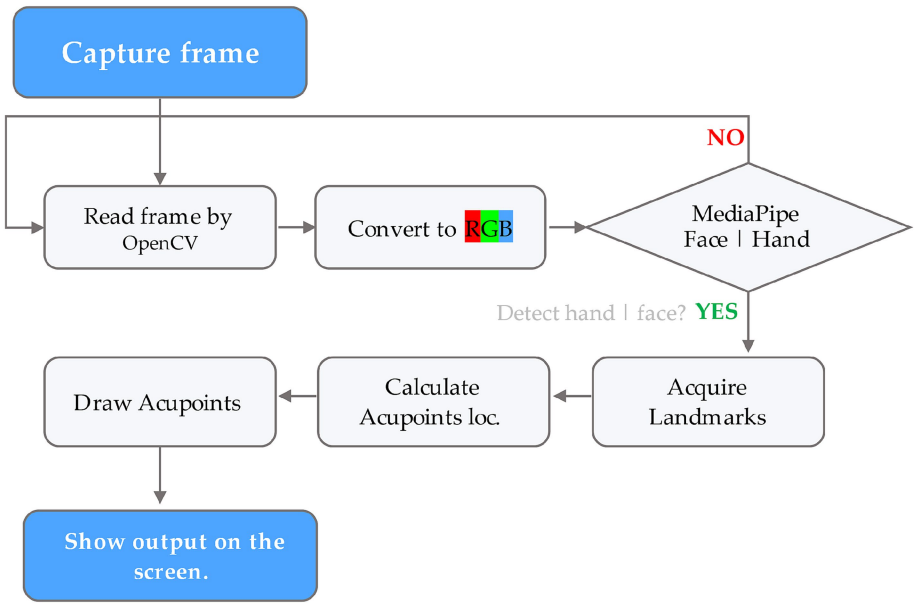


FIGURE 2
Flowchart illustrating the acupoint detection pipeline using the MediaPipe framework.

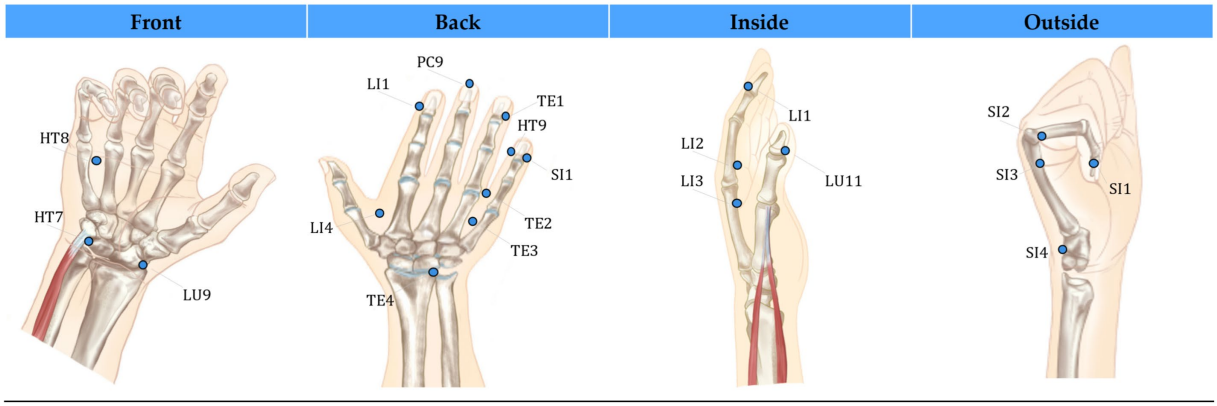


FIGURE 3
Visualization of hand acupuncture points organized by four postures. Segmenting points by posture enables clear visualization and access across hand surfaces (Images of hand from the source National University of Korean Medicine, Graduate School of Korean Medicine, Meridian and Acupoint Studies Textbook Compilation Committee, 2020).

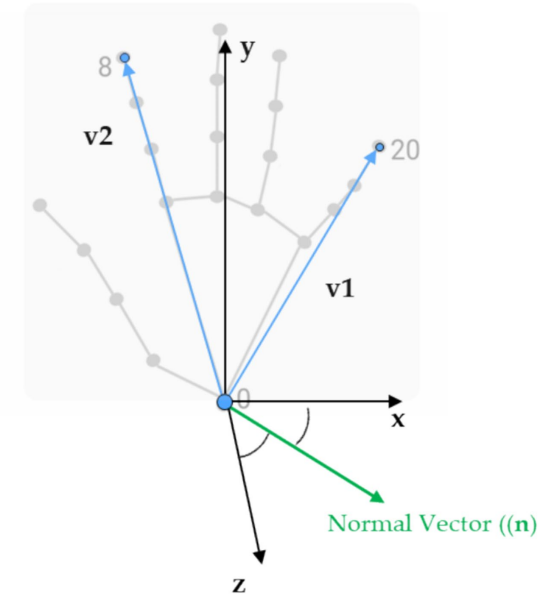


FIGURE 4
The 3D landmarks we used and the specific ones selected to calculate the palm normal and the angle that determined hand postures.

Figure 4), with one landmark serving as the reference point. Vectors were calculated from this reference point to the other two landmarks. Taking the cross product of these two vectors produced the palm's 3D orientation vector. The angle between this palm vector and the global *Z*-axis was computed using the dot product. This angle measurement enabled classifying the hand into one of the four posture categories based on how much it diverged from the *Z*-axis orientation.

A similar methodology was utilized to model the face. The facial region was divided into three key postures—center, left, and right—in order to account for horizontal head rotation. Each of these three poses had a specific set of facial landmarks that were visible and could be detected. The proportional distances and angles between these landmarks (calculated using Equations 1 and 2) are then used to mathematically derive the predicted locations of associated acupoints.

For example, the coordinates of the HT8 (Shaofu) acupoint, which is located on the palm of the hand, in the depression between the fourth and fifth metacarpal bones, proximal to the fifth metacarpophalangeal joint, is calculated in relation to the distance of landmarks 5 and 17 as shown in Figure 5. The Euclidean distance between these skeletal landmarks is first computed (base_distance). Next, based on a proportional measurement, the distance from HT8 to the point between landmarks 13 and 17 is calculated as 1/5 of that length (base_distance) toward landmark 0. Finally, we project the HT8 coordinates at the proper location along the palm.

$$d = \| \overrightarrow{Pi} - \overrightarrow{Pj} \| = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \tag{1}$$

$$\theta = \cos^{-1}\left(\frac{\overrightarrow{Pi} \cdot \overrightarrow{Pj}}{\| \overrightarrow{Pi} \| \cdot \| \overrightarrow{Pj} \|} \right) \tag{2}$$

Note that for facial acupuncture points, there are a greater number of anatomical landmarks (468) that can be used as reference points, which makes estimating the acupoint locations on the face easier compared to hand region with fewer identifiable landmarks. In addtion, there is no anatomical landmarks that can be reliably used as reference points for locating acupoints like LI11, LI10, and TE5 that located on forearm. Thus, this work focuses on acupoint localization for the hand given the greater challenges in accurately identifying forearm acupoints without established anatomical landmark provided by MediaPipe hand or pose estimation model.

In essence, classical acupuncture proportional methods are translated into computational geometric transformations in order to map key reference points on the body to known acupoint locations based on their relative positions. Further optimization of these formulaic projection techniques could enhance precision.

Step 1: Get the location of $\mathbf{p}_0$, $\mathbf{p}_5$, $\mathbf{p}_{13}$, $\mathbf{p}_{17}$, and $\mathbf{p}_{mid\_17\_13}$ in the image space.

Step 2: Calculate the total distance between $\mathbf{p}_0$ and $\mathbf{p}_{mid\_17\_13}$:

$$\text{total distance} = \| \mathbf{p}_0 - \mathbf{p}_{mid\_17\_13} \|$$

Step 3: Calculate the fraction based on the given distance:

$$\text{base distance} = \|\mathbf{p}_5 - \mathbf{p}_{17}\|$$

$$\text{fraction} = \alpha \times \text{base distance}$$

Step 4: Find the $\mathbf{p}_{HT8}$:

$$\mathbf{p}_{HT8} = \mathbf{p}_{mid\_17\_13} + \left(\frac{\mathbf{p}_0 - \mathbf{p}_{mid\_17\_13}}{\text{total distance}} \right) \times \text{fraction}$$

** The operation $\|\cdot\|$ represents the norm (magnitude) of a vector, and the × symbol denotes scalar-vector multiplication.

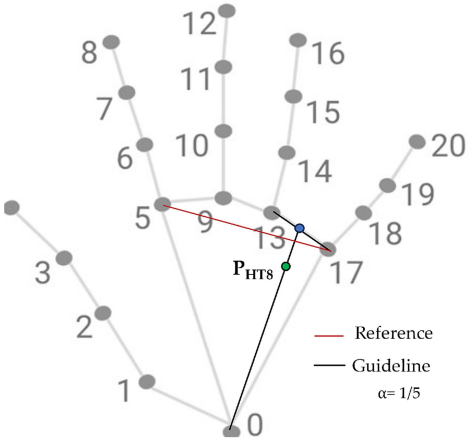


FIGURE 5
Example approach to localize an acupoint (HT8).

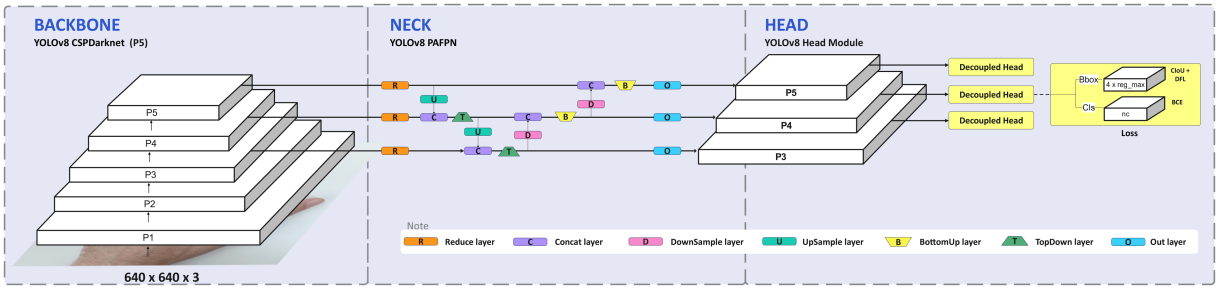


FIGURE 6
YOLOv8 architecture. The head can be decoupled to process objectness, classification, and regression tasks independently (Jocher et al., 2023; MMYOLO Contributors, 2023).

2.2 Data-driven pose estimation approach

In addition to the proportional mapping approach, a data-driven deep learning model based on YOLOv8-pose was also developed to provide a comparative solution. Ultralytics released a version of the YOLO object detection model, providing state-of-the-art accuracy and speed for detection tasks. This version of YOLO has the same overall architecture (Figure 6) as previous versions, but it includes many enhancements compared to earlier iterations. It uses a new neural network design that combines feature pyramid network (FPN) and path aggregation network (PAN) architectures (Jocher et al., 2023). YOLO models are generally known for their computational efficiency and real-time performance, which aligns with the study's goal of developing a real-time acupuncture point detection system.

YOLOv8 comes in 5 sizes and expands the capabilities beyond just detection to also include segmentation, pose estimation, tracking and classification. This new comprehensive computer vision system aims to provide an all-in-one solution for real-world applications (Terven et al., 2023). The YOLOv8 architecture leverages a convolutional neural network (Terven et al., 2023) to spatially localize and predict keypoints within the images. However, an official paper has yet to be released. We implement the code from the Ultralytics repository (Jocher et al., 2023).

2.2.1 Dataset collection and preprocessing

To train a real-time acupoint detection model, we collected a dataset comprising 5,997 acupoint-annotated images of arms at a resolution of 1,488 × 837 pixels. These images were sourced from 194 participants (49 male, 45 female, age range 19–68 years) at Pukyong National University and Dongshin University in South Korea, captured in a controlled laboratory environment with a white background. The dataset contains annotations marking five common acupoints on arm and hand—LI4 (Hegu), TE3 (Zhongzhu), TE5 (Waiguan), LI10 (Shousanli) and LI11 (Quchi)—localized according to the standard acupuncture point locations in the Western Pacific Region defined by the World Health Organization (Sulong and Randles, 2023) and verified by experts in oriental medicine (Figure 7). The annotations include bounding boxes around each arm and keypoint locations for the acupoints. The annotations were done using the COCO Annotator tool (Stefanics and Fox, 2022).

The data was then split into a training set of 5,392 images and a validation set of 605 images. A limitation of this dataset is that the arm poses and sizes are relatively uniform, lacking diversity. To help mitigate this, data augmentation techniques like rotation, scaling, and cropping applied on-the-fly to the training images to increase the diversity of the training data. Supplementary Figure S1 provides example input images from the dataset used by the model.

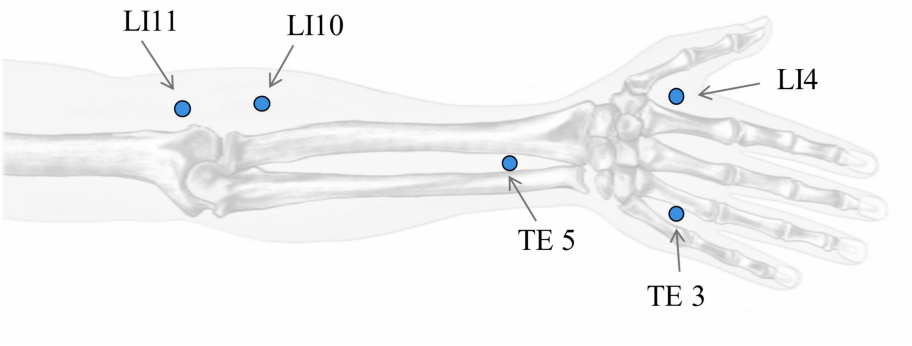


FIGURE 7
Location of acupoints on the hand. Shown are LI4 (Hegu), TE3 (Zhongzhu), TE5 (Waiguan), LI10 (Shousanli), and LI11 (Quchi). (Hand image reproduce from source National University of Korean Medicine, Graduate School of Korean Medicine, Meridian and Acupoint Studies Textbook Compilation Committee, 2020).

TABLE 2 Parameter settings for model training.

Parameters[a]	Values
Image size	640 × 640
#Epochs	300
#Batch-size	16
Initial learning rate	0.01
Main optimizer	SGD
Loss	CIoU_loss + DFL + Kobj_ BCE + KIoU_loss[b]

[a]Full list of parameters used provided in Supplementary param.yaml file.
[b]IoU, intersection over union; CIoU, complete IoU; DFL, distribution focal loss; KBCE, keypoint objectness binary cross entropy; KIoU, Keypoint IoU (keypoint oks loss).

A limitation of this dataset is that the arm poses and sizes are relatively uniform, which may restrict the model's ability to generalize to real-world scenarios with greater variability. To mitigate this, we employed data augmentation techniques during training. These techniques included rotation, scaling, and cropping, which were applied on-the-fly to the training images. This process introduced artificial variations in arm poses and sizes, enhancing the model's exposure to a wider range of potential inputs. To minimize the impact of potential similarity between images from the same participant, we split the dataset into training and validation sets based on participants. The data was then split into a training set of 5,392 images and a validation set of 605 images. While these measures were taken to enhance the dataset's diversity and mitigate potential biases, it is important to acknowledge that the validation process may still be limited by the relatively controlled nature of the data. Further evaluation on a more diverse dataset with a wider range of arm poses and sizes would be beneficial for a comprehensive assessment of the model's generalizability. Supplementary Figure S1 provides example input images from the dataset used by the model.

2.2.2 Model training and evaluation metrics

We decide to implement transfer learning and initialize our models with pre-trained weights from YOLOv8l-pose (large), which was pre-trained on human pose estimation using the COCO dataset. Evaluated on COCO Keypoints validation 2017 dataset, YOLOv8l-pose achieved an mAP_{50-95} of 67.6% and mAP_{50} of 90.0% with an image size of 640 pixels (Jocher et al., 2023). We then begin fine-tuning this base model on our custom dataset of acupoints on arm and hand images that as mentioned was split into a 90% training set and 10% validation set to adapt the model to specifically identify acupoints on hands. This transfer learning approach allows us to leverage the representations learned by the pre-trained YOLOv8-pose model to accelerate training on our more specialized acupoint detection task. In addition, it's clear that a diverse dataset is crucial for deep learning models to make precise predictions. To enhance the performance of our pose estimation model, we implemented various data augmentation techniques. The augmentations we implemented were horizontal flipping of the images, rotation by varying degrees, mixup which combines samples through linear interpolation, and Mosaic augmentation that stitches together regions from multiple samples. These methods increased the diversity of our training data, which helped the model learn more robust features and improved accuracy.

For implementation, we utilized an Nvidia RTX 4090 GPU with 24GB RAM to efficiently train the acupoint detection model. Table 2 outlines the key training parameters used in the training process.

After model training was complete, several validation metrics were used to evaluate the performance of the acupoint detection model, including distance error (E), precision, recall, mean average precision (mAP), and object keypoint similarity (OKS), as outlined in Equations 3–7. The error E between the predicted acupoint position P_{pred} and the annotated ground truth acupoint position P_{gt} is defined as the Euclidean distance between them in the image space. The OKS metric specifically measures the similarity between predicted and ground truth keypoints, which is relevant for evaluating acupoint detection performance.

The specific formulas for calculating these metrics are:

$$E = \| P_{\text{pred}} - P_{\text{gt}} \| \tag{3}$$

$$\text{Precison} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\% \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\% \quad (5)$$

$$\text{mAP} = \frac{\sum_{i=1}^{C} AP_i}{C} \quad (6)$$

$$\text{OKS} = \exp\left(-\frac{d_i^2}{2s^2k_i^2}\right) \quad (7)$$

where: TP = true positives, FP = false positives, FN = false negatives, TN = true negatives, C = total number of categories, AP_i = average precision for the ith category and mAP was calculated as the mean of average precision scores across all categories, to summarize the model's overall precision. For each predicted keypoint, the OKS is calculated based on the Euclidean distance between the predicted and ground truth keypoint (d_i), adjusted by the scale (s) which normalizes for object size, and a per-keypoint constant (k) that controls falloff. In our dataset, we used a constant k value of 0.02 for all keypoints. The OKS scores can then be averaged across keypoints and images to evaluate overall localization performance.

These metrics were computed on a validation set to evaluate the performance of the acupoint detection model after training.

3 Results

3.1 Landmark detection and proportional mapping approach

Through integrating principles of oriental medicine, literature references, and the MediaPipe framework, real-time performance in localizing 38 acupoints was accomplished in this study. Figures 8, 9 presents exemplary outcomes, demonstrating the proficiency of the proposed approach in detecting acupoints across various postures. Additionally, Supplementary Videos S1, S2 provide more extensive examples showcasing acupoint detection across a wide range of motions and poses.

We only evaluated the accuracy of our proposed model using a subset of 188 images from the larger dataset mentioned previously, which included 8 acupoints localization. These 188 images contain annotated acupuncture points that serve as ground truth landmarks. The images have annotations for 8 common acupoints on the back of the hand: LI4 (Hegu), TE3 (Zhongzhu), SI1 (Shaoze), HT9 (Shaochong), TE1 (Guanchong), PC9 (Zhongchong), LI1 (Shangyang), and TE2 (Erjian), These acupoints were selected for evaluation because of their frequent utilization in acupuncture therapy.

Quantitative evaluation of model performance utilized the Euclidean distance metric (see Equation 3) to compute error between predicted and ground truth acupoint coordinates across all images of dataset. The average distance error achieved by this method was less than 10 pixels over all annotated landmarks (see Figure 10b). The low average distance error signifies that the predicted acupoint locations from the model closely correspond to the true anatomical locations demarcated by experienced practitioners.

We also analyzed the errors for localizing each individual acupoint location as shown in Figure 10. The box plots summarize the distribution of errors over all test images for each point. The median error varied based on the size and distinguishability of each point, ranging from ~4.0 pixels for the prominent PC9 acupoint to ~9.0 pixels for the TE3 acupoint. These results demonstrate that the model can detect acupoint near fingertips with high accuracy, localizing them within ~10 pixels for the majority of validation cases. These pixel-level errors correspond to approximately sub-centimeter accuracy in real-world coordinates.

To convert pixel errors to real-world coordinates, we used a simple calibration method. The images utilized for validation in this analysis were captured at a resolution of 1,488 × 837 pixels. A sheet with known horizontal length of approximately 80 cm was placed in the scene as a scale reference. This sheet spanned roughly 1,488 pixels

FIGURE 8
Example result of showing acupoints on the face and hand.

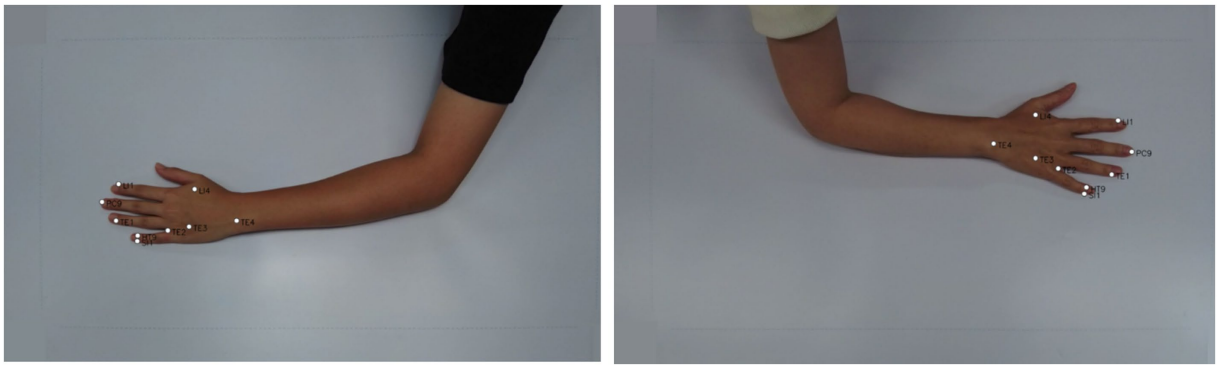


FIGURE 9
Exemplary images from dataset with landmark-based model outputs depicting acupoint locations on the back side of the hand, including LI4, TE1, TE2, TE3, LI1, PC9, SI1, HT9.

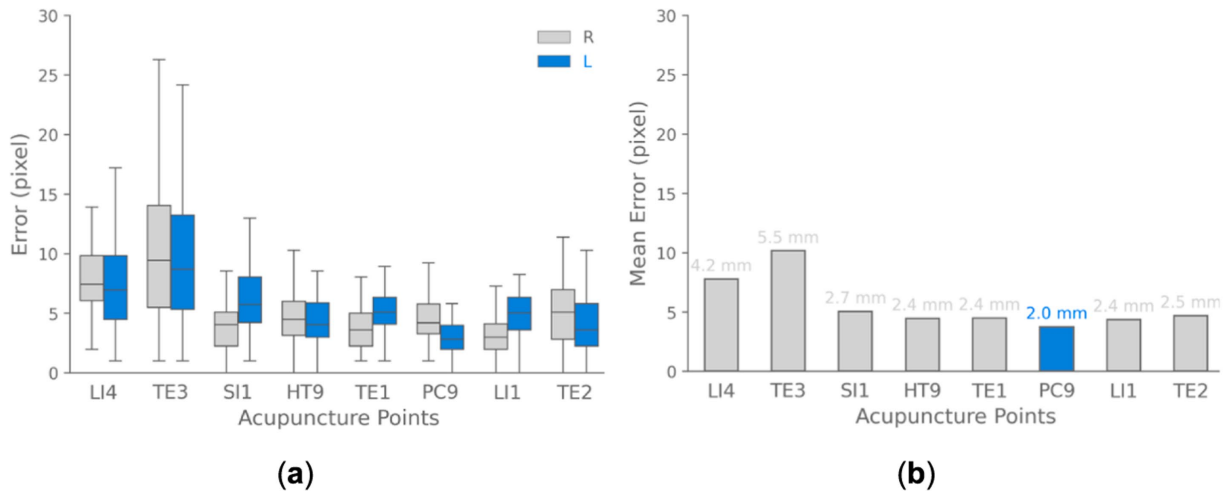


FIGURE 10
Acupoint localization accuracy landmark-based approach. **(a)** Boxplots depicting the distribution of Euclidean distance between predicted and ground truth acupoint locations for each evaluated acupoint separated for each hand (right and left). **(b)** Bar chart visualizing the mean of localization errors across different acupoints. The results demonstrate that the majority of points are localized with sub-centimeter accuracy.

horizontally across the image. Using the known real-world length and corresponding pixel length, we estimated a conversion factor of approximately 0.0537 cm per pixel. Utilizing this pixel-to-physical space calibration, the quantified pixel-level errors can be translated to real-world spatial errors with approximately sub-centimeter accuracy. With this calibration, for example, a pixel error of 10 pixels would translate to around 5.37 mm error in real-world coordinates.

To further assess the accuracy of predicted acupoint coordinates, we expanded our evaluation beyond the Euclidean distance metric. This comprehensive approach incorporates multiple statistical measures and visualizations, providing a more understanding of the model's performance. In addition to the average distance error reported earlier, we calculated confidence intervals, and conducted statistical Kolmogorov–Smirnov tests to examine the significance of differences between predicted and actual coordinates as shown in Table 3. The test is a non-parametric statistical test that compares two distributions to see if they differ significantly. The mean distance between actual and predicted points is 5.58 pixels, with a narrow 95% confidence interval (5.38, 5.78), reflecting high accuracy. The Kolmogorov–Smirnov tests for both X and Y axes yield statistics of 0.010 and 0.012, respectively, with p-values of 1.000, suggesting that the error distributions are well-matched to the expected distributions.

TABLE 3 **Summary of landmark-based approach model performance metrics and statistical tests.**

Metric/test	Value (pixel)/ statistic	95% CI/p-value
Mean distance	5.58	5.38–5.78
Kolmogorov–Smirnov test (X-axis)	Statistic = 0.010	$p = 1.000$
Kolmogorov–Smirnov test (Y-axis)	Statistic = 0.012	$p = 1.000$

The model's performance was assessed using multiple visualizations as shown in Figure 11. The scatter plot of actual vs.

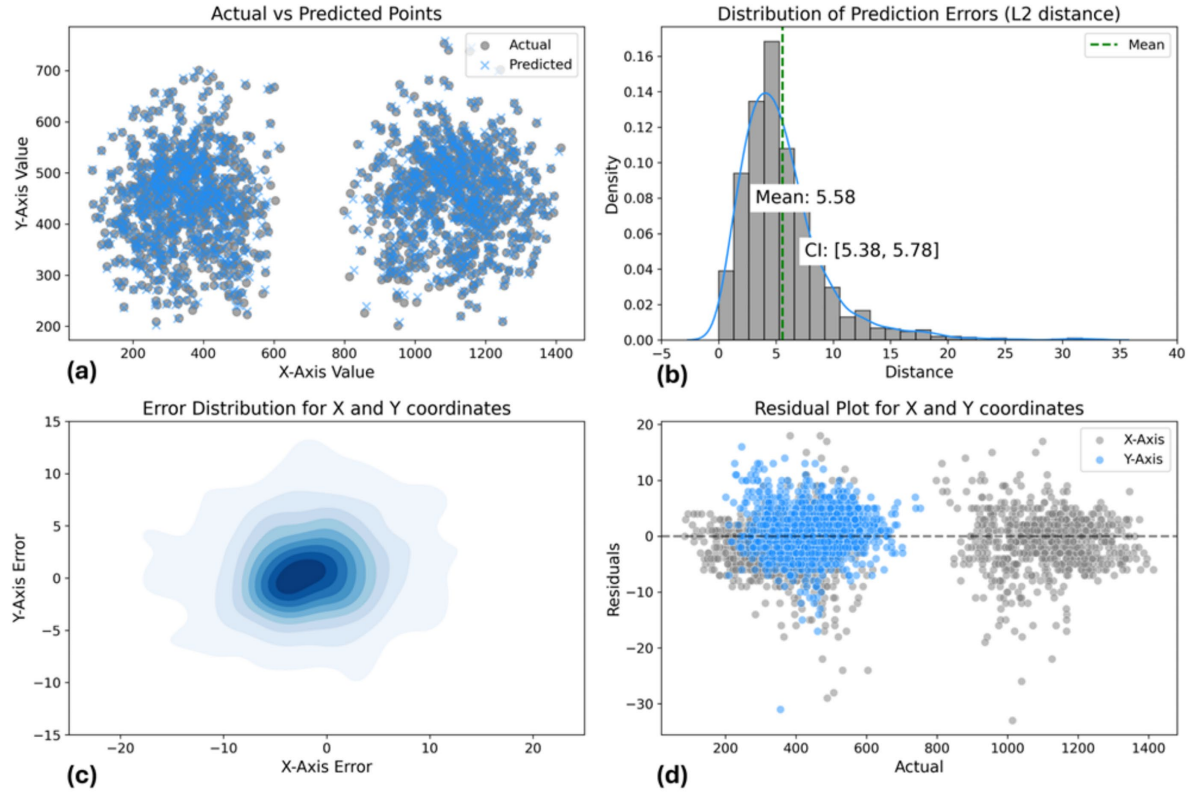


FIGURE 11
Landmark-based approach model performance evaluation. **(a)** Actual vs. predicted points scatter plot. **(b)** L2 distance error distribution [mean: 6.81 pixels, CI: (6.65, 6.98)]. **(c)** 2D error distribution for *X* and *Y* coordinates. **(d)** Residual plot showing prediction errors across coordinate range.

predicted points demonstrates a strong overall correspondence, with predicted points (blue) closely overlapping actual points (gray) across the coordinate space. The distribution of prediction errors reveals a mean L2 distance of 5.58 pixels, with a tight 95% confidence interval of (5.38, 5.78), indicating consistent accuracy. The error distribution for *X* and *Y* coordinates, visualized as a 2D density plot, shows a concentrated, symmetric pattern centered around zero, suggesting unbiased predictions. The residual plot further supports this, displaying a relatively even spread of errors around the zero line for both *X* and *Y* axes, with most residuals falling within ±10 pixels. Notably, there's a clear separation in the residual values for the *X*-axis. This is due to the acupoints being predominantly associated with either the left or right hand, leading to distinct coordinate predictions based on hand position. Overall, these results demonstrate the model's high precision in predicting spatial coordinates, with a small average error and well-suited error distributions across the prediction space.

3.2 Data-driven pose estimation approach

Data-driven pose estimation model achieves good accuracy for acupoint localization given the constraints of this dataset. The results validate the effectiveness of YOLOv8-pose for this medical imaging application and computer vision task.

Figure 12 visualizes two example outputs on the validation set for acupoint localization. More examples of validation batch results are shown in the Supplementary Figures S3, S4. Additionally, videos demonstrating the model's acupoint localization on full motion sequences are provided in Supplementary Video S3. Qualitatively, YOLOv8 appears able to predict acupoint locations that closely match the ground truth in this controlled dataset. Some slight variations are visible upon close inspection, but overall YOLOv8-pose demonstrates acceptable performance for this acupoint localization task.

Quantitatively, YOLOv8-pose demonstrates high performance on acupoint localization as evidenced by high mean Average Precision (mAP) scores on the validation set. Specifically, it achieves an mAP at OKS 50% of 0.99 and 50–95 of 0.76 pose estimation. The complete quantitative results while training are presented in Supplementary Table S2. These high mAP values indicate that the model is able to accurately localize and identify acupoints in the validation images. Table 4 summarizes the model's localization accuracy for each acupoint by reporting the Mean distance error in mm between the predicted and true acupoint positions. Note that to convert from pixels to mm, the pixel-to-mm conversion approach outlined in section 3.1 was used.

Furthermore, we calculated confidence intervals and conducted statistical tests to evaluate differences between predicted and actual coordinates, as shown in Table 5. The mean distance between actual and predicted points is 6.81 pixels, with a 95% confidence interval of (6.65, 6.98), indicating good accuracy. The Kolmogorov–Smirnov tests for the *X* and *Y* axes yield statistics of 0.010 and 0.015, with *p*-values of 0.997 and 0.906, respectively, suggesting well-matched error distributions.

The results visualized in Supplementary Figure S2 show the loss and accuracy curves for both the training and validation data across

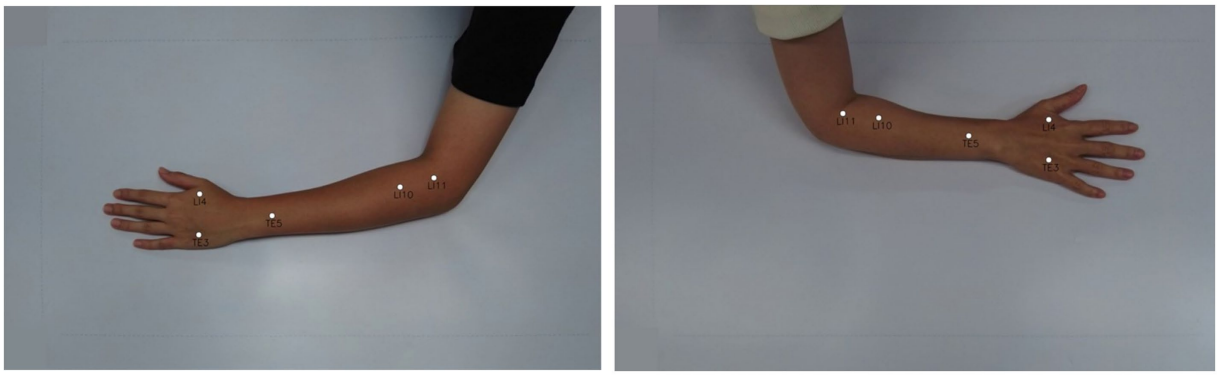


FIGURE 12
Acupoint localization accuracy of data-driven pose estimation approach. Shows the predicted acupoint locations from YOLOv8-pose.

TABLE 4 Performance of YOLOv8-pose on the custom dataset of arm acupoints after 300 training epochs with an input size of 640 × 640 pixels.

Model	mAP^{val}_{50} (pose)	mAP^{val}_{50-95} (pose)	LI11 (mm)	LI10 (mm)	TE5 (mm)	LI4 (mm)	TE3 (mm)
YOLOV8l-pose (Pretraind)	0.99	0.76	3.68	3.97	4.14	2.93	3.82
			±(2.44)	±(2.65)	±(2.89)	±(2.59)	±(3.08)

TABLE 5 Summary of data-driven pose estimation approach model performance metrics and statistical tests.

Metric/test	Value (pixel)/ statistic	95% CI/*p*-value
Mean distance	6.81	6.65–6.98
Kolmogorov–Smirnov test (*X*-axis)	Statistic = 0.010	$p = 0.997$
Kolmogorov–Smirnov test (*Y*-axis)	Statistic = 0.015	$p = 0.906$

training epochs. As demonstrated, the training and validation results showed that the YOLOv8-pose model for acupoint detection exhibited good convergence for this dataset. Specifically, the loss curve declined rapidly then flattened, indicating effective optimization. Meanwhile, the precision, recall, and mAP metrics increased quickly then stabilized, demonstrating model performance on the validation set.

Additionally, Figure 13 illustrates the acupoint localization accuracy achieved by the YOLOv8-pose model. Boxplots in panel (a) show the distribution of Euclidean distance errors between predicted and ground truth locations for each acupoint. The bar chart in panel (b) visualizes the mean of localization errors.

Regarding Figure 14, the scatter plot shows strong alignment between actual and predicted points, with a mean L2 error of 6.81 pixels and a 95% confidence interval of (6.65, 6.98), indicating consistent accuracy. The 2D density plot reveals a symmetric error distribution centered around zero, though some variability is observed. The residual plot highlights errors within ±10 pixels. Overall, the model performs well, but improvements could be made in reducing prediction variability and enhancing accuracy for points farther from the center.

3.3 Application development

To demonstrate the practical application of these models, a simple desktop application was developed using Tkinter, a Python library for creating graphical user interfaces. The application allows users to use a webcam feed for real-time acupoint localization. Upon camera activation, the landmark detection framework or the fine-tuned pose estimation model processes the input, and the identified acupoints are visualized on the screen.

The application provides an intuitive interface for users (Figure 15). Further enhancing user experience, the application allows practitioners to choose specific acupoints of interest. Once the webcam is activated and the desired model and acupoints are selected, the system processes the live video feed. Identified acupoints are then dynamically overlaid onto the displayed image, providing practitioners with precise visual guidance. Looking ahead, we plan to expand the application's capabilities to include acupoint localization on the legs, which are crucial for treating conditions affecting the lower body regions. Overall, this straightforward and user-friendly application showcases the potential of integrating AI-based acupoint localization into acupuncture treatments.

4 Discussion

This study investigated the feasibility of leveraging computer vision techniques to automate the localization of acupuncture points on the face and hands. Our study focused on these areas due to their frequent use in clinical practice and relative accessibility for imaging. These areas offer a high density of commonly used acupoints in a compact region, facilitating efficient data collection and analysis. Specifically, we explored two distinct approaches: Utilizing a real-time landmark detection framework to identify anatomical keypoints and map acupoint locations based on classical proportional

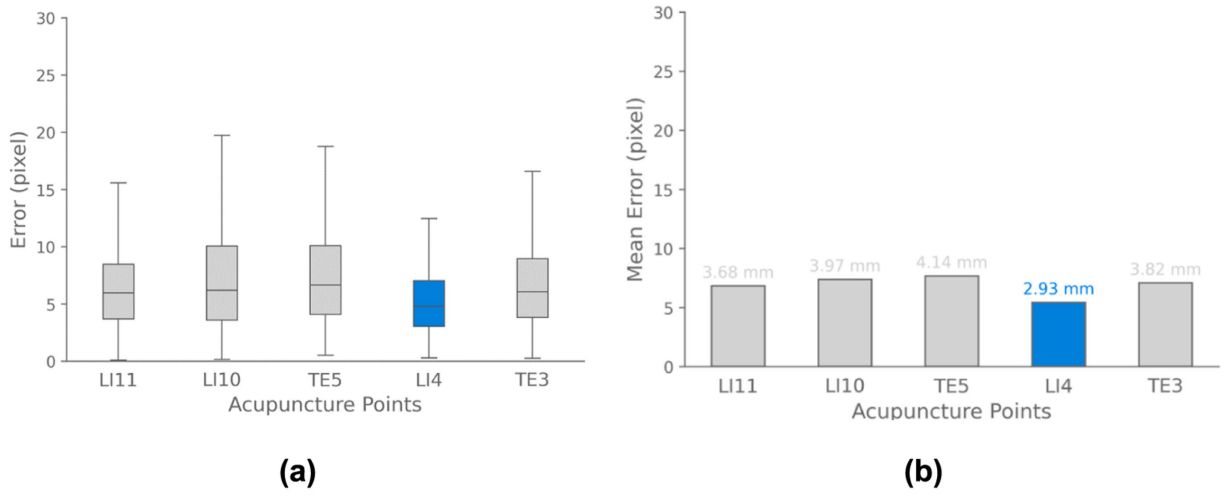


FIGURE 13
Acupoint localization accuracy for data-driven pose estimation approach. **(a)** Boxplots depicting the distribution of Euclidean distance errors between predicted and ground truth acupoint locations for each evaluated point. **(b)** Bar chart visualizing the mean localization errors across different acupoints. The results demonstrate that the majority of points are localized with sub-centimeter accuracy.

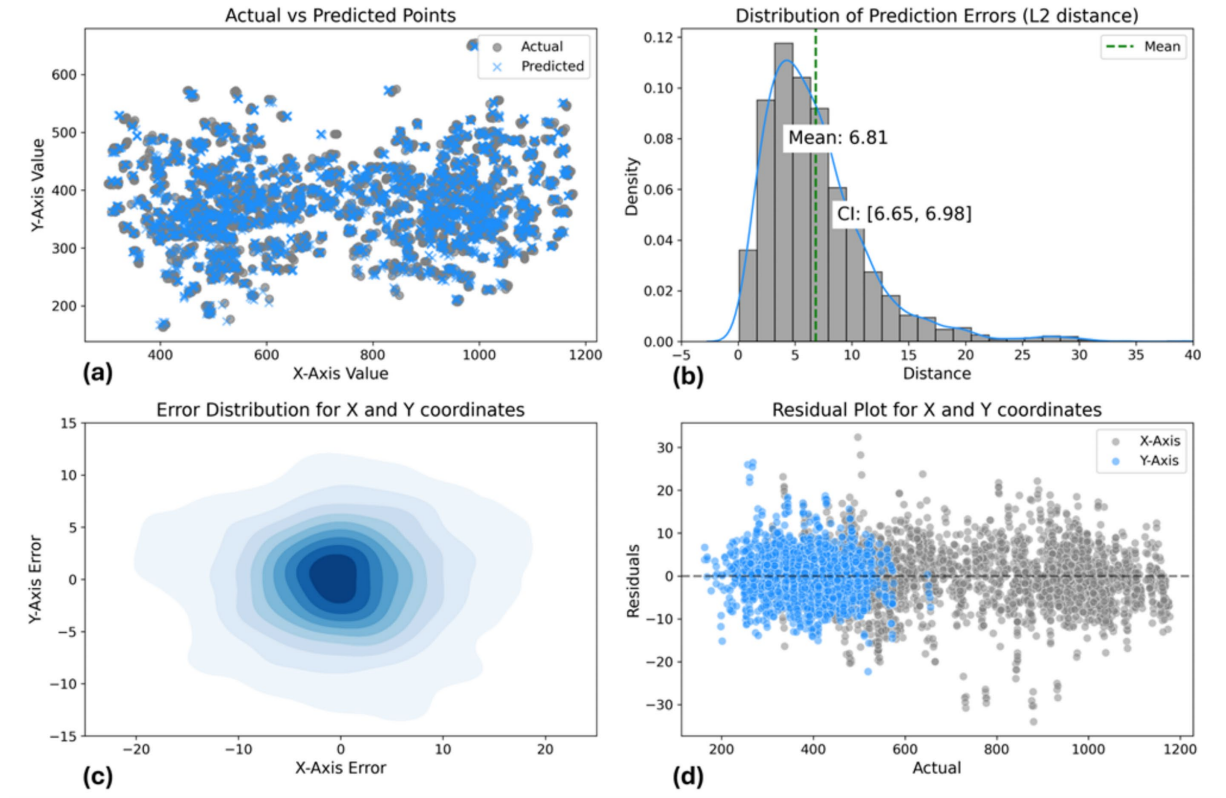


FIGURE 14
Data-driven pose estimation approach model performance evaluation: **(a)** Actual vs. predicted points scatter plot. **(b)** L2 distance error distribution [mean: 6.81 pixels, CI: (6.65, 6.98)]. **(c)** 2D error distribution for *X* and *Y* coordinates. **(d)** Residual plot showing prediction errors across coordinate range.

measurement methods from acupuncture literature, and; fine-tuning a state-of-the-art pose estimation model on a custom dataset to directly predict acupoint coordinates through data-driven deep learning. Both methodologies demonstrated promising results in accurately identifying and visualizing acupuncture points in real-time settings.

The proposed landmark-based approach effectively detected anatomical keypoints to visually guide acupoint positioning. This

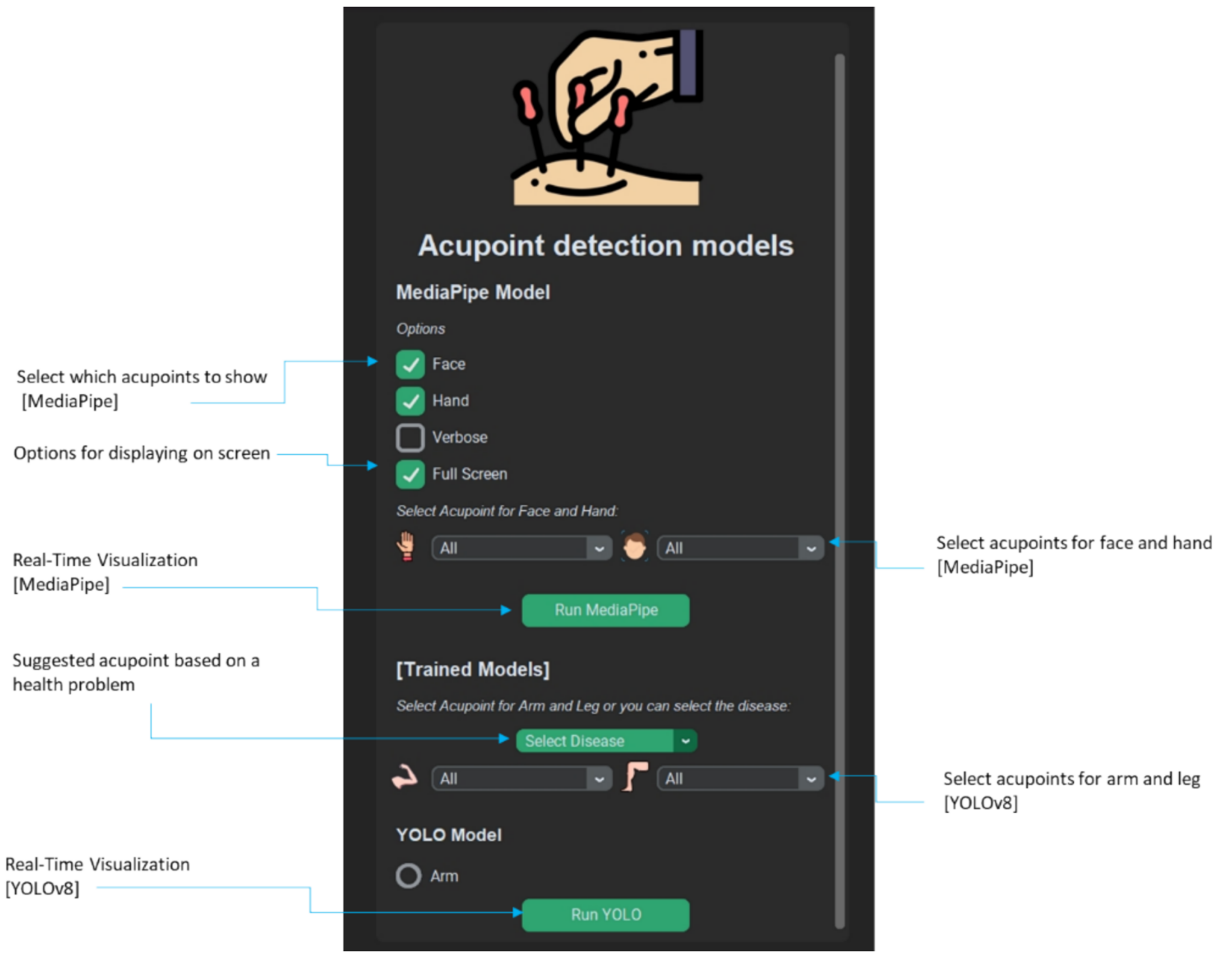


FIGURE 15
Interactive desktop application for real-time acupoint localization using pose estimation models on webcam feed (Additional usage examples provided in the Supplementary Figure S5 and Supplementary Video S4).

could enable innovative acupuncture-assistive tools by providing practitioners with rapid on-screen guidance. A key advantage of our approach is the integration of the efficient framework, enabling real-time landmark detection and proportional mapping of acupoints across various hand and facial poses. As the architecture and training details of the MediaPipe models are proprietary, we cannot replicate the models directly through training our own model from scratch. Therefore, the methodology applies classical formulas to anatomically map and proportionally estimate acupoint locations. It can be easily adapted to different hand and face postures and allows for real-time visualization of acupoints. However, challenges remain in improving resilience to scale and rotation variances and using it for different body parts not accounted for in the original framework models. Qualitative assessment shows that accuracy depends heavily on the ability to reliably detect and track key anatomical landmarks. This can introduce some inaccuracies into the system's final output. Furthermore, mathematical transformations may lack adaptability across heterogeneous populations. For a solution, transitioning to data-driven machine learning techniques could potentially address these limitations. The qualitative results show that the acupoints on the face have more invariance to transitional and rotational movement, which may be due to the higher number of landmarks in that region (468 keypoints). The quantitative results for hand dataset of 188 validation images reveal that the accuracy is higher for areas around the fingertips. This suggests it is easier for the model to locate these points compared to acupoints in the middle of the back of the hand.

In contrast to classical mapping techniques, data-driven deep learning approaches like the one employed by Sun et al. (2020, 2022) using U-Net and HRNet architectures can learn acupoint features directly from data. Our pose estimation model achieved acceptable acupoint localization accuracy (mAP at OKS 50–95% = 0.76, Mean error less than ~5 mm) in constrained arm dataset images for locating five acupoints. Compared to Sun et al.'s method that detected only 2 acupoints, our model localized 5 hand acupoints with high precision. Nevertheless, from Figure 10 it is evident that that the majority of the predicted acupoint locations for LI4, TE3, and LI11 exhibit high accuracy when approximately converted from pixel coordinates to physical distances based on the defined pixel-to-cm conversion factor. However, acupoints LI10 and TE5 exhibit higher localization errors compared to other acupoints. This suggests these two acupoints are more challenging for the model to precisely predict, perhaps due to greater variability in their location or appearance in the dataset. Further analysis of these acupoints may be needed to improve localization performance. While this model demonstrates acceptable performance

for acupoint localization for controlled lab images, it may not generalize as well to more complex real-world hand images with cluttered background compared to MediaPipe that their model for hand landmarks trained on more than 30,000 images of hand. These initial results are encouraging, further evaluation is needed to determine how the model generalizes to real-world settings outside of the lab.

However, a significant limitation shared across studies in this scope, including ours, is the lack of large, publicly available datasets with expert-annotated acupoints. This hinders the ability to benchmark and compare the performance of different computer vision models specifically designed for acupuncture point detection, as we do not have access to comparable datasets or models (Sun et al., 2020; Sun et al., 2022; Chen et al., 2021). Finally, in this study the quantitative evaluation of facial acupoint localization was not addressed due to lack of a dataset of faces with known acupoint locations in this study, presenting an area for future investigation. Combining data-driven techniques with domain expertise in oriental medicine can pave the way for more advanced and integrative acupoint recognition systems.

Looking ahead, integrating the 3D capabilities of real-time landmark detection framework (including depth information) could enable 3D acupoint visualization and localization, a capability not explored in prior works. In contrast, the pose estimation currently lacks support for depth estimation, presenting an area for potential enhancement. Furthermore, investigating few-shot learning or domain adaptation techniques could enhance the generalization of data-driven models to handle real-world diversity beyond limited training data.

5 Conclusion

In conclusion, this study explored the potential of leveraging computer vision techniques for automating the localization of acupuncture points on the face and hands. Two distinct approaches were investigated: (1) utilizing a real-time landmark detection framework to map acupoints based on anatomical landmarks and proportional measurements, and (2) fine-tuning a state-of-the-art pose estimation model on a custom dataset for direct acupoint detection. The landmark-based system showed promising real-time acupoint visualization capabilities but had limitations due to potential landmark detection inaccuracies and rigid mapping formulas. The pose estimation model achieved sub-centimeter mean localization accuracy when fine-tuned on a controlled dataset but may face performance degradation in complex, real-world scenarios beyond the training dataset constraints. While both methodologies exhibit encouraging preliminary results, several challenges persist. These include the lack of large, diverse datasets for training and benchmarking acupoint detection models, as well as the need for further generalization and robustness to real-world variations. To address these challenges, our future work plans to curate a comprehensive dataset encompassing acupoints on legs and arms across a wide range of cluttered backgrounds and poses. Ultimately, the successful integration of computer vision and artificial intelligence into acupuncture practice holds immense potential for streamlining treatments, enhancing precision, and providing valuable assistive capabilities to practitioners. This work represents an important step towards realizing automated, technology-aided acupuncture, paving the way for further advancements in modernizing this ancient healing modality.

Data availability statement

The datasets presented in this article are not readily available because current agreements do not allow us to make this data publicly available. Requests to access the datasets should be directed to myunggi@pknu.ac.kr.

Ethics statement

The studies involving humans were approved by Institutional Review Board (or Ethics Committee) of Pukyong National University (1041386-202207-HR-41-02) and Dongshin University (202209-SB-040). The studies were conducted in accordance with the local legislation and institutional requirements. The participants provided their written informed consent to participate in this study. Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

Author contributions

HM: Conceptualization, Data curation, Methodology, Software, Visualization, Writing – original draft, Writing – review & editing. S-DS: Data curation, Writing – review & editing. JC: Data curation, Writing – review & editing. C-SN: Funding acquisition, Investigation, Writing – review & editing. B-iL: Conceptualization, Funding acquisition, Project administration, Supervision, Writing – review & editing. MY: Conceptualization, Methodology, Project administration, Supervision, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by the National Research Foundation of Korea (NRF), funded by the Ministry of Science and ICT (No. 2022M3A9B6082791).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product

that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2024.1484038/full#supplementary-material

SUPPLEMENTARY VIDEO S1
Real-time facial acupoint detection using MediaPipe.

SUPPLEMENTARY VIDEO S2
Real-time hand acupoint detection using MediaPipe.

SUPPLEMENTARY VIDEO S3
Real-time hand acupoint detection using YOLOv8 pose estimation model.

SUPPLEMENTARY VIDEO S4
Real-time customized hand acupoint detection using Mediapipe through the GUI.

References

Chan, T. W., Zhang, C., Ip, W. H., and Choy, A. W.. (2021). A combined deep learning and anatomical inch measurement approach to robotic acupuncture points positioning. 2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC). Mexico: IEEE. 2597–2600

Chen, Y. Z., Maigre, C., Hu, M. C., and Lan, K. C.. (2017). Localization of acupoints using augmented reality. Proceedings of the 8th ACM on Multimedia Systems Conference. New York, NY, USA: Association for Computing Machinery. 239–241

Chen, X., Yang, H., Ji, Y., and Chen, D.. (2021). 3D real-time face acupoints recognition system based on HoloLens 2. 2021 7th International Conference on Computer and Communications (ICCC). Chengdu, China: IEEE. 932–938

Focks, C. (Ed.) (2008). Atlas of acupuncture. Edinburgh: Churchill Livingstone-Elsevier.

Formenti, P., Piuri, G., Bisatti, R., Pinciroli, R., and Umbrello, M. (2023). Role of acupuncture in critically ill patients: a systematic review. *J. Tradit. Complement. Med.* 13, 62–71. doi: 10.1016/j.jtcme.2022.10.005

Hou, P. W., Fu, P. K., Hsu, H. C., and Hsieh, C. L. (2015). Traditional Chinese medicine in patients with osteoarthritis of the knee. *J. Tradit. Complement. Med.* 5, 182–196. doi: 10.1016/j.jtcme.2015.06.002

Jocher, G., Chaurasia, A., and Qiu, J.. (2023). Ultralytics YOLO. Available at: https://github.com/ultralytics/ultralytics. (Accessed July 23, 2024)

Li, Y., Teng, Y., Huang, Y., Huang, L., Yang, S., Liu, J., et al. (2024). AIR-Net: acupoint image registration network for automatic acupoint recognition and localization. *Displays* 83:102743. doi: 10.1016/j.displa.2024.102743

Liu, Y. B., Qin, J. H., and Zeng, G. F. (2023). Back acupoint location method based on prior information and deep learning. *Int. J. Numer. Methods Biomed. Eng.* 39:e3776. doi: 10.1002/cnm.3776

Lu, D. P., and Lu, G. P. (2008). Comparing the clinical effect of five varying locations of LI.4 acupoint. *Acupunct. Electrother. Res.* 33, 135–143. doi: 10.3727/036012908803861104

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., et al. (2019). MediaPipe: a framework for building perception pipelines. *arXiv*. Available at: http://arxiv.org/abs/1906.08172. [Epub ahead of preprint]

Mao, J. J., and Khanna, M. M. (2012). "Integrating acupuncture with Western medicine in cancer treatment" in Acupuncture and moxibustion as an evidence-based therapy for cancer. ed. W. Cho (Dordrecht: Springer), 313–328.

Mayer, D. J. (2000, 2000). Acupuncture: an evidence-based review of the clinical literature. *Annu. Rev. Med.* 51, 49–63. doi: 10.1146/annurev.med.51.1.49

MMYOLO Contributors. (2023). MMYOLO: openMMLab YOLO series toolbox and benchmark. Available at: https://github.com/open-mmlab/mmyolo

National University of Korean Medicine, Graduate School of Korean Medicine, Meridian and Acupoint Studies Textbook Compilation Committee. (2020). Acupuncture points. Available at: https://product.kyobobook.co.kr/detail/S000001249781

Park, M. S., Kim, C., Choi, I. W., Chae, I. C., Hur, W., Park, S. S., et al. (2023). MARS-PD: Meridian activation remedy system for Parkinson's disease. *J. Int. Korean Med.* 44, 1–11. doi: 10.22246/jikm.2023.44.1.1

Stefanics, D., and Fox, M. (2022). COCO annotator: web-based image segmentation tool for object detection, localization, and Keypoints. *ACM SIGMultimedia Rec.* 13:7. doi: 10.1145/3578495.3578502

Sulong, G. B., and Randles, M. (2023). Computer vision using pose estimation. *Wasit J. Comput. Math. Sci.* 2, 54–58. doi: 10.31185/wjcm.111

Sun, L., Sun, S., Fu, Y., and Zhao, X.. (2020). Acupoint detection based on deep convolutional neural network. 2020 39th Chinese Control Conference (CCC). 7418–7422.

Sun, S., Xu, H., Sun, L., Fu, Y., Zhang, Y., and Zhao, X.. (2022). Hand acupoint detection from images based on improved HRNet. 2022 International Joint Conference on Neural Networks (IJCNN). Padua, Italy: IEEE. 1–8

Tegiacchi, T. (2021). Manual palpation could be a useful tool to guide traditional acupuncture point selection. *Acupunct. Med.* 39:560. doi: 10.1177/0964528420987568

Terven, J., Córdova-Esparza, D. M., and Romero-González, J. A. (2023). A comprehensive review of YOLO architectures in computer vision: from YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* 5, 1680–1716. doi: 10.3390/make5040083

Vanderploeg, K., and Yi, X. (2009). Acupuncture in modern society. *J. Acupunct. Meridian Stud.* 2, 26–33. doi: 10.1016/S2005-2901(09)60012-1

Wang, Y., Shi, X., Efferth, T., and Shang, D. (2022). Artificial intelligence-directed acupuncture: a review. *Chin. Med.* 17:80. doi: 10.1186/s13020-022-00636-1

World Health Organization (2008). WHO standard acupuncture point locations in the Western Pacific region. Manila: World Health Organization.

Yang, E. S., Li, P. W., Nilius, B., and Li, G. (2011). Ancient Chinese medicine and mechanistic evidence of acupuncture physiology. *Pflugers Arch.* 462, 645–653. doi: 10.1007/s00424-011-1017-3

Yang, S., Ryu, C., Kim, S., and Kim, J. (2021). A development of an acupoints education table using 3D technology and augmented reality. *Korean J Acupunct.* 38, 267–274. doi: 10.14406/acu.2021.036

Yuan, Z., Shao, P., Li, J., Wang, Y., Zhu, Z., Qiu, W., et al. (2024). YOLOv8-ACU: improved YOLOv8-pose for facial acupoint detection. *Front. Neurorobot.* 18:1355857. doi: 10.3389/fnbot.2024.1355857

Zhang, M., Schulze, J., and Zhang, D.. (2022). FaceAtlasAR: atlas of facial acupuncture points in augmented reality. *arXiv*. Available at: https://doi.org/10.48550/arXiv.2111.14755. [Epub ahead of preprint]

Zhang, B., Shi, H., Cao, S., Xie, L., Ren, P., Wang, J., et al. (2022). Revealing the magic of acupuncture based on biological mechanisms: a literature review. *Biosci. Trends* 16, 73–90. doi: 10.5582/bst.2022.01039

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 26 November 2024
DOI 10.3389/fnbot.2024.1439195

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Zhigang Liu,
University of Chinese Academy of Sciences, China
Xiaoyu Shi,
Chinese Academy of Sciences (CAS), China
Guancheng Wang,
Guangdong Ocean University, China

*CORRESPONDENCE
Zhang Juan
✉ zhangjuan5688@126.com

RECEIVED 27 May 2024
ACCEPTED 14 October 2024
PUBLISHED 26 November 2024

CITATION
Juan Z, Zhang J and Gao M (2024) A multimodal travel route recommendation system leveraging visual Transformers and self-attention mechanisms.
Front. Neurorobot. 18:1439195.
doi: 10.3389/fnbot.2024.1439195

A multimodal travel route recommendation system leveraging visual Transformers and self-attention mechanisms

Zhang Juan[1]*, Jing Zhang[2] and Ming Gao[2]

[1]Fanli Business School, Nanyang Institute of Technology, Nanyang, Henan, China, [2]Hospitality Management Department, Tourism College of Zhejiang, Hangzhou, Zhejiang, China

Introduction: With the rapid development of the tourism industry, the demand for accurate and personalized travel route recommendations has significantly increased. However, traditional methods often fail to effectively integrate visual and sequential information, leading to recommendations that are both less accurate and less personalized.

Methods: This paper introduces SelfAM-Vtrans, a novel algorithm that leverages multimodal data—combining visual Transformers, LSTMs, and self-attention mechanisms—to enhance the accuracy and personalization of travel route recommendations. SelfAM-Vtrans integrates visual and sequential information by employing a visual Transformer to extract features from travel images, thereby capturing spatial relationships within them. Concurrently, a Long Short-Term Memory (LSTM) network encodes sequential data to capture the temporal dependencies within travel sequences. To effectively merge these two modalities, a self-attention mechanism fuses the visual features and sequential encodings, thoroughly accounting for their interdependencies. Based on this fused representation, a classification or regression model is trained using real travel datasets to recommend optimal travel routes.

Results and discussion: The algorithm was rigorously evaluated through experiments conducted on real-world travel datasets, and its performance was benchmarked against other route recommendation methods. The results demonstrate that SelfAM-Vtrans significantly outperforms traditional approaches in terms of both recommendation accuracy and personalization. By comprehensively incorporating both visual and sequential data, this method offers travelers more tailored and precise route suggestions, thereby enriching the overall travel experience.

KEYWORDS

multimodal travel recommendation, visual Transformer, self-attention mechanism, image and sequence fusion, deep learning

1 Introduction

In recent years, with the improvement of living standards and the increasing demand for travel, efficiently recommending personalized travel itineraries has become an urgent problem to be addressed. Traditional travel route recommendation methods often rely on expert experience, which struggles to meet personalized needs and fails to handle large-scale and complex data effectively (Renjith et al., 2020). With the introduction of machine learning techniques, it has become possible to process big data more efficiently and provide accurate personalized recommendations based on users' historical behavior and preferences (Ji et al., 2020). Through the iterative optimization process of machine learning

algorithms, the accuracy and efficiency of recommendation systems have been significantly improved, leading to a more satisfying user experience (Egli et al., 2020). Therefore, research on travel itinerary recommendations not only holds theoretical significance but also promotes its practical application.

Traditional travel recommendation methods primarily rely on symbolic AI and knowledge representation, usually implemented through expert systems that simulate human experts' decision-making processes. These systems can encode expert knowledge and provide clear explanations for each recommendation, such as the multi-agent knowledge system proposed by Lorenzi (2007). Another category of methods relies on predefined rules, exhibiting high determinism and reliability, which perform well in complex or dynamic travel scenarios. Gandhi et al. (2014) introduced a rule-based system for automated travel analysis, while Jiang and Dai (2024) presented a rule-based system framework for analyzing travel performance. Although these methods offer strong interpretability and transparency, they fall short in handling large-scale data and complex travel demands. Simulation computing techniques can predict and analyze travel behavior by constructing and running simulation models, but they are still insufficient for addressing complex, dynamic needs, and processing large-scale data (Gong et al., 2023; Khan et al., 2023).

To overcome the limitations of traditional algorithms in terms of adaptability and handling complex requirements, data-driven and machine learning-based algorithms have optimized recommendations by analyzing large volumes of user data and historical behavior, offering high accuracy and personalized recommendations. Decision tree-based methods have been widely applied for user classification and tourism recommendations. Kesorn et al. (2017) used the C4.5 decision tree algorithm to recommend travel regions for tourists, while Kbaier et al. (2017) proposed a personalized hybrid travel recommendation system that uses decision tree algorithms to recommend attractions based on user preferences. Random forest algorithms improve the stability and accuracy of recommendations by combining predictions from multiple decision trees (Li, 2024), and Support Vector Machines (SVMs) excel in handling high-dimensional data and nonlinear classification problems (Lahagun et al., 2024; Yuan, 2022). However, these methods face significant challenges regarding computational complexity, particularly when dealing with dynamic and large-scale data.

The application of deep learning algorithms addresses the limitations of statistical and machine learning algorithms in terms of adaptability and handling complex requirements. Convolutional Neural Networks (CNNs) capture user interests and generate personalized recommendations, significantly improving recommendation accuracy and user satisfaction (Wang, 2020). Reinforcement Learning (RL) dynamically adjusts recommendation strategies to optimize user experience (Kong et al., 2022). Transformer models, due to their powerful sequence modeling capabilities, have demonstrated excellent performance in travel recommendations (Yang et al., 2022). However, these methods still face challenges related to high computational complexity and the demand for processing large-scale data.

Although previous travel route recommendation systems have made some progress in personalization, they often rely on single data modalities (such as user behavior data, geographical data, etc.) or simple recommendation algorithms, failing to fully leverage users' multimodal information (such as visual and sequential information). Specifically, traditional methods have limitations in several areas. First, many travel recommendation systems overlook the potential of visual information, with most relying on text data or user behavior data. However, travelers' decisions are often heavily influenced by photos or videos of attractions. Therefore, integrating visual information effectively into recommendation systems has been a critical unsolved problem. Second, existing systems often fall short in handling temporal information. Travel decisions usually exhibit time dependence, with users often planning future trips based on previously visited locations or activities. However, many recommendation algorithms fail to capture the temporal patterns in user behavior effectively, leading to recommendations that lack sufficient personalization. Additionally, the integration of multimodal data has been a significant challenge in the field of recommendation systems. How to effectively fuse visual and temporal information and fully explore the connections between them remains a major issue. Thus, the primary motivation of this study is to fill the gap in combining visual and temporal information effectively in travel route recommendations and provide a more personalized recommendation system. Our proposed SelfAM-Vtrans model extracts spatial relationships from images through the visual Transformer, processes temporal information using the LSTM network, and integrates these two modalities using the self-attention mechanism. This comprehensive approach captures user preferences and provides more personalized and accurate travel route recommendations. Addressing these issues not only improves the performance of recommendation systems but also significantly enhances user experience, helping users receive more precise suggestions for complex travel decisions. Therefore, this research is of great theoretical significance and also holds broad potential for practical applications.

Contributions of this paper:

- We propose a travel route recommendation algorithm that comprehensively considers visual and sequential information. By combining Vision Transformer, LSTM, and self-attention mechanisms, we can fully utilize image and sequence information, improving the accuracy and personalization of route recommendations.
- We conducted experiments on real travel datasets and compared them with other travel route recommendation methods. The results show that our algorithm significantly outperforms traditional methods in terms of recommendation accuracy and personalization.
- Our research provides a novel method combining deep learning and machine learning technologies, offering more accurate and personalized route recommendations for travelers. This is of great significance for improving travelers' experiences and meeting their personalized needs.

2 Related work

2.1 Travel route recommendation

With the development of deep learning and machine learning, the application of multimodal data fusion in travel route recommendation is receiving increasing attention. Multimodal data, including images, text, and audio, can provide a more comprehensive understanding of users' needs and preferences by integrating these different types of information, thereby offering more accurate and personalized route recommendations (Jin et al., 2017). In multimodal fusion methods, a common strategy is to use deep neural networks to encode and represent data from different modalities. For instance, Convolutional Neural Networks (CNNs) can be employed to extract features from images, while Recurrent Neural Networks (RNNs) or self-attention mechanisms are used to process text sequences, and audio recognition technologies handle audio data. Then, by fusing the representations from different modalities, an integrated multimodal representation can be formed for route recommendation (Lin et al., 2024b). Another important aspect is the alignment and fusion of multimodal data. Since data from different modalities often have distinct characteristics and representational forms, effectively aligning and fusing them is a key challenge. A common approach is to use attention mechanisms to learn the associative weights between modalities, allowing for a weighted integration of information from different sources. Additionally, joint training can be employed, where multiple modal representation networks are trained simultaneously to maintain consistency in the representational space (Jin et al., 2018). Furthermore, multimodal fusion methods can also incorporate collaborative filtering and reinforcement learning techniques from recommendation systems to further enhance route recommendation effectiveness. For example, collaborative filtering methods can be used to learn user preferences from historical data, which can then be combined with multimodal data fusion to generate personalized route recommendations.

2.2 Reinforcement learning

Traditional travel route recommendation methods often rely on users' historical data and preferences, overlooking the interactions and feedback during the recommendation process. However, reinforcement learning-based travel route recommendation methods can dynamically learn and optimize recommendation strategies through interactions with users, providing more personalized and adaptive route recommendations (Jin et al., 2015). In reinforcement learning-based methods, the route recommendation problem can be modeled as a Markov Decision Process (MDP). The traveler, acting as an agent, interacts with the environment, chooses actions based on the current state, receives rewards, and updates strategies. Through continuous interaction and learning with users, the system can gradually optimize the route recommendation strategy, offering recommendations that better meet user needs (Lin et al., 2024a). In practice, deep reinforcement learning methods can be utilized to address the travel route recommendation problem. For instance, Deep Q-Networks (DQNs) can be used to learn the action-value functions of travelers, choosing the optimal actions based on the current state. Additionally, policy gradient methods can be used to train a policy network that directly outputs the probability distribution of route recommendations. The advantage of reinforcement learning methods in travel route recommendation is their flexibility in adapting to different user preferences and environmental changes. Through interaction and feedback from users, the system can proactively learn users' likes and preferences, thereby providing more personalized and satisfying route recommendations (Zhang et al., 2024). However, reinforcement learning-based methods also face challenges. Firstly, establishing accurate state representations and reward functions is crucial and requires careful consideration of user needs and environmental characteristics. Secondly, reinforcement learning methods typically require extensive interaction and training time, which may pose limitations for real-time travel recommendation systems.

2.3 Neural networks

Neural Networks, as a computational model that mimics the workings of the human nervous system, have made significant advancements in the field of artificial intelligence in recent years (Abbasi-Moud et al., 2021). They are composed of a large number of simple processing units called neurons, which use learning algorithms to handle complex pattern recognition and decision-making tasks. Neural networks were originally proposed by biologist McCulloch and mathematician Pitts in 1943 and further developed into the perceptron model by Rosenblatt in the early 1950's. However, due to limitations in computational power and data availability at the time, the development of neural networks stagnated. It wasn't until the late 1980's and early 1990's that multilayer neural networks (multilayer perceptrons) regained attention with the introduction of the backpropagation algorithm and advancements in computer technology. They achieved some progress in fields such as speech recognition and image recognition (Wong et al., 2020). In 2006, Hinton and colleagues introduced Deep Belief Networks, marking the rise of deep learning. Deep learning, through multiple layers of nonlinear transformations, can effectively learn and represent complex patterns in data (Lin C. et al., 2024). Since then, deep learning has made significant breakthroughs in computer vision, natural language processing, recommendation systems, and other fields, becoming one of the mainstream technologies in modern artificial intelligence (Wang et al., 2023). In recent years, with advancements in hardware computational power and the widespread availability of big data, neural network architectures have been continuously evolving and optimizing. From the early days of Convolutional Neural Networks (CNNs) to subsequent models like Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Transformers, each architecture provides efficient solutions for specific tasks and data types. In the future, neural networks are expected to continue playing important roles in fields such as medical diagnostics, intelligent transportation, and smart manufacturing. As researchers explore new network structures

and optimization methods, the application prospects of neural networks will become even broader, further driving the continuous development and innovation of artificial intelligence technologies (Xiao et al., 2023).

3 Methodology

3.1 Overview of our network

In this study, we propose a novel model architecture called "SelfAM-Vtrans Net," which combines self-attention mechanism with Vision Transformer (ViT) for recommending travel itineraries by utilizing multimodal information from images and text data. Specifically, the SelfAM-Vtrans Net incorporates Vision Transformer (ViT) for processing image data and enhances the capability of LSTM in handling textual data through self-attention mechanism (SelfAM). The model architecture involves ViT for extracting high-level semantic features from tourism destination images by dividing the images into fixed-sized patches and feeding them into the Transformer network. LSTM, along with the self-attention mechanism, processes travel itinerary descriptions and user reviews, capturing the temporal information of the text and improving focus on important textual information. The multimodal feature fusion combines the image features extracted by ViT with the text features processed by LSTM through self-attention mechanism, generating a comprehensive feature representation for the recommendation task. As shown in Figure 1, the proposed model integrates multiple components to enhance recommendation accuracy.

Multimodal fusion methods leverage deep learning technologies to integrate different types of data, such as images, text, and audio, to obtain a more comprehensive understanding of user needs and preferences. The principle includes: first, using appropriate neural network models (such as CNN, RNN, etc.) to encode and represent data from different modalities; second, using attention mechanisms or joint training to fuse representations from different modalities into a comprehensive multimodal representation; finally, applying the multimodal representation to the travel route recommendation task to generate personalized recommendation results. Reinforcement learning-based methods dynamically learn and optimize route recommendation strategies through interaction and feedback with users. The principle includes: first, modeling the travel route recommendation problem as a Markov Decision Process (MDP), where the traveler interacts with the environment as an agent; second, using deep reinforcement learning methods (such as DQN, policy gradient, etc.) to learn the traveler's action-value functions or policy network, choosing the optimal action based on the current state; finally, continuously updating strategies through interaction with users to optimize route recommendation results. Social network-based methods use user relationships and user-generated content within social networks to provide personalized and trustworthy route recommendations. The principle includes: first, analyzing relationships, interests, and travel experiences among users to construct user social feature representations; second, utilizing travel experiences, photos, comments, and other content shared by users on social networks to obtain user characteristics and travel-related information; finally, using social recommendation and social influence propagation mechanisms to recommend travel routes related to user interests and disseminate recommendations through users' social relationships.

Data collection and preprocessing: collect multimodal data, user social relationship data, and user-generated content data, and perform data cleaning and preprocessing. Implementation of Multimodal Fusion Methods: a. Use appropriate neural network models to encode and represent data from different modalities. b. Use attention mechanisms or joint training to fuse representations from different modalities into a comprehensive multimodal representation. c. Apply the multimodal representation to the travel route recommendation task to generate personalized recommendation results. Implementation of Reinforcement Learning-Based Methods: a. Model the travel route recommendation problem as a Markov Decision Process (MDP). b. Use deep reinforcement learning methods to learn the traveler's action-value functions or policy network, choosing the optimal action based on the current state. c. Continuously update strategies through interaction with users to optimize route recommendation results. Implementation of Social Network-Based Methods: a. Analyze relationships, interests, and travel experiences among users to construct user social feature representations. b. Utilize content shared by users on social networks to obtain user characteristics and travel-related information. c. Use social recommendation and social influence propagation mechanisms to recommend travel routes related to user interests and disseminate recommendations through users' social relationships. Integrate multimodal fusion, reinforcement learning-based, and social network-based methods, considering multiple factors to generate the final personalized travel route recommendation results. Evaluate and optimize recommendation results, continuously improving the algorithm's performance and accuracy. Provide a user interface or API interface, allowing users to easily input their needs and receive personalized travel route recommendations.

Firstly, while the combination of visual Transformers and LSTMs is theoretically feasible, its specific application in travel route recommendation presents numerous challenges. The key difficulty lies in the effective integration of multimodal information, particularly the heterogeneity between visual and sequential data. Image data and sequential data possess distinct characteristics in both spatial and temporal dimensions. A significant challenge we addressed in this research is how to effectively fuse these through the self-attention mechanism. Secondly, the travel route recommendation problem involves not only route selection but also the improvement of personalization and accuracy. When dealing with large-scale and complex user behavior data, the proposed model needs to capture user preferences while also adapting to dynamically changing environments and user demands. By combining the visual feature extraction capabilities of the visual Transformer with the strength of LSTM in handling sequential data, and using the self-attention mechanism to balance the importance of the two, especially in terms of multimodal data collaboration, careful model design and tuning are required to ensure the system's real-time performance and efficiency. Additionally, we have conducted extensive experiments

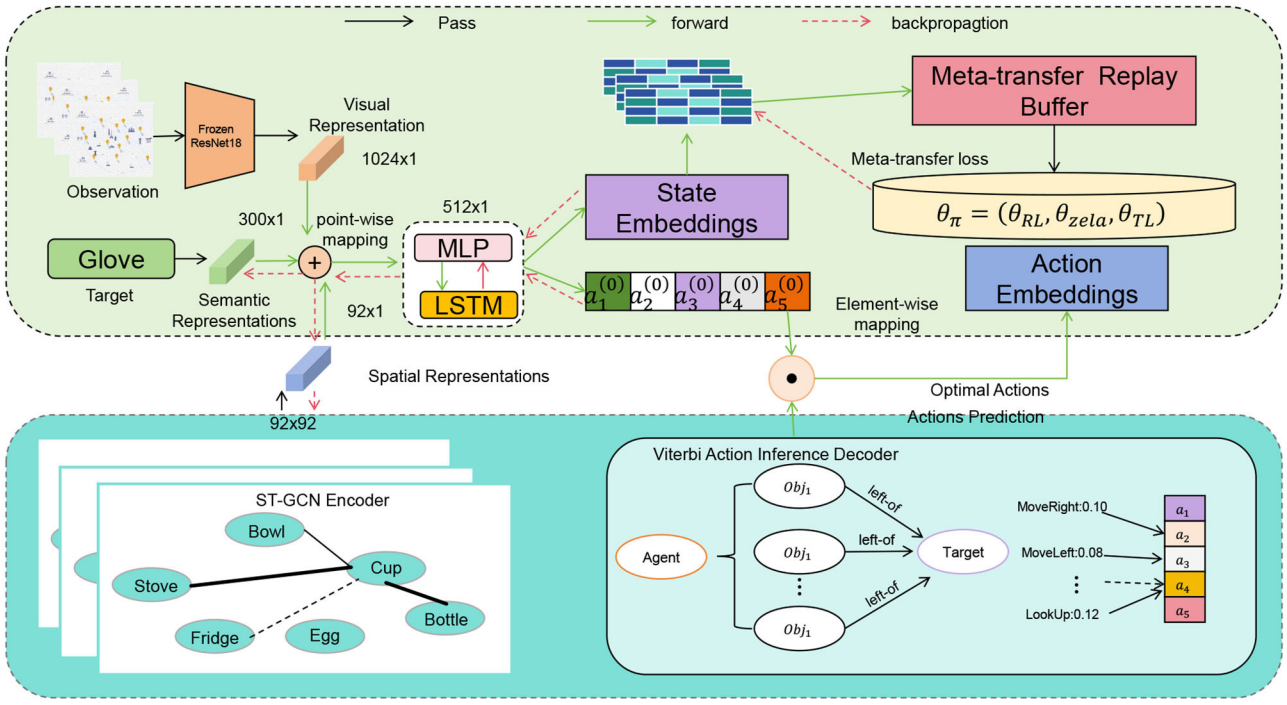


FIGURE 1
The overall framework of the proposed SelfAM-Vtrans model, illustrating the integration of visual Transformer, LSTM, and self-attention mechanisms for travel route recommendations.

demonstrating that this model outperforms traditional methods across different datasets. This further proves the effectiveness of our approach and its potential for real-world applications. In the revised manuscript, we will provide a more detailed explanation of these challenges and how we addressed them, thereby better showcasing the novelty of our research.

3.2 Vision-Transformer model

The Vision Transformer (ViT) is a deep learning model based on the Transformer architecture, designed to process and analyze visual data such as images (Abdelraouf et al., 2022). While traditional Convolutional Neural Networks (CNNs) have achieved tremendous success in computer vision tasks, ViT introduces a novel approach by incorporating the self-attention mechanism into the visual domain, allowing the model to process images without convolutional layers (Pramanick et al., 2022).

The core idea of the ViT model is to segment an image into a set of fixed-size patches (Figure 2), which are then transformed into a sequence. Each patch is mapped to a lower-dimensional vector representation, known as an embedding vector, through a linear transformation (typically a fully connected layer). These embedding vectors are fed into the Transformer encoder in a sequential format. The Transformer encoder consists of multiple self-attention layers and feed-forward neural network layers. Self-attention layers use the attention mechanism to model the relationships between different positions in the sequence to capture global contextual information. In ViT, the self-attention mechanism is used to capture dependencies between patches, achieving a global understanding of the image. Through iterative processing by the self-attention layers, the model gradually integrates information between patches and generates feature representations with global awareness. In the ViT model, positional encodings are introduced to imbue the model with positional information. Positional encoding is a technique to embed positional information of each patch into the feature representation, usually generated using sine and cosine functions, so that the model can perceive the relative distances and order in the sequence.

In this method, the ViT model plays a part in the multimodal fusion approach, responsible for processing image data and generating corresponding embedding vectors. Its role can be broadly divided into two aspects: Image Feature Extraction: The ViT model possesses powerful image feature extraction capabilities. By segmenting the input image into patches and converting them into a sequence of embedding vectors, ViT can globally understand and encode the image. These embedding vectors capture the semantic and contextual information of the image, effectively representing its features. Multimodal Fusion: In the multimodal fusion method, the image embedding vectors generated by the ViT model are fused with representations from other modalities (such as text, audio, etc.) to obtain a comprehensive multimodal representation. This fusion can be achieved through attention mechanisms or joint training, integrating and influencing information across different modalities. By combining image information with other modal information,

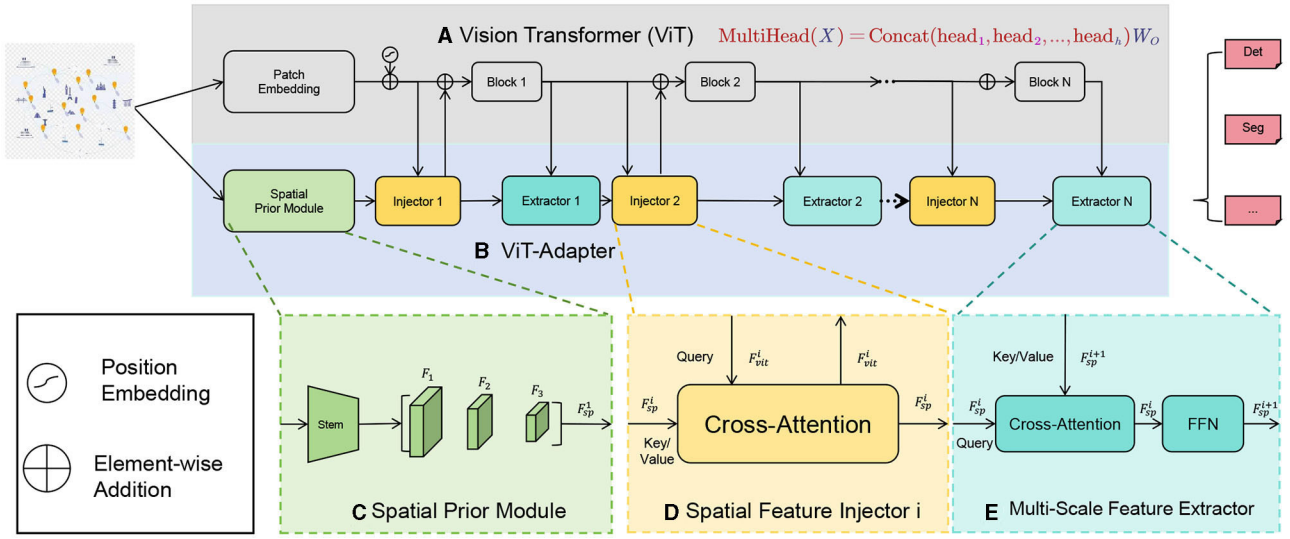


FIGURE 2
(A) is the baseline model of ViT. **(C)** Represents the Spatial Prior Module in **(B)**. **(D)** Represents the Spatial Feature Injector Module in **(B)**. **(E)** Represents the Extractor Module in **(B)**. And the colors are also one-to-one corresponding.

the comprehensive multimodal representation more fully reflects user needs and preferences, providing more accurate travel route recommendations.

The formula of the Vision Transformer model is as follows:

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h)W_O \quad (1)$$

Among them, the explanation of variables is as follows: X: input sequence, corresponding to patch embedding vectors in the image. head_i: The output of the ith attention head. h: The number of attention heads. Concat(·): Concatenate the output of all attention heads. W_O: The weight matrix of the output matrix.

The calculation process of each attention head can be expressed as:

$$\begin{aligned}\text{head } i &= \text{Attention}\,(XWQi, XW_{Ki}, XW_{Vi})\ \text{Attention}(Q, K, V) \\ &= \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \end{aligned} \quad (2)$$

Among them, the explanation of variables is as follows: W_{Qi}, W_{Ki}, W_{Vi}: are the linear transformation weight matrices of query (Query), key (Key), and value (Value), respectively. d: The dimension of the embedding vector. In the above formula, each attention head obtains the query, key and value by linearly transforming the input sequence X, and uses them as the input of the self-attention mechanism. Through the self-attention mechanism, the model can model the dependencies between patches in the input sequence and obtain global contextual information. Finally, the outputs of all attention heads are connected and linearly transformed to obtain the final multi-head attention representation.

3.3 LSTM model

The LSTM unit consists of three key parts: input gate, forget gate and output gate. Each gate control unit consists of a sigmoid activation function and a dot product operation to control the flow of information (Sun et al., 2020). Figure 3 is a schematic diagram of the principle of Vision-Transformer model.

First, for each time step t, LSTM receives the input x_t and the hidden state h_{t-1} of the previous moment as input. Then, calculate the activation value i_t of the input gate, the activation value f_t of the forgetting gate, and the activation value o_t of the output gate. You can use the following formula:

$$\begin{aligned} i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i)\, f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \end{aligned} \quad (3)$$

Among them, W and b are learnable weight and bias parameters, and σ represents the sigmoid activation function.

Next, calculate the candidate memory cell state $\tilde{C}_t$ and the memory cell state C_t at the current moment:

$$\tilde{C}t = \tanh(Wcxx_t + W_{ch}h_{t-1} + b_c)\, C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (4)$$

Among them, $\odot$ represents element-wise multiplication, and tanh represents the hyperbolic tangent activation function.

Finally, calculate the hidden state h_t at the current moment based on the output gate o_t and the memory cell state C_t:

$$h_t = o_t \odot \tanh(C_t) \quad (5)$$

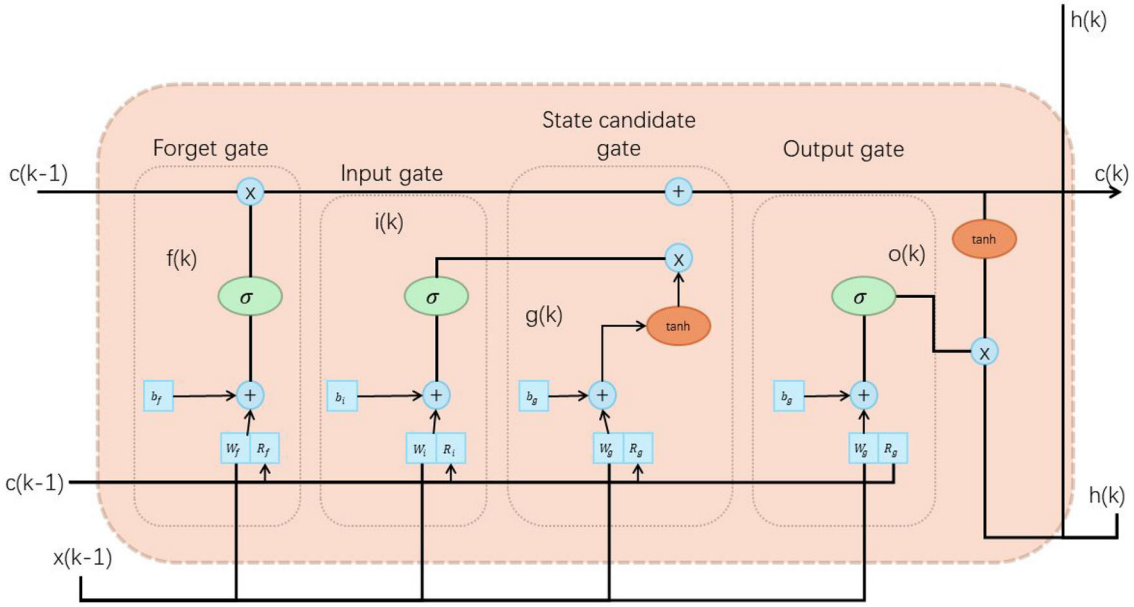


FIGURE 3
A schematic diagram of the principle of Vision-Transformer model.

The hidden state h_t in the LSTM model can be passed to the next time step and used for prediction or further processing.

The role of LSTM in time series data processing is as follows:

Long-term dependency modeling: LSTM can selectively retain or forget past information through the mechanism of forget gate and input gate, thereby effectively handling long-term dependencies. This enables LSTM to better capture dependencies with long time intervals when processing time series data, such as sentence structure and semantic relationships in natural language processing. Gradient stability: Due to the gating mechanism of LSTM, it can alleviate the gradient disappearance and gradient explosion problems, allowing the model to learn and update parameters more stably during the training process. This makes LSTM perform well in tasks that deal with long sequences and complex time dependencies. Multi-step prediction: LSTM can achieve multi-step prediction by passing the hidden state of the current time step to the next time step. This enables LSTM to generate continuous output sequences in tasks such as sequence generation, machine translation, etc.

3.4 Self-attention mechanism

The self-attention mechanism is an attention mechanism used to process sequence data and was originally introduced in the Transformer model (Shi et al., 2022). It is able to establish associations between different positions and adaptively learn dependencies within the input sequence (Chen et al., 2022). The basic principle of the self-attention mechanism is to calculate the correlation weight of each input position with other positions, and then perform a weighted sum of the inputs based on these weights.

Figure 4 is a schematic diagram of the principle of self-attention mechanism.

The following is a detailed introduction to self-attention: Input representation: suppose there is an input sequence $X = x_1, x_2, ..., x_n$, where x_i represents the ith element in the input sequence. In Self-attention, the input sequence is usually represented as a matrix $X \in \mathbb{R}^{n\times d}$, where n is the sequence length and d is the dimension of each element. Query, key and value: in order to calculate the relevance weight of each position, Self-attention introduces three linear transformations, which are used to calculate the query, key, and value, respectively. These transformations map the input sequence X to different representation spaces by learning trainable weight matrices. Specifically, for the input sequence X, the query matrix $Q \in \mathbb{R}^{n\times d_k}$, the key matrix $K \in \mathbb{R}^{n\times d_k}$, and the value matrix $V \in \mathbb{R}^{n\times d_v}$ are obtained through the following linear transformation:

$$Q = XW_Q\ K = XW_K\ V = XW_V \tag{6}$$

Among them, $W_Q \in \mathbb{R}^{d\times d_k}$, $W_K \in \mathbb{R}^{d\times d_k}$ and $W_V \in \mathbb{R}^{d\times d_v}$ is a learnable weight matrix, d_k and d_v represent the dimensions of the query and key values, respectively.

Relevance weight calculation: obtain the correlation weight by calculating the similarity between the query matrix Q and the key matrix K. A common calculation method is to use dot-product attention, which calculates the similarity score between the query and the key through the inner product. To scale the attention score and improve stability, it can be divided by $\sqrt{d_k}$. Specifically, the correlation weight matrix $A \in \mathbb{R}^{n\times n}$ can be calculated as follows:

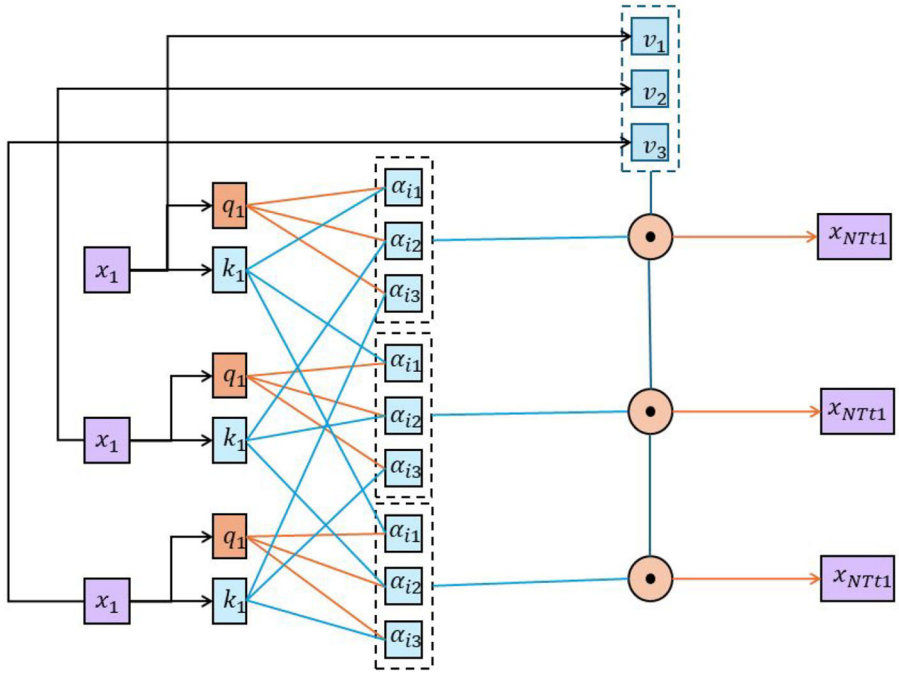


FIGURE 4
A schematic diagram of the principle of self-attention mechanism.

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{7}$$

Among them, the softmax function is used to convert the similarity score into a probability distribution to ensure that the weight sum of each position is 1. Weighted sum: By multiplying the correlation weight matrix A with the value matrix V, a weighted sum representation of each position can be obtained. Specifically, the output matrix $Y \in \mathbb{R}^{n \times d_v}$ of Self-attention can be calculated as follows:

$$Y = AV \tag{8}$$

The output matrix Y contains a weighted representation of each position relative to other positions, where the weight of each position is determined by the correlation weight.

The role of the Self-attention mechanism in the model is as follows:

Establish long-distance dependencies: Traditional recurrent neural networks (RNN) face the problems of gradient disappearance and gradient explosion when processing long sequences, making it difficult to capture long-distance dependencies. The Self-attention mechanism can directly establish the association between any two locations, no matter how far apart they are, thereby effectively capturing long-distance dependencies. This enables the model to better capture contextual information when processing long sequences. Parallel computing: Since the Self-attention mechanism can directly calculate the correlation weight between any two positions, the calculation process can be highly parallelized. This means the model can process large-scale sequence data more efficiently, speeding up training and inference. Context awareness: The Self-attention mechanism can adaptively learn weights based on different parts of the input sequence, allowing the model to better focus on contextual information related to the current position. By calculating the correlation weight, the model can dynamically adjust the importance of each position based on the input semantic information, thereby better capturing the semantic characteristics of the sequence. Feature interaction: The Self-attention mechanism can promote feature interaction and information transfer between different locations. By calculating correlation weights and performing a weighted sum of values, the model can interact information from each location with other locations to integrate global context and extract a richer feature representation. In short, the self-attention mechanism can establish global associations in the model, capture long-distance dependencies, parallel computing, context awareness, and Feature interaction. This enables the model to better handle sequence data and achieve significant performance improvements in tasks such as natural language processing and machine translation. The training process of the proposed SelfAM-Vtrans model is summarized in Algorithm 1.

4 Experiment

4.1 Datasets

In our experiments, we utilized several multimodal datasets that include both visual and sequential information to evaluate the

```
Input : Training dataset, Validation dataset, Test
        dataset
Output: Trained SelfAM-Vtrans Net model
Initialize the SelfAM-Vtrans Net model;
Initialize hyperparameters, optimizer, and loss
function;
while not converged do
    for each mini-batch of training data do
        Perform data augmentation on the mini-batch;
        Encode the input data using V-Net;
        Apply attention mechanisms to capture
        relevant information;
        Apply cross-attention to incorporate
        information from different sources;
        Apply Transformer layers for feature
        extraction;
        Pass the processed data through the
        SelfAM-Vtrans Net model;
        Calculate the loss with the predicted
        outputs and ground truth labels;
        Update the model parameters using
        backpropagation;
    end
    Calculate evaluation metrics on the validation
    dataset (e.g., Recall, Precision);
    if performance on validation dataset improves then
        Save the current best model;
    end
    if no improvement in validation performance then
        Stop training;
    end
end
Load the best saved model;
Evaluate the model on the test dataset;
Calculate evaluation metrics on the test dataset
(e.g., Recall, Precision);
```

Algorithm 1. Training of SelfAM-Vtrans Net.

effectiveness of the proposed SelfAM-Vtrans model. Specifically, we employed four datasets: TripAdvisor Dataset (Nilizadeh et al., 2019), Expedia Dataset (Goldenberg and Levin, 2021), Yelp Dataset (Asghar, 2016), and Open Images Dataset (Kuznetsova et al., 2020). The TripAdvisor and Expedia datasets provide user-generated reviews and travel-related data such as geographic locations, reviews, and ratings of hotels, restaurants, and tourist attractions, capturing textual information relevant to user preferences. The Yelp dataset contains similar multimodal information, with reviews, ratings, and user-generated content related to businesses such as restaurants and shops. The Open Images dataset, used to supplement the visual modality, consists of large-scale image data annotated with relevant tags and metadata, enabling the model to extract visual features from travel-related images. These datasets together offer a comprehensive multimodal context, combining both textual and visual information that is crucial for personalized travel recommendations. In terms of size, each dataset varies, with the TripAdvisor and Yelp datasets containing millions of user reviews and the Open Images dataset containing ∼9 million images. This multimodal composition allows our model to fully leverage both image and sequential data, improving recommendation accuracy by capturing spatial relationships within images and temporal patterns within sequences.

4.2 Experimental details

The experiments were conducted on an NVIDIA DGX-1 system equipped with 8 NVIDIA A100 GPUs, each with 40 GB HBM2 memory, using PyTorch 1.8.0 as the main deep learning framework. The evaluation metrics include Accuracy, AUC (Area Under the Curve), Recall, and F1 score, which were used to comprehensively assess the model's predictive performance. We trained the ViT-Base visual Transformer model (∼86 M parameters, 17.6 GFLOPs computational complexity) in combination with an LSTM layer with 256 hidden units, resulting in a total of about 91M parameters. The model training used the Adam optimizer with an initial learning rate of 1×10^{-4}, and we applied a learning rate warm-up strategy. Each training batch had a size of 64, and training was conducted for 200 epochs. We also used L_2 regularization (weight decay of 0.01) to prevent overfitting and applied a cosine annealing learning rate scheduler to gradually reduce the learning rate, improving the model's generalization ability. Regarding data processing, we performed text cleaning, tokenization, and padding for textual data (such as the TripAdvisor and Yelp datasets), and image resizing and normalization for image data (such as the Open Images dataset). The experiments were divided into training, validation, and test sets, and we recorded the model's training time, inference time, total number of parameters, and computational complexity (FLOPs). Early stopping based on performance on the validation set was employed to avoid overfitting and ensure the model's generalization. We also conducted ablation studies to analyze the impact of different modules on the model's performance. Providing a detailed description of these experimental settings and parameters ensures the reproducibility of the results and verifies the model's superiority and stability across different datasets and evaluation metrics. Data Processing: We used four datasets, including TripAdvisor, Expedia, Yelp, and Open Images. For textual data (e.g., TripAdvisor and Yelp datasets), we first performed text cleaning and tokenization, constructed a vocabulary, and padded the text sequences to a fixed length. For image data, we resized and normalized the images. Experimental Procedure: All experiments followed a standard split of 80% for the training set, 10% for the validation set, and 10% for the test set. In each experiment, we recorded the model's training time, inference time, total number of parameters, and computational complexity (FLOPs). Early stopping based on validation set performance was applied to prevent overfitting. Additionally, we conducted ablation studies to analyze the impact of different model components on the final performance.

TABLE 1 Key performance metrics for different methods on various datasets.

References	TripAdvisor dataset				Expedia dataset			
	Accuracy	Recall	F1 score	AUC	Accuracy	Recall	F1 score	AUC
Chen et al. (2021)	87.48	93.14	91.22	86.14	85.73	91.83	90.91	91.97
Park and Liu (2022)	86.08	91.66	88.04	89.55	86.53	85.20	88.25	93.34
Duan et al. (2020)	89.00	84.34	89.74	93.64	89.72	88.80	86.53	86.52
Zhou et al. (2020)	92.20	87.94	84.97	85.72	94.32	88.10	86.49	87.81
Hu et al. (2020)	94.76	90.07	89.28	88.41	96.27	90.69	88.70	85.32
Sharma (2024)	94.00	87.76	86.02	88.98	89.16	87.84	89.23	90.74
Ours	**97.03**	**94.01**	**93.85**	**96.22**	**97.38**	**95.08**	**93.71**	**95.71**
References	Yelp dataset				Open Images dataset			
	Accuracy	Recall	F1 score	AUC	Accuracy	Recall	F1 score	AUC
Chen et al. (2021)	87.92	84.49	86.12	88.45	88.50	90.42	85.89	90.60
Park and Liu (2022)	89.77	90.46	88.06	92.25	87.72	89.18	88.18	89.02
Duan et al. (2020)	88.19	86.35	85.05	85.88	87.40	85.35	89.85	91.71
Zhou et al. (2020)	88.44	89.89	90.12	87.87	95.55	88.96	85.78	92.98
Hu et al. (2020)	92.92	88.52	84.64	88.97	87.41	86.96	87.87	87.33
Sharma (2024)	86.88	87.07	88.64	91.70	89.33	88.09	89.13	89.64
Ours	**96.88**	**95.56**	**93.19**	**95.22**	**97.71**	**95.52**	**92.58**	**96.17**

Bold fonts indicate the best value.

4.3 Experimental results and analysis

Table 1 presents the specific results of various models on the TripAdvisor dataset, Expedia dataset, Yelp dataset, and Open Images dataset. Our method performs exceptionally well on all datasets. For example, on the TripAdvisor dataset, our method achieves accuracy, recall, F1 score, and AUC of 97.03, 94.01, 93.85, and 96.22%, respectively, which are significantly higher than the methods of Chen et al. (2021) (87.48, 93.14, 91.22, 86.14%) and Park and Liu (2022) (86.08, 91.66, 88.04, 89.55%). Similarly, on the Expedia dataset, our method achieves accuracy, recall, F1 score, and AUC of 97.38, 95.08, 93.71, and 95.71%, respectively, far surpassing the methods of Duan et al. (2020) (89, 84.34, 89.74, 93.64%) and Zhou et al. (2020) (92.2, 87.94, 84.97, 85.72%). On the Yelp dataset, our method achieves accuracy, recall, F1 score, and AUC of 96.88, 95.56, 93.19, and 95.22%, respectively, which are significantly better than the methods of Hu et al. (2020) (92.92, 88.52, 84.64, 88.97%) and Sharma (2024) (86.88, 87.07, 88.64, 91.7%). On the Open Images dataset, our method achieves accuracy, recall, F1 score, and AUC of 97.71, 95.52, 92.58, and 96.17%, respectively, once again outperforming other comparative methods.These results indicate that SelfAM-Vtrans Net exhibits superior accuracy and reliability.

Experimental in Table 2 presents the key performance metrics of different methods on the TripAdvisor dataset, Expedia dataset, Yelp dataset, and Open Images dataset. Our method (Ours) outperforms other methods significantly on all datasets, demonstrating fewer parameters and lower computational complexity. For example, on the TripAdvisor dataset, our method has only 177.00 M parameters and 222.90 G FLOPs, much lower than Chen et al. (2021) (388.10 M, 375.78 G), Park and Liu (2022) (389.89 M, 361.01 G), and others. On the other datasets (Expedia, Yelp, and Open Images), our method also maintains a lower parameter count and FLOPs, reflecting advantages in resource utilization efficiency. Additionally, our method exhibits significantly shorter inference and training times. For instance, on the Open Images dataset, our inference time is only 118.94 ms, much less than other methods such as Chen et al. (2021) (240.08 ms), Park and Liu (2022) (223.75 ms), and others, while also demonstrating excellent training time. Our method not only excels in accuracy (97.03%) and performance metrics but also possesses noticeable advantages in key metrics such as parameter count, computational complexity, inference time, and training time. These advantages positively impact efficiency and cost in practical deployment and application, establishing a solid foundation for real-world applications.

The results of our ablation study, as shown in Table 3, demonstrate the superior performance of our approach compared to other methods (CNN, GRU, and BiLSTM) across various datasets. Our method, utilizing the LSTM module, excelled in key metrics such as Accuracy, Recall, F1 score, and AUC.

TABLE 2 Inference and training times for various methods on different datasets.

References	TripAdvisor dataset				Expedia dataset			
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)
Chen et al. (2021)	388.10	375.78	316.33	333.84	392.92	387.56	322.59	231.39
Park and Liu (2022)	389.89	361.01	249.66	308.66	366.67	210.75	399.53	381.73
Duan et al. (2020)	300.70	379.74	372.24	258.53	351.51	261.25	242.28	264.86
Zhou et al. (2020)	349.32	283.54	274.55	305.18	280.76	229.60	214.61	349.32
Hu et al. (2020)	215.34	335.34	361.48	221.11	339.58	237.88	200.41	301.59
Sharma (2024)	326.29	305.51	303.25	371.38	304.10	232.16	334.90	380.34
Ours	**177.00**	**222.90**	**158.58**	**114.97**	**192.73**	**133.35**	**108.83**	**197.33**
References	Yelp dataset				Open Images dataset			
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)
Chen et al. (2021)	319.72	254.78	205.01	370.22	316.82	281.96	240.08	390.39
Park and Liu (2022)	312.27	339.02	223.11	210.46	317.75	361.67	223.75	358.71
Duan et al. (2020)	246.54	379.88	376.79	225.63	329.81	396.55	358.34	365.83
Zhou et al. (2020)	298.49	398.13	358.91	200.38	203.88	395.75	311.97	302.91
Hu et al. (2020)	264.36	233.56	369.14	235.64	389.59	314.20	227.29	309.49
Sharma (2024)	227.60	335.66	215.48	390.91	395.46	346.65	209.33	355.87
Ours	**191.45**	**122.30**	**208.18**	**128.27**	**143.93**	**118.94**	**202.95**	**126.93**

Bold fonts indicate the best value.

TABLE 3 Ablation experiments on LSTM modules compare the accuracy and performance metrics of various methods from different datasets.

Model	TripAdvisor dataset				Expedia dataset			
	Accuracy (%)	Recall (%)	F1 score (%)	AUC (%)	Accuracy (%)	Recall (%)	F1 score (%)	AUC (%)
CNN	93.78	85.76	85.36	92.12	92.14	89.02	87.66	89.26
GRU	93.59	92.35	88.26	90.16	94.56	88.10	85.69	93.33
BiLSTM	85.88	92.53	89.46	85.35	88.10	83.99	85.30	88.99
SelfAM-Vtrans	**98.16**	**94.90**	**92.87**	**92.93**	**97.31**	**95.08**	**92.62**	**92.22**
Model	Yelp dataset				Open Images dataset			
	Accuracy (%)	Recall (%)	F1 score (%)	AUC (%)	Accuracy (%)	Recall (%)	F1 score (%)	AUC (%)
CNN	89.08	86.97	87.83	88.38	89.77	85.81	87.25	89.26
GRU	86.33	92.82	90.81	93.12	87.70	91.60	88.60	84.50
BiLSTM	93.87	87.49	86.98	84.25	93.92	93.17	85.31	89.49
SelfAM-Vtrans	**98.45**	**94.29**	**93.32**	**91.34**	**98.18**	**94.33**	**91.96**	**91.56**

Bold fonts indicate the best value.

Specifically, on the TripAdvisor dataset, our method achieved an accuracy of 98.16%, surpassing other methods, with Recall and F1 score reaching 94.9 and 92.87%, respectively. The AUC metric also scored high at 92.93%. Similarly, on the Expedia dataset, our method outperformed others with an accuracy of 97.31%, Recall of 95.08%, F1 score of 92.62%, and AUC of

TABLE 4 Ablation experiments on LSTM modules comparing the inference and training time of various methods from different datasets.

Method	TripAdvisor dataset				Expedia dataset			
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)
Swin-transformer	328.21	304.96	249.11	303.41	290.37	250.07	331.06	282.68
Graph-transformer	259.14	246.81	275.01	231.44	284.81	396.08	249.56	215.22
Transformer	269.87	338.84	376.72	389.20	229.19	360.81	279.19	305.23
SelfAM-Vtrans	**107.88**	**121.12**	**158.02**	**213.89**	**137.65**	**168.31**	**222.03**	**154.32**
Method	Yelp dataset				Open Images dataset			
	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)	Parameters (M)	Flops (G)	Inference time (ms)	Training time (s)
Swin-transformer	341.81	257.36	318.86	284.97	296.08	208.04	249.22	282.81
Graph-transformer	399.37	306.77	335.97	330.68	399.67	338.17	287.30	262.95
Transformer	255.55	257.07	341.24	351.83	379.39	391.93	335.80	370.03
SelfAM-Vtrans	**207.48**	**103.57**	**108.22**	**157.11**	**151.60**	**201.48**	**211.11**	**104.16**

Bold fonts indicate the best value.

92.22%. Furthermore, on the Yelp dataset, our method achieved an accuracy of 98.45%, Recall of 94.29%, F1 score of 93.32%, and AUC of 91.34%. On the Open Images dataset, our method's accuracy was 98.18%, Recall was 94.33%, F1 score was 91.96%, and AUC was 91.56%, showcasing consistent high performance across all datasets. Our method leverages the LSTM module's memory capabilities and contextual information, along with proposed improvements and new techniques, to enhance the model's understanding and classification ability for textual data. Overall, our approach demonstrated exceptional performance in the ablation study, showcasing high accuracy, recall, and F1 Scores, as well as strong results in the AUC metric, solidifying its superiority over comparative methods.

Table 4 presents presents the key performance metrics of Swin Transformer, Graph Transformer, Vanilla Transformer, and our method (Ours) on the four datasets. Our method (Ours) exhibits fewer parameters and lower computational complexity on all datasets. For example, on the TripAdvisor dataset, our method has only 107.88 M parameters and 121.12 G FLOPs, much lower than Swin Transformer (328.21 M, 304.96 G), Graph Transformer (259.14 M, 246.81 G), Vanilla Transformer (269.87 M, 338.84 G), and others. Similarly, on the other datasets (Expedia, Yelp, and Open Images), our method demonstrates similar advantages, reflecting our excellent performance in resource utilization efficiency. Additionally, our method performs exceptionally well in terms of inference time and training time. For instance, on the Open Images Dataset, our inference time is only 211.11 ms, significantly lower than Swin Transformer (249.22 ms), Graph Transformer (287.30 ms), Vanilla Transformer (335.80 ms), and others. Our training time is also competitive.Our method not only excels in accuracy and performance metrics but also shows significant advantages in key metrics such as parameter count, computational complexity, inference time, and training time. These advantages are of crucial importance for efficiency and cost in practical deployment and application, providing a solid foundation for real-world applications.

5 Conclusion and discussion

This paper aims to address the problem of travel route recommendation and proposes a method based on the Vision Transformer (ViT) and LSTM, combined with a self-attention mechanism. This method integrates visual features and sequence information to provide personalized travel route recommendations. Initially, the method utilizes the Vision Transformer (ViT) to extract visual features from images related to travel destinations or attractions. Subsequently, an LSTM model is used to encode the sequence of users' historical travel data, including visited locations, preferences, and trip durations. A self-attention mechanism is then introduced to capture relationships and dependencies between different travel features. Finally, the visual features extracted by the ViT and the sequence information encoded by the LSTM are integrated into a comprehensive model. In the experiments, a travel dataset containing users' travel preferences, historical routes, and destination attributes was used. The data were first cleaned and preprocessed, including removing duplicates, handling missing values, and encoding categorical variables. The pre-trained ViT model was then used to extract image features, and the LSTM model encoded the sequence information. Next, the self-attention mechanism was employed to capture the relationships between features, and the visual features and sequence information were integrated into a comprehensive model. The model was trained using historical travel data and target routes from the training set, learning model parameters. Finally, the model was used to recommend travel routes for new user queries

or contexts. Experimental results show that the travel route recommendation algorithm based on the Vision Transformer and LSTM combined with a self-attention mechanism achieves good performance in personalized recommendations. The algorithm can combine image features and sequence information to provide travel route suggestions that align with users' preferences and context.

However, this method also has some limitations. First, the method has a high demand for large datasets, particularly image data and users' historical travel data. Collecting and labeling large datasets may pose challenges and costs. Future research could explore how to effectively acquire and utilize limited data to improve model performance. Second, combining Vision Transformer and LSTM could increase computational complexity, especially during the recommendation phase. This may lead to decreased efficiency in real-time recommendation scenarios. Further research could explore how to reduce the model's computational complexity to provide efficient travel route recommendations in real-time applications. Future research could extend and improve the travel route recommendation algorithm based on Vision Transformer and LSTM in the following areas: Considering multimodal information: In addition to images and sequence information, integrating other types of information, such as user reviews and social media data, could more comprehensively model users' travel preferences and context. While this study focuses on the fusion of multimodal data to enhance the accuracy and personalization of travel route recommendations, we recognize the importance of incorporating user feedback for dynamic adaptation. As part of our future work, we plan to explore reinforcement learning-based methods to integrate user feedback into the recommendation process. This would enable the system to continuously adjust its recommendations based on real-time user interactions and preferences, further enhancing its ability to provide personalized and contextually relevant travel suggestions.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

ZJ: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. JZ: Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing. MG: Visualization, Supervision, Funding acquisition, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. National Social Science Foundation Art Project, Research on the driving mechanism and improvement path of digital cultural consumption under the guidance of expanding domestic demand, Project No: 23BH155.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2024.1439195/full#supplementary-material

References

Abbasi-Moud, Z., Vahdat-Nejad, H., and Sadri, J. (2021). Tourism recommendation system based on semantic clustering and sentiment analysis. *Expert Syst. Appl.* 167:114324. doi: 10.1016/j.eswa.2020.114324

Abdelraouf, A., Abdel-Aty, M., and Wu, Y. (2022). Using vision transformers for spatial-context-aware rain and road surface condition detection on freeways. *IEEE Trans. Intell. Transport. Syst.* 23, 18546–18556. doi: 10.1109/TITS.2022.3150715

Asghar, N. (2016). Yelp dataset challenge: review rating prediction. *arXiv preprint arXiv:1605.05362*. doi: 10.48550/arXiv.1605.05362

Chen, C., Zhang, S., Yu, Q., Ye, Z., Ye, Z., and Hu, F. (2021). Personalized travel route recommendation algorithm based on improved genetic algorithm. *J. Intell. Fuzzy Syst.* 40, 4407–4423. doi: 10.3233/JIFS-201218

Chen, L., Cao, J., Wang, Y., Liang, W., and Zhu, G. (2022). Multi-view graph attention network for travel recommendation. *Expert Syst. Appl.* 191:116234. doi: 10.1016/j.eswa.2021.116234

Duan, Z., Gao, Y., Feng, J., Zhang, X., and Wang, J. (2020). "Personalized tourism route recommendation based on user's active interests," in *2020 21st IEEE International Conference on Mobile Data Management (MDM)* (Versailles: IEEE), 729–734.

Egli, V., Mackay, L., Jelleyman, C., Ikeda, E., Hopkins, S., and Smith, M. (2020). Social relationships, nature, and traffic: findings from a child-centred approach to measuring active school travel route perceptions. *Child. Geograph.* 18, 667–683. doi: 10.1080/14733285.2019.1685074

Gandhi, M., Mistry, K., and Patel, M. (2014). A modified approach towards tourism recommendation system with collaborative filtering and association rule mining. *Int. J. Comput. Appl.* 91, 17–21. doi: 10.5120/15885-4685

Goldenberg, D., and Levin, P. (2021). "Booking.com multi-destination trips dataset," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2457–2462. doi: 10.1145/3404835.3463240

Gong, H., Zhu, X., Hu, Z., and Diao, J. (2023). "Two-stage trajectory generation model for realistic human mobility simulation," in *2023 IEEE International Conference on High Performance Computing & Communications, Data Science & Systems, Smart City & Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)* (Melbourne, VIC: IEEE), 934–941.

Hu, G., Qin, Y., and Shao, J. (2020). Personalized travel route recommendation from multi-source social media data. *Multimedia Tools Appl.* 79, 33365–33380. doi: 10.1007/s11042-018-6776-9

Ji, S., Wang, Z., Li, T., and Zheng, Y. (2020). Spatio-temporal feature fusion for dynamic taxi route recommendation via deep reinforcement learning. *Knowl. Based Syst.* 205:106302. doi: 10.1016/j.knosys.2020.106302

Jiang, R., and Dai, B. (2024). Cultural tourism attraction recommendation model based on optimized weighted association rule algorithm. *Syst. Soft Comput.* 6:200094. doi: 10.1016/j.sasc.2024.200094

Jin, L., Li, S., La, H. M., and Luo, X. (2017). Manipulability optimization of redundant manipulators using dynamic neural networks. *IEEE Trans. Industr. Electron.* 64, 4710–4720. doi: 10.1109/TIE.2017.2674624

Jin, L., Li, S., Yu, J., and He, J. (2018). Robot manipulator control using neural networks: a survey. *Neurocomputing* 285, 23–34. doi: 10.1016/j.neucom.2018.01.002

Jin, L., Zhang, Y., and Li, S. (2015). Integration-enhanced zhang neural network for real-time-varying matrix inversion in the presence of various kinds of noises. *IEEE Trans. Neural Netw. Learn. Syst.* 27, 2615–2627. doi: 10.1109/TNNLS.2015.2497715

Kbaier, M. E. B. H., Masri, H., and Krichen, S. (2017). "A personalized hybrid tourism recommender system," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)* (Hammamet: IEEE), 244–250.

Kesorn, K., Juraphanthong, W., and Salaiwarakul, A. (2017). Personalized attraction recommendation system for tourists through check-in data. *IEEE Access* 5, 26703–26721. doi: 10.1109/ACCESS.2017.2778293

Khan, I., Sadad, A., Ali, G., ElAffendi, M., Khan, R., and Sadad, T. (2023). NPR-LBN: next point of interest recommendation using large bipartite networks with edge and cloud computing. *J. Cloud Comput.* 12:54. doi: 10.1186/s13677-023-00427-5

Kong, W.-K., Zheng, S.-Y., Nguyen, M.-L., and Ma, Q. (2022). *A Multi-agent Reinforcement Learning Approach Towards Congestion-Aware Route Recommendation for Tourists. DEIM2022.* Available at: https://proceedings-of-deim.github.io/DEIM2022/papers/D24-5.pdf

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., et al. (2020). The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale. *Int. J. Comput. Vis.* 128, 1956–1981. doi: 10.1007/s11263-020-01316-z

Lahagun, P., Devkota, B., Giri, S., and Budha, P. (2024). Machine learning-based social media review analysis for recommending tourist spots. *J. Eng. Sci.* 3, 45–52. doi: 10.3126/jes2.v3i1.66234

Li, J. (2024). Tourist attraction recommendation model based on RFPAP-NNPAP algorithm. *Int. J. Adv. Comput. Sci. Appl.* 15:561. doi: 10.14569/IJACSA.2024.0150561

Lin, C., Mao, X., Qiu, C., and Zou, L. (2024). DTCNet: transformer-CNN distillation for super-resolution of remote sensing image. *IEEE J. Select. Top. Appl. Earth Observ. Rem. Sens.* 2024:3409808. doi: 10.1109/JSTARS.2024.3409808

Lin, Z., Wang, C., Li, Z., Wang, Z., Liu, X., and Zhu, Y. (2024a). Neural radiance fields convert 2D to 3D texture. *Appl. Sci. Biotechnol. J. Adv. Res.* 3, 40–44. Available at: https://www.abjar.vandanapublications.com/index.php/ojs/article/view/69

Lin, Z., Wang, Z., Zhu, Y., Li, Z., and Qin, H. (2024b). Text sentiment detection and classification based on integrated learning algorithm. *Appl. Sci. Eng. J. Adv. Res.* 3, 27–33. Available at: https://asejar.singhpublication.com/index.php/ojs/article/view/101

Lorenzi, F. (2007). "A multiagent knowledge-based recommender approach with truth maintenance," in *Proceedings of the 2007 ACM Conference on Recommender Systems*, 195–198. Available at: https://asejar.singhpublication.com/index.php/ojs/article/view/101

Nilizadeh, S., Aghakhani, H., Gustafson, E., Kruegel, C., and Vigna, G. (2019). "Think outside the dataset: finding fraudulent reviews using cross-dataset analysis," in *The World Wide Web Conference*, 3108–3115. doi: 10.1145/3308558.3313647

Park, S.-T., and Liu, C. (2022). A study on topic models using lda and word2vec in travel route recommendation: focus on convergence travel and tours reviews. *Person. Ubiquit. Comput.* 2, 1–17. doi: 10.1007/s00779-020-01476-2

Pramanick, S., Nowara, E. M., Gleason, J., Castillo, C. D., and Chellappa, R. (2022). "Where in the world is this image? transformer-based geo-localization in the wild," in *European Conference on Computer Vision* (Berlin: Springer), 196–215.

Renjith, S., Sreekumar, A., and Jathavedan, M. (2020). An extensive study on the evolution of context-aware personalized travel recommender systems. *Inform. Process. Manag.* 57:102078. doi: 10.1016/j.ipm.2019.102078

Sharma, R. (2024). "Genetic algorithm based personalized travel recommendation system," in *2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)* (IEEE), 867–874. Available at: https://ieeexplore.ieee.org/abstract/document/10467470/

Shi, L., Luo, J., Zhang, P., Han, H., El Baz, D., Cheng, G., et al. (2022). Understanding user preferences in location-based social networks via a novel self-attention mechanism. *Sustainability* 14:16414. doi: 10.3390/su142416414

Sun, K., Qian, T., Chen, T., Liang, Y., Nguyen, Q. V. H., and Yin, H. (2020). Where to go next: modeling long-and short-term user preferences for point-of-interest recommendation. *Proc. AAAI Conf. Artif. Intell.* 34, 214–221. doi: 10.1609/aaai.v34i01.5353

Wang, G., Hao, Z., Li, H., and Zhang, B. (2023). An activated variable parameter gradient-based neural network for time-variant constrained quadratic programming and its applications. *CAAI Trans. Intell. Technol.* 8, 670–679. doi: 10.1049/cit2.12192

Wang, M. (2020). Applying internet information technology combined with deep learning to tourism collaborative recommendation system. *PLoS ONE* 15:e0240656. doi: 10.1371/journal.pone.0240656

Wong, J. W. C., Lai, I. K. W., and Tao, Z. (2020). Sharing memorable tourism experiences on mobile social media and how it influences further travel decisions. *Curr. Iss. Tour.* 23, 1773–1787. doi: 10.1080/13683500.2019.1649372

Xiao, X., Jiang, C., Jin, L., Huang, H., and Wang, G. (2023). Nonlinear rnn with noise-immune: a robust and learning-free method for hyperspectral image target detection. *Expert Syst. Appl.* 229:120490. doi: 10.1016/j.eswa.2023.120490

Yang, S., Liu, J., and Zhao, K. (2022). "GETNext: trajectory flow map enhanced transformer for next poi recommendation," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1144–1153. doi: 10.1145/3477495.3531983

Yuan, Z. (2022). Big data recommendation research based on travel consumer sentiment analysis. *Front. Psychol.* 13:857292. doi: 10.3389/fpsyg.2022.857292

Zhang, X., Ge, Y., Wang, Y., Wang, J., Wang, W., and Lu, L. (2024). Residual learning-based robotic image analysis model for low-voltage distributed photovoltaic fault identification and positioning. *Front. Neurorobot.* 18:1396979. doi: 10.3389/fnbot.2024.1396979

Zhou, X., Su, M., Feng, G., and Zhou, X. (2020). Intelligent tourism recommendation algorithm based on text mining and MP nerve cell model of multivariate transportation modes. *IEEE Access* 9, 8121–8157. doi: 10.1109/ACCESS.2020.3047264

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 04 December 2024
DOI 10.3389/fnbot.2024.1443678

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Xiaoyin Zheng,
XMotors.ai, United States
Yuriy Kondratenko,
Petro Mohyla Black Sea State University, Ukraine

*CORRESPONDENCE
Touseef Sadiq
touseef.sadiq@uia.no
Ahmad Jalal
ahmadjalal@mail.au.edu.pk

RECEIVED 04 June 2024
ACCEPTED 18 November 2024
PUBLISHED 04 December 2024

CITATION
Abbas Y, Al Mudawi N, Alabdullah B, Sadiq T, Algarni A, Rahman H and Jalal A (2024) Unmanned aerial vehicles for human detection and recognition using neural-network model.
Front. Neurorobot. 18:1443678.
doi: 10.3389/fnbot.2024.1443678

Unmanned aerial vehicles for human detection and recognition using neural-network model

Yawar Abbas[1], Naif Al Mudawi[2], Bayan Alabdullah[3], Touseef Sadiq[4]*, Asaad Algarni[5], Hameedur Rahman[1] and Ahmad Jalal[1,6]*

[1]Faculty of Computer Science and AI, Air University, Islamabad, Pakistan, [2]Department of Computer Science, College of Computer Science and Information System, Najran University, Najran, Saudi Arabia, [3]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia, [4]Department of Information and Communication Technology, Centre for Artificial Intelligence Research, University of Agder, Grimstad, Norway, [5]Department of Computer Sciences, Faculty of Computing and Information Technology, Northern Border University, Rafha, Saudi Arabia, [6]Department of Computer Science and Engineering, College of Informatics, Korea University, Seoul, Republic of Korea

Introduction: Recognizing human actions is crucial for allowing machines to understand and recognize human behavior, with applications spanning video based surveillance systems, human-robot collaboration, sports analysis systems, and entertainment. The immense diversity in human movement and appearance poses a significant challenge in this field, especially when dealing with drone-recorded (RGB) videos. Factors such as dynamic backgrounds, motion blur, occlusions, varying video capture angles, and exposure issues greatly complicate recognition tasks.

Methods: In this study, we suggest a method that addresses these challenges in RGB videos captured by drones. Our approach begins by segmenting the video into individual frames, followed by preprocessing steps applied to these RGB frames. The preprocessing aims to reduce computational costs, optimize image quality, and enhance foreground objects while removing the background.

Result: This results in improved visibility of foreground objects while eliminating background noise. Next, we employ the YOLOv9 detection algorithm to identify human bodies within the images. From the grayscale silhouette, we extract the human skeleton and identify 15 important locations, such as the head, neck, shoulders (left and right), elbows, wrists, hips, knees, ankles, and hips (left and right), and belly button. By using all these points, we extract specific positions, angular and distance relationships between them, as well as 3D point clouds and fiducial points. Subsequently, we optimize this data using the kernel discriminant analysis (KDA) optimizer, followed by classification using a deep neural network (CNN). To validate our system, we conducted experiments on three benchmark datasets: UAV-Human, UCF, and Drone-Action.

Discussion: On these datasets, our suggested model produced corresponding action recognition accuracies of 0.68, 0.75, and 0.83.

KEYWORDS

neural network, sequential data processing, convolutional neural network (CNNs), decision-making processes, unmanned aerial vehicles neural network, unmanned aerial vehicles

1 Introduction

Recognizing human actions from drone-captured video is a challenging task that requires processing visual data to gather information about the motion of human and automatically identify human actions performed by humans. This approach is used in different applications and systems, like enhancing video based surveillance, human motion detection, sports activity analysis, human-robot interaction, and also in the rehabilitation process. For instance, in the rehabilitation process, we employ this approach when a patient suffers from a stroke and certain body parts are malfunctioning. As a result, we reduced the disability rate. In video surveillance, action recognition can help identify security threats like individuals hearting someone or using weapons to threaten someone, thus enhancing public safety and reducing and detecting criminal activities. In human-robot interaction, identifying the actions performed by humans can help the robots understand and classify humans' behaviors and react accordingly (Perera et al., 2019b). The medical field also employs this technique. Coaches use this technique to learn the player's physical health, performance, and team dynamics in support. Due to this, management and coaching staff decision-making power will increase, and they will know more about the player and be able to make better team selections and improve their success. In the field of gaming and entertainment, action recognition improves and makes the gaming experience more enjoyable. This is, as we know, a very interesting field, and most researchers do their research in it. Researchers still face so many challenges in this field. When we perform action recognition of a human, we must consider numerous factors such as the human's pose in the current frame of the video, the appearance of the object in the frame, whether the object is moving, the calculation of the object's speed, and time constraints. All of the factors mentioned above make it challenging to make an effective algorithm that works accurately across different settings.

For human action recognition, labeled data collection is an expensive and time-consuming procedure (Skakodub et al., 2021). We also have a smaller dataset available for training our model and getting accurate results. When we want to recognize the action of the human, first of all, we should understand the sequence of the human's moments in the given video. When we use video capture by drones, we face more difficulties because of the variety of camera viewpoints, as action may appear differently from various angles. Moreover, achieving real-time performance is crucial for applications like surveillance and robotics, while maintaining accuracy poses a significant challenge. Drone-mounted cameras add complexity as the image's background changes with the drone's motion (Sidenko et al., 2023). In a previous system, they developed an action recognition system based on traditional computer vision and applied some machine learning techniques to the RGB image and depth of the video data. This system has several steps, like splitting video into frames, using a bilateral filter for noise reduction, region extraction using SLIC segmentation, and body joint estimation using EM-GMM. As we already mentioned, the system uses the depth information of the video to detect the motion of the object, so this dependency on depth information reduces its acceptance because, in real life, we have very complex data and also because the environment may affect the process. We propose a new system that detects human action from aerial RGB videos, addressing the limitations of the previous work. Video capture by drone, so it did not relay in-depth information about the object. This system uses quick-shift segmentation to segment humans and extracts features. However, to enhance accuracy and performance, we propose a new system that concentrates on aerial RBG data and does not rely on depth information. This system uses a deep neural architecture like CNN instead of depth information. In this process, first of all, the RGB aerial video is converted into frames, a Gaussian blur filter is applied to remove noise and reduce the computational cost, and background effects are removed from the results. We also remove the background of the human, apply the YOLO algorithm to detect the human from the frames, and extract features such as angle between joints, distance between detected landmarks, 3D point cloud, and fiducial points. We use Kernel Discriminant Analysis (KDA) as an optimizer. CNNs optimize feature extraction and enhance action classification. Our proposed method shows highest performance compared with the existing previous version. With accurate human detection using YOLO and deep-learning-based feature extraction and classification, this system has gained acceptance. This study's key contributions include:

- A specialized approach that addresses the main challenges of human actions recognitions in aerial RGB videos makes our system independent of in-depth information and also increases the performance and accuracy of the system.
- Improved feature extraction and action classification through CNN's deep-learning model.
- Efficient human detection using the YOLO algorithm.
- 3D point cloud and fiducial point's algorithms aid in accurate action identification.
- Showing higher action recognition accuracy as compared with previous techniques.
- KDA is used as a feature optimizer.

2 Literature review

Researcher have made significant strides in developing computer vision algorithms for recognizing human actions in recent past years. In the literature related to our study, we distinguish between two main areas.

2.1 Human action recognition by machine learning

On the basis of motion patterns, Arunnehru et al. conduct research on human action classification and recognition, concentrating on examining how a subject's location changes over time. This system began by converting RGB input videos to grayscale and then applying a noise-removal filter to enhance the features. To extract the motion feature, they utilized the frame difference method, which calculates the intensity difference between two consecutive frames, to find the motion of moving object in given frame. Additionally, this uses traditional machine learning algorithms, which impact its accuracy and limit its ability to capture complex patterns across different action classes. For action classification, the system uses support vector machines (SVM) and random forest classifiers (Sun et al., 2021). To address these limitations, our proposed system incorporates deep-learning architectures, leverages spatial information in aerial RBG

videos, and utilizes a convolutional neural network for improved action recognition and classification. For the classification of human action from videos Zhen et al. use local methods based on spatio-and temporal interest points such as sparse coding, the Naïve Bayes nearest neighbor classifier, and a vector of locally aggregated descriptors. These local approaches were effective in the image domain, but their performance might not directly work on video data. To address the challenge, our new approach considers both spatial and temporal relationships found in the video sequences and successfully recognizes action. A new framework is introduced by Yang et al., which recognizes human actions in video sequences captured by a depth camera. They utilized a strategy called Super-Normal Vector to aggregate low-level polynomials into a discriminative representation. However, this proposed approach depends on depth information and does not fully rely on RGB. Our system analyzes RGB videos, not only the depth information of the object. It also analyzes the color and texture features of the video to understand human activations. A novel approach is proposed for action recognition using joint regression-based learning. This approach focuses mostly on dynamic appearance, not whole body features. In contrast, our proposed model first extracts the features of the whole body, then uses a deep-learning architecture to classify the given classes based on these features. This makes our system more robust and generalizable.

2.2 Human action recognition by deep learning

A completely connected deep (LSTM) system for human skeleton-based action identification was proposed by the authors. The study highlighted how the coexistence of skeletal joints naturally provides vital aspects of human behavior. In order to obtain this, a unique regularization approach was devised to learn the co-occurrence properties of the skeleton joints, and the skeleton was treated as input at each time slot. But without taking into account other modalities like RGB or depth information, this work concentrated only on skeleton-based representations. On the other hand, our method works directly with RGB films, eliminating the need for skeleton-based representations and allowing for the extraction of rich visual data from aerial imagery. Li et al. addressed the shortcomings of earlier approaches that mainly relied on short-term temporal information and did not explicitly represent long-range dynamics by introducing a unique strategy for action recognition termed VLAD for Deep Dynamics (VLAD3). Different layers of video dynamics were merged in VLAD3, with Linear Dynamic Systems (LDS) modeling medium-range dynamics and deep CNN features capturing short-term dynamics. Nevertheless, the reliance of that model on trained deep network (CNN) and the LDS model's linearity assumption may restrict its capacity to manage intricate non-linear temporal dynamics. Our method, on the other hand, works directly with RGB videos and does not merely rely on pre-trained networks. This allows us to extract rich visual information and capture non-linear temporal dynamics. To obtain a dependable long-term motion representation, Shi et al. (2017) introduced a novel descriptor called the Sequential Deep Trajectory Descriptor (sDTD). To address the issue of effectively capturing motion data over extended periods of time, the proposed sDTD projected dense trajectories into two-dimensional planes. A CNN-RNN network was trained to learn a meaningful representation for long-term motion by finding both spatial and temporal correlations in the motion data. However, this approach relied on dense trajectory extraction, which could be risky in settings with a lot of clutter or noise.

Our proposed method offers a solution by operating on RGB videos right away without requiring explicit trajectory extraction. Using the ability of hierarchical recurrent neural networks (HRNNs) to effectively simulate long-term contextual information in temporal sequences, Du et al. developed an end-to-end HRNN for skeleton-based action recognition. Rather than using the entire skeleton as input, the authors divided it into five pieces based on the physical characteristics of people. However, the majority of this strategy depended on skeleton data, which was not always readily available.

2.3 Human action recognition using drones

Sanjay Kumar et al. (2024) analysis on combination of facial recognition and object detection for drone surveillance. The authors present a new model that integrates analytical tools based on machine learning for the improvement of detection in real time. This they say, proves that integration of these technologies enhance the effectiveness as well as the efficiency of the surveillance process. Incorporating this work will help us show how similar methods can be used for recognizing human action and thus link object detection with human behavior analysis. Hybrid grey wolf algorithms for optimizing fuzzy systems are the focus of discussion in the paper by Kozlov et al. (2022). The authors describe a method for enhancing the flexibility and effectiveness of UAV control approaches. Through discussing the parametric optimization methods, this paper contributes to the understanding of how the control of drones needs to be improved in order to capture the human behavior in real life situations. Kozlov et al. (2024) describes an IoT control system for UAVs for meteorological measurements. To this end, the authors examine assorted communication protocols and control strategies that allow drones to operate on their own while gathering data. The significance for us in this study is the opportunity of implementing some of the IoT frameworks developed in this study for enhancing situation understanding for drones in human action identification tasks.

3 System methodology

The approach that we propose is designed to deal with these issues, particularly for RGB videos captured by drones. Our methodology entails dividing the video into individual frames and implementing several pre-processing procedures on these RGB frames. During pre-processing, our focus lies on reducing computational complexity, resizing image quality, and improving foreground object visibility by eliminating background noise. Additionally, we employ YOLO to detect humans within the frames, enabling us to extract human skeletal structures and identify key points representing crucial body parts (the head, neck, shoulders (left and right), elbows, wrists, hips, knees, ankles, and hips (left and right)., and belly button). These key points, including significant joints like the head, wrists, elbows, thighs, knees, and ankles, serve as the foundation for deriving normalized positions, angular relationships, distance measurements, and 3D point clouds. To optimize features,

we utilize the Kernel Discriminant allocation approach (kDA), followed by classification using CNN. Our experimentation was carried out on three standard benchmark datasets: UCF, Drone-Action, and UAV-Human. The model accomplished recognition of the appropriate action accuracies of 0.75, 0.83, and 0.69 on these datasets. Figure 1 shows the architectural layout of the suggested system.

3.1 Preprocessing

In our proposed system, we utilize a dataset comprised of drone footage to train our model. The UAV-Human, UCF, and Drone-Action datasets consist of video recordings; thus, our system takes a video as its input. Since the algorithms employed in our system operate on images, the initial step involves converting the video into individual frames. These frames or images extracted from the video undergo Gaussian blur processing to reduce noise. By using Equation 1.

$$G(a,b) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (1)$$

In this equation, G (a, b) denotes the value of the Gaussian function at coordinates (x, y). The formula calculates the weight of each pixel in an image's local neighborhood using a Gaussian kernel. This kernel is represented by a two-dimensional matrix where the weights decrease from the center, where the highest weight pixel is positioned. The parameter σ corresponds to the standard deviation of the Gaussian distribution(Chen et al., 2023). A higher σ value results in more pronounced blurring of the image. This mathematical representation allows for the convolution of the image with the Gaussian kernel, effectively reducing noise and enhancing the image's quality. Following the Gaussian blur process, the images remain in the RGB color space (Papaioannidis et al., 2021). However, since our focus is not on color but rather on image description, which can sometimes impact the information within the image, we further process the images. To achieve this, we utilize the blurred images as input and apply a grayscale conversion algorithm to them, aiding in noise reduction. By using Equation 2.

$$S = 0.299R + 0.587G + 0.114B \quad (2)$$

This equation represents the luminance (S) value calculated from the RGB components of a color image. The original frame and the frame following preprocessing are illustrated in Figure 2.

3.2 Human detection

Computer vision and deep learning intersect in the realm of identifying and locating objects or humans within images, offering wide-ranging applications across fields like robotics, autonomous vehicles, and drone-based surveillance systems. We commonly categorize detection algorithms into two primary types: single-shot detector algorithms and two-stage detector algorithms. One notable approach for object detection is YOLOv9 (You Only Look Once), which has been pivotal in transforming the field (Sobhan et al., 2021). YOLOv9 stands out for its ability to predict object attributes in a single pass, greatly boosting real-time performance and achieving top-tier results. YOLO's strength lies in using a single fully connected layer for its predictions, unlike methods like Faster R-CNN that rely on a region proposal network and separate recognition steps. This streamlined strategy significantly reduces computational load, requiring only one iteration per image

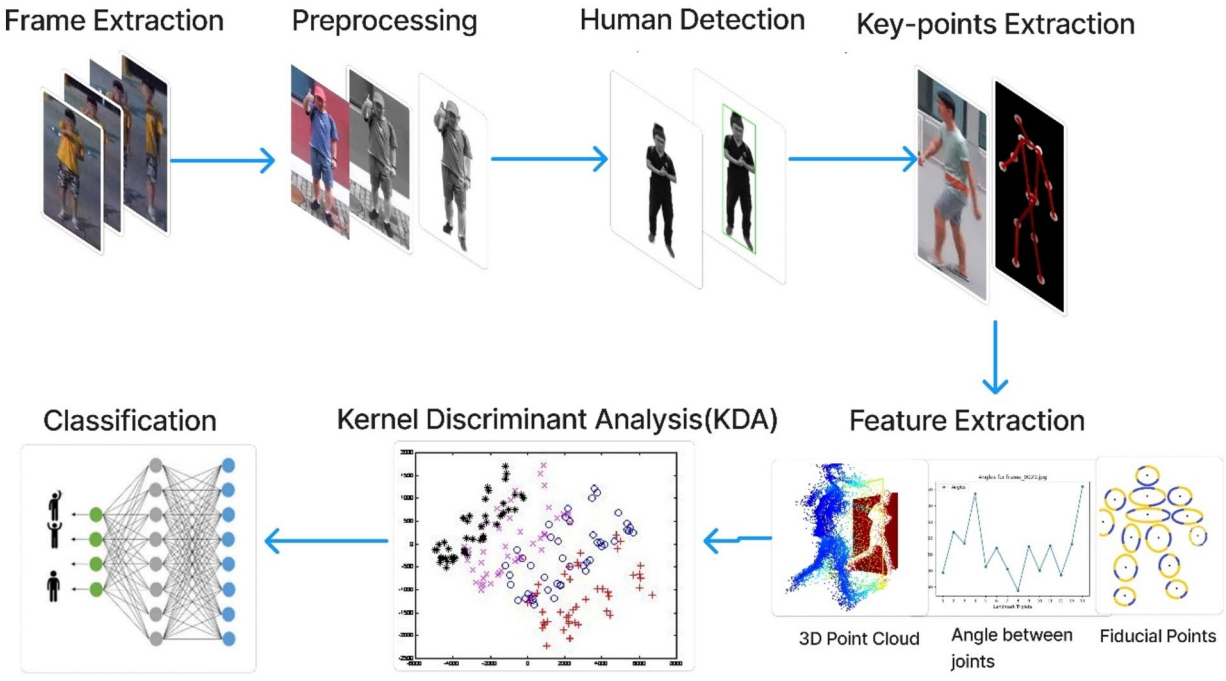


FIGURE 1
The architecture of the proposed system.

compared to the multiple iterations needed by approaches using region proposal networks (Hwang et al., 2023).

When tailoring the YOLOv9 algorithm for individual detection, the main goal is to accurately forecast bounding boxes with strong confidence scores, particularly for the human class. This necessitates fine-tuning the training process and potentially adjusting the YOLOv9 network's architecture to concentrate specifically on human detection. We introduce adjustments to interpret outputs from a human-centric viewpoint, while keeping the core equations governing the algorithm unchanged. The prediction of bounding boxes remains central, with a focus on identifying boxes with notable probabilities of containing a human. Consequently, during inference, we retain only bounding boxes associated with humans, eliminating those related to other object classes. Simplifying the class prediction process by considering solely the confidence score for the human class further bolsters detection accuracy. By using Equation 3.

$$A_{i,j,c} = q_{i,j,c} \times Tr\left(Object\right) \times IOU\left(d_{i,j}, d^{truth}\right) \tag{3}$$

In this equation:

$A_{i,j,c}$ represents the predicted bounding box for class c at grid cell i, j. $q_{i,j,c}$ this is the confidence score for the presence of an object within that bounding box. $Tr\left(Object\right)$ this is the probability that an object exists in the box. $IOU\left(d_{i,j}, d^{truth}\right)$ this represents the ground truth box truth and the expected box's intersection over union (IOU) (see Figure 3).

In Table 1 we displays the accuracy rates of various YOLO models evaluated on three distinct datasets: There are three datasets namely UAV-Human, UCF, and Drone-Action. This research presents results demonstrating that with each subsequent release of YOLO, there is an improvement in the model's precision, described broader improvements in human action recognition capacity. Starting with YOLOv1, one can see that on all the analyzed datasets, there is a continuous growth in accuracy with the trends of improvement in the model architecture and training processes from YOLOv1 to YOLOv9. This progression suggests further development for deep learning models for the processing of the aerial imagery (Jiang and He, 2020; Nadeem et al., 2020).

3.3 Key-points extraction

The Yolo algorithm is employed to analyze images extracted from videos, facilitating human detection within these images.

FIGURE 2
Preprocessing outcomes for **(A)** Drone Action **(B)** UAV human.

FIGURE 3
We observe the YOLO method in action for human detection, with representations for **(A)** Drone Action **(B)** UAV human.

TABLE 1 Comparison of YOLO versions with proposed model.

Propose Model + YOLOv	UAV-human accuracy	UCF accuracy	Drone-action accuracy
YOLOv1	0.50	0.60	0.65
YOLOv2	0.55	0.64	0.69
YOLOv3	0.60	0.69	0.75
YOLOv4	0.63	0.71	0.78
YOLOv5	0.65	0.72	0.84
YOLOv6	0.66	0.73	0.86
YOLOv7	0.67	0.73	0.89
YOLOv8	0.68	0.74	0.91
YOLOv9	**0.68**	**0.75**	**0.92**

Bold value indicates the accuracy of my system when i use yolov9.

Subsequently, critical points of the human body are identified to enable further analysis. An Opencv pose estimator is utilized for human skeleton detection within an image, a pivotal step in determining the precise position of each body part. This skeleton is instrumental in calculating the angles and distances between joints of the human body. Our proposed system relies on 15 key points: head, neck, shoulders (left and right), elbows, wrists, hips, knees, ankles, and hips (left and right), and belly button. These identified key points contribute to height accuracy within our system. Notably, Opencv does not detect the neck, belly button, or specific key points besides the head. To address this, we compute the midpoints of these landmarks. For instance, the midway of the left and right shoulders is used to calculate the position of the neck. The calculation of midpoints between two given key points is based on their respective x and y coordinates (see Figure 4).

$$Am = \frac{(a_1 + a_2)}{2} \tag{4}$$

$$Bm = \frac{(b_1 + b_2)}{2} \tag{5}$$

Where:

(a1, b1) keypoint 1 coordinates and (a2, b2) keypoint 2 coordinates. To calculate the midpoint between two key points (*Am, Bm*), Equations 4, 5 are employed: This method allows us to precisely locate three specific critical points within the human body. Figure 5 provides a summary of identified landmarks belonging to various categories.

3.4 Feature extraction for action recognition

During the system development process, considerable attention is devoted to selecting features that effectively represent the outcomes. Optimal feature selection is crucial for attaining desirable results, given its substantial influence on system accuracy. The chosen features must possess autonomy and reliability. We extract multiple features from the photos and aggregate their numerical values into a single file for subsequent analysis (Chéron et al., 2015).

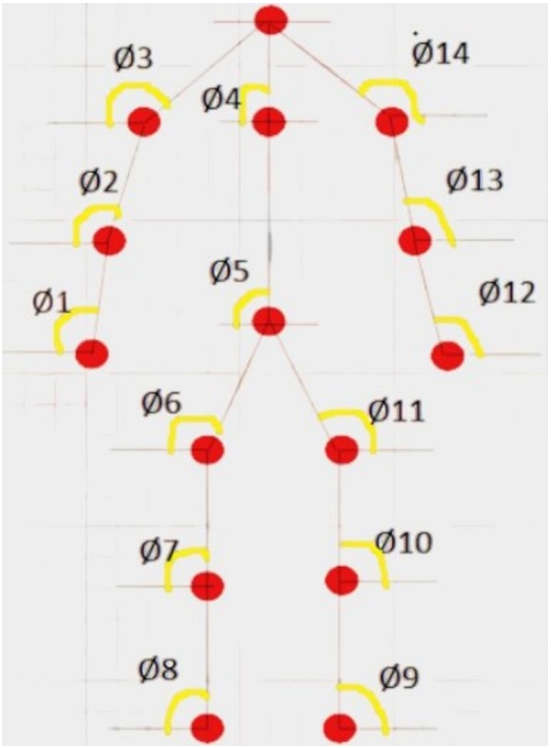


FIGURE 4
Relative joint angles for body key-points.

3.4.1 Relative angle between joints

The orientation of the body during various movements is determined by the angles formed between joints or specific anatomical points that we identify (Reddy et al., 2016). These angles dynamically alter relative to each other as humans engage in different actions. Continuously monitoring these angles as subjects move aids in enhancing the precision of our system. To achieve this, we focus on tracking fifteen key points across the body. The angle between two points was calculated using the following Equation 6:

$$\varphi = \tan{-}1\left(\mathrm{b2} - \mathrm{b1} / \mathrm{a2} - \mathrm{a1}\right) \tag{6}$$

Here, (a1, b1) and (a2, b2) indicate the coordinates of the two spots that are being examined. Figure 6 demonstrates the angles computed as one-dimensional signals for some activities.

3.4.2 Relative distance between joints

Once a human starts moving, every single one of their body parts moves until it stops. The measurement of this motion involves assessing the distance traveled by various key points from one frame to the next. This evaluation typically employs a comparison of two consecutive frames. Utilizing the Euclidean distance formula, expressed as Equation 7, facilitates the calculation of the distance between these key points:

$$v = \frac{\Delta t}{\Delta d} \tag{7}$$

Where Δd represents the change in distance between two points (the relative distance between joints in this context). Δt is the change in time

FIGURE 5
Key-points extraction-with: **(A)** Drone Action **(B)** UAV human.

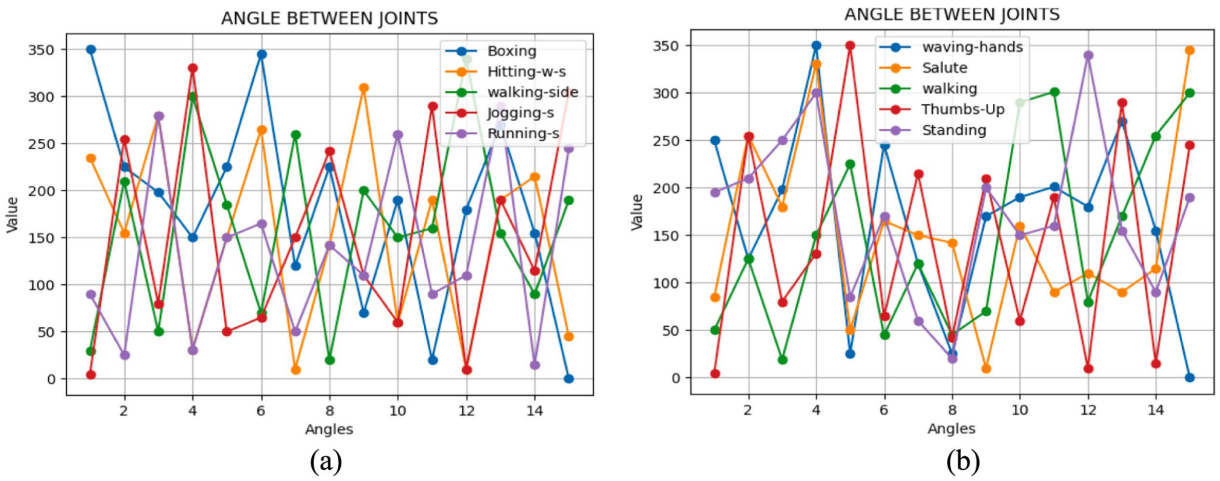


FIGURE 6
We examined the angular positions of the joints during various movements with **(A)** Drone Action **(B)** UAV human.

between two frames. This formula quantifies the change rate of distance as compare with time, offering insights into the pace at which the distance between joints alters as the body undergoes motion (see Figure 7).

3.4.3 Landmark fiducial points

Fiducial points serve as crucial landmarks within an image and are utilized for various calculations. Our proposed system employs fifteen such points, such as the head, neck, shoulders (left and right), elbows, wrists, hips, knees, ankles, and hips (left and right), and belly button. The successful detection of these landmarks in each frame of the provided video greatly facilitates object motion detection through their positional data (Guo et al.,2022). These points are strategically positioned along the contours of each body part, and their visualization is achieved through the ellipsoids encompassing these body regions. Within the ellipsoid, where the interior is depicted in black, transitions from high to low values signify points along the right border, while transitions from low to high values denote points along the left edge. We then determine the local minima and maxima for each border after accurately identifying the left and right borders. Equations 8, 9 articulate this mathematical process (see Figure 8).

$$\text{maxima} = \left\{ ai \mid ai^{'} \geq 0 \text{ and } ai+1^{'} < 0 \right\} \tag{8}$$

$$\text{minima} = \left\{ ai \mid ai^{'} \leq 0 \text{ and } ai+1^{'} > 0 \right\} \tag{9}$$

3.4.4 3D point cloud

In our proposed system, we leverage the representation of objects in 3D space, a widely employed feature across various applications, for tracking object motion. Specifically, we focus on utilizing the x, y, and z dimensions of the central pixel within an RGB image. To determine the z coordinate, we employ both the relative RGB image and its grayscale counterpart, enabling us to calculate the z coordinate of the pixel. Utilizing the YOLO algorithm for human detection in images, our process initiates by identifying humans within the image. Subsequently, the algorithm identifies the central pixel value within the YOLO bounding box encompassing the object. Upon locating the central pixel, the algorithm proceeds to iterate through all pixels within the bounding box, facilitating the determination of the z coordinate of each pixel using Equation 10.

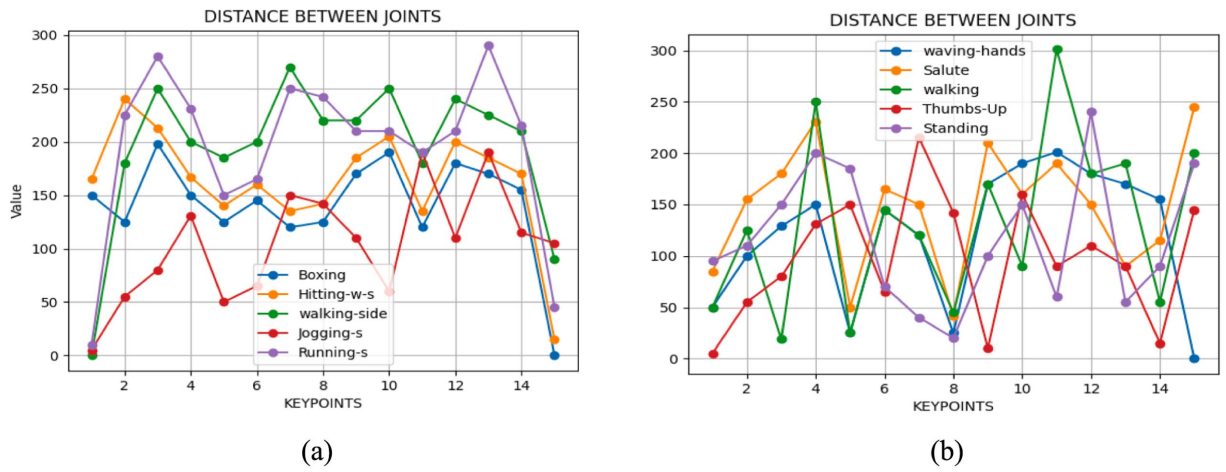


FIGURE 7
We examined the angular distance between joints during various movements with **(A)** Drone Action **(B)** UAV human.

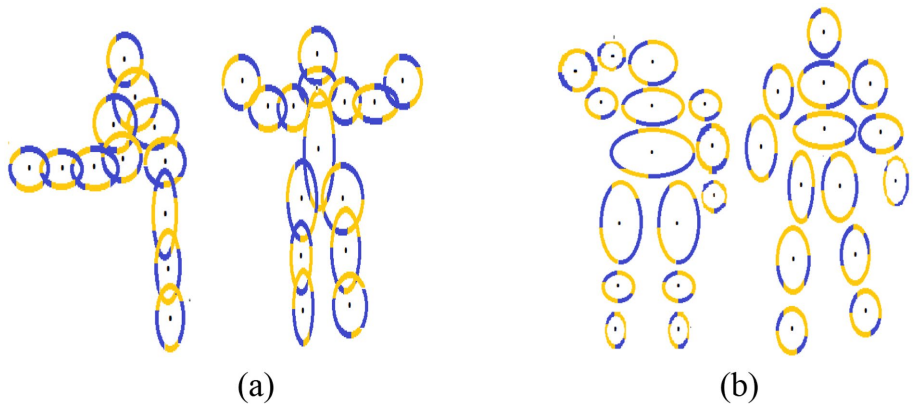


FIGURE 8
Results of fiducial points on **(A)** Drone Action **(B)** UAV human.

$$\frac{1}{Scaling\ Factor} \times Grey(p,q) \tag{10}$$

Where p, q are the x, y-coordinates of the pixel that is under observation. P, Q are calculated by Equations 11, 12.

$$P = X = \frac{Z}{Focal\ Length} \times (P - Cp) \tag{11}$$

$$Q = Y = \frac{Z}{Focal\ Length} \times (q - Cq) \tag{12}$$

The algorithm presented in this study is designed to identify all the pixel values corresponding to objects within an image and then compile them into an Excel file. This Excel file serves as a basis for applying a voxel filter, allowing visualization of these pixels within a 3D space (Azmat et al., 2023). The classification of these pixel values is essential for enhancing the accuracy of our system. Figure 9 illustrates the resulting point clouds.

Algorithm 1 shows the working of 3D point cloud algorithm.

Algorithm-1 Generating point cloud from silhouette image

 # Input:
image path: Path to the image
- F: Focal length
- SF: Scaling factor
 # Output:
# - Downsampled point cloud saved as a CSV file
try:
 # Specify Output Folder
output_folder = specify_output_folder()
 # Initialize Parameters
F = 100 # Replace with actual focal length
SF = 1.0 # Replace with actual scaling factor
 # Calculate Central Pixel Coordinates
Cx, Cy = calculate_central_pixel_coordinates(silhouette_image)
 # Initialize Point Cloud
point_cloud = []

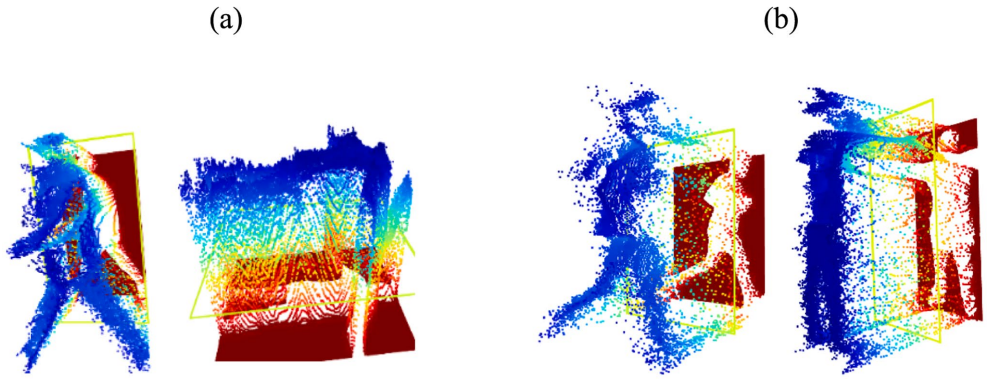


FIGURE 9
Results of 3D point cloud feature on **(A)** Drone Action **(B)** UAV human.

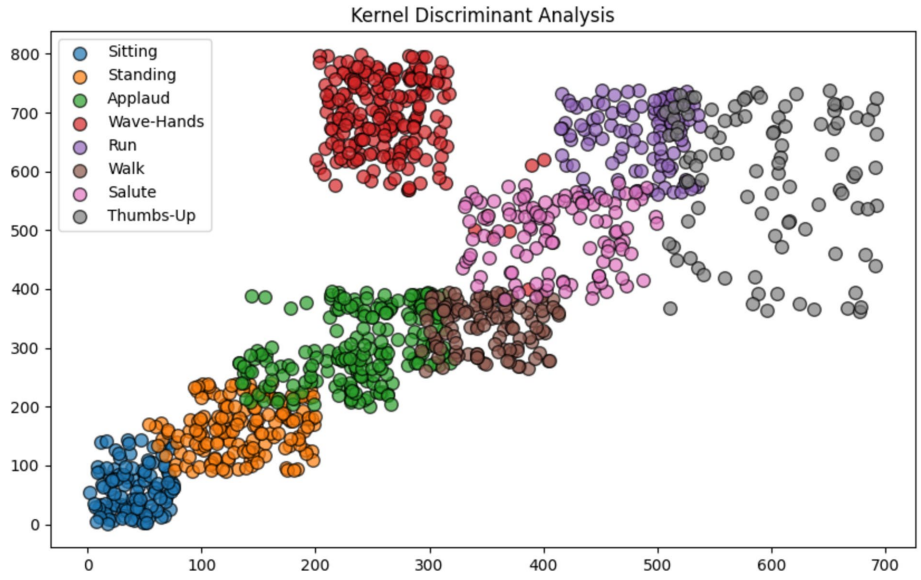


FIGURE 10
Enhanced feature allocation via kernel discriminant analysis (KDA).

```
# Iterate Over Image Pixels
for v in range(silhouette_image.shape[0]):
    for u in range(silhouette_image.shape[1]):
        X, Y, Z = calculate_3d_coordinates(u, v, silhouette_image,
        F, SF, Cx, Cy)
        point_cloud.append([X, Y, Z])
# Convert to NumPy Array
point_cloud_np = np.array(point_cloud)
# Convert to Open3D Point Cloud
o3d_point_cloud = convert_to_open3d_point_cloud
(point_cloud_np)
# Downsample the Point Cloud
downsampled_point_cloud = downsample_point_cloud
(o3d_point_cloud)
# Save Downsampled Point Cloud
save_point_cloud(downsampled_point_cloud, output_folder)
print("Downsampled point cloud saved successfully")
except Exception as e:
    print(f"An error occurred: {e}")
```

3.5 Kernel discriminant analysis

Kernel Discriminant Analysis (KDA) stands out as a method in machine learning, emphasizing the identification of a blend of features that effectively distinguishes classes within a dataset. Unlike the conventional approach of Linear Discriminant Analysis (LDA), which presupposes the linearity of data separability, KDA employs a kernel function to transform data into a higher-dimensional space where potential linear separability may exist. This adaptation enables KDA to handle datasets with non-linear separability more adeptly compared to LDA. By prioritizing the maximization of the ratio between-class variance and within-class variance, KDA strives to uncover a projection that optimizes the discrimination among various classes. This technique

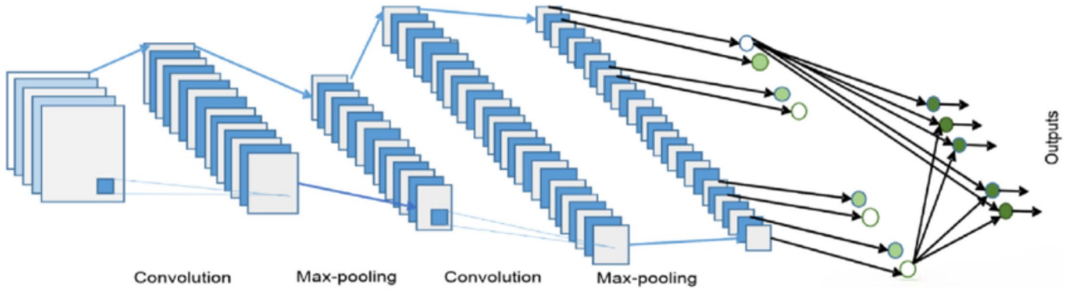


FIGURE 11
CNN architecture for proposed system.

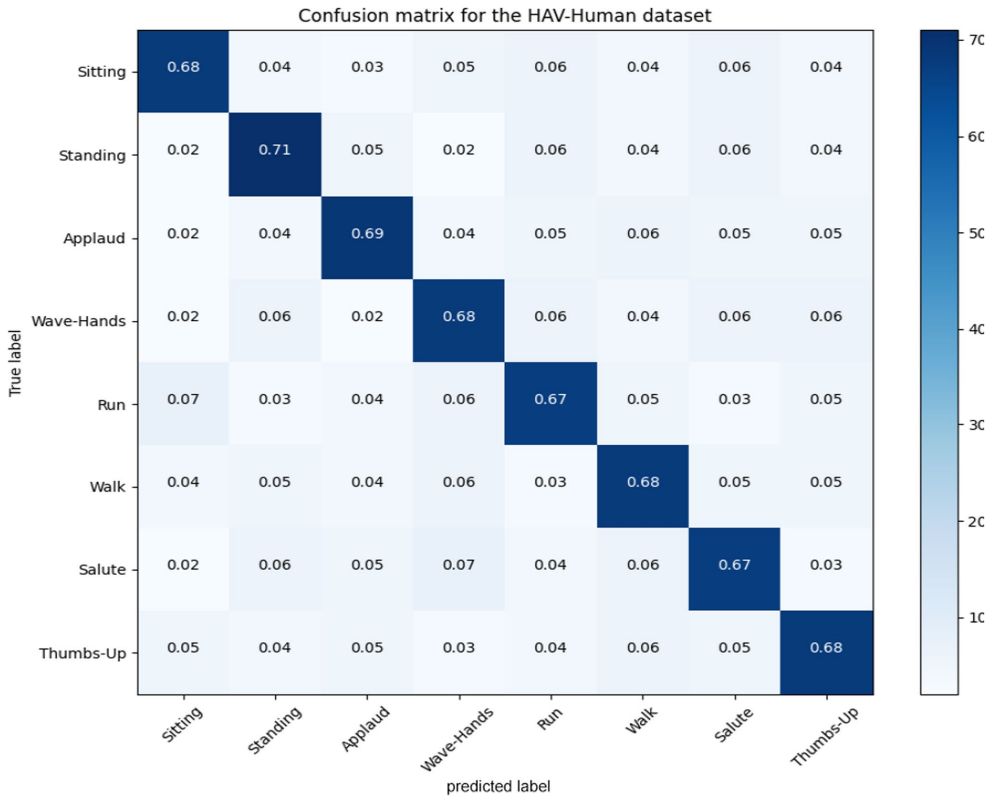


FIGURE 12
Confusion matrix for the UAV-human dataset.

finds applications across diverse domains such as pattern recognition, computer vision, and bioinformatics, where addressing classification challenges characterized by intricate decision boundaries is paramount. By using Equation 13.

$$KBw = \lambda KWw \tag{13}$$

3.6 Classification

In the classification process, a Convolutional Neural Network (CNN) is employed (Azmat et al., 2023). The general equation governing the convolutional operation within a CNN is outlined by using Equation 14:

$$F_{ijk} = \sum_{p=0}^{k-1}\sum_{q=0}^{k-1}\sum_{c=0}^{cin-1} Q_{pqck} P_{i+p,j+q,c} + b_k \tag{14}$$

In this case, P stands for the input matrix, Q for the weights, b for the bias, and F for the convolutional layer's output. The suggested CNN architecture for classifying human actions is shown in Figure 10. The features are first formatted and supplied into the CNN model. 32 filters with a stride of 1 are first applied. The input size is then decreased by implementing a max pooling layer of size. Next, another max pooling layer of size is applied, after which 64 convolutional filters of size and stride of 1 are applied. This is where the outcome size becomes. Next, a layer that is flattened and then densely placed. Ultimately, the probability

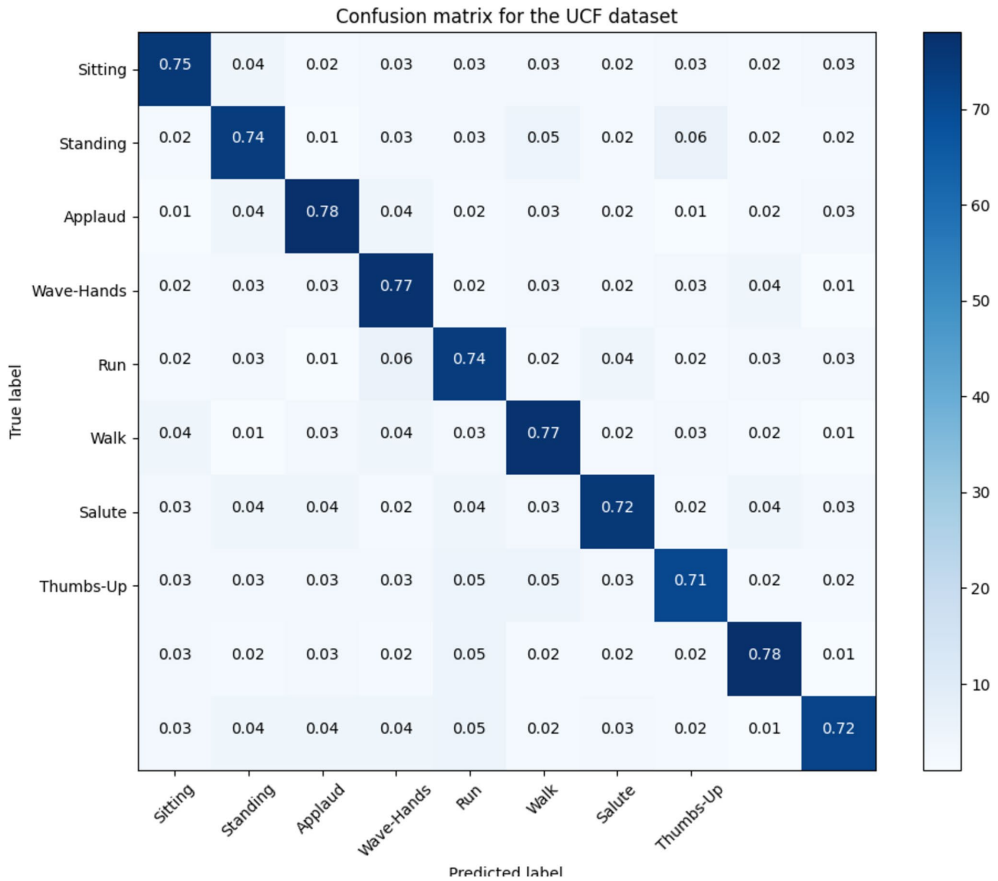


FIGURE 13
Confusion matrix for UCF dataset.

distribution for the final forecast is produced by the softmax function (see Figure 11).

4 Experimental setup and datasets

4.1 Experimental setup

To carry out the experiments outlined in this study, a laptop equipped with an Intel Core i5 CPU and 8 GB of RAM was utilized. The operating system employed was a 64-bit version of Windows 10, along with the pyCharm integrated development environment for programming tasks. Furthermore, the research involved capturing RGB footage utilizing a drone camera, capturing various perspectives. Three benchmark Human Activity Recognition (HAR) datasets were employed, specifically the Drone-Action dataset.

4.2 Dataset description

4.2.1 HAV human dataset

The UAV-Human dataset encompasses a diverse array of human activities, comprising 67,428 videos captured with the participation of 119 individuals over a duration of three months. These recordings were conducted in both urban and rural settings, facilitated by Unmanned Aerial Vehicles (UAVs), thereby presenting a multitude of challenges such as varied backgrounds, occlusions, weather conditions, and camera movements. This study focuses on eight specific human action categories extracted from the UAV-Human dataset: sitting down, standing up, applauding, waving hands, running, walking, giving a thumbs-up, and saluting.

4.2.2 UCF dataset

The UCF Ariel Video Dataset is a curated collection of aerial footage intended for academic exploration in computer vision and machine learning. It contains a diverse selection of scenes captured from aerial viewpoints, including urban and rural environments, as well as various weather conditions. Researchers leverage this dataset to develop and assess algorithms for tasks such as object detection, tracking, and understanding aerial scenes, without relying on AI-generated content.

4.2.3 Drone Action dataset

Within the Drone-Action dataset, there exist 13 distinct categories, namely: boxing, clapping, hitting-bottle, hitting-stick, jogging-front, jogging-side, kicking, running-front, running-side, stabling, walking-front, walking-side, and waving hands. This dataset diverges from an object-oriented structure due to instances where multiple entities engage in identical actions simultaneously. Each class in the dataset

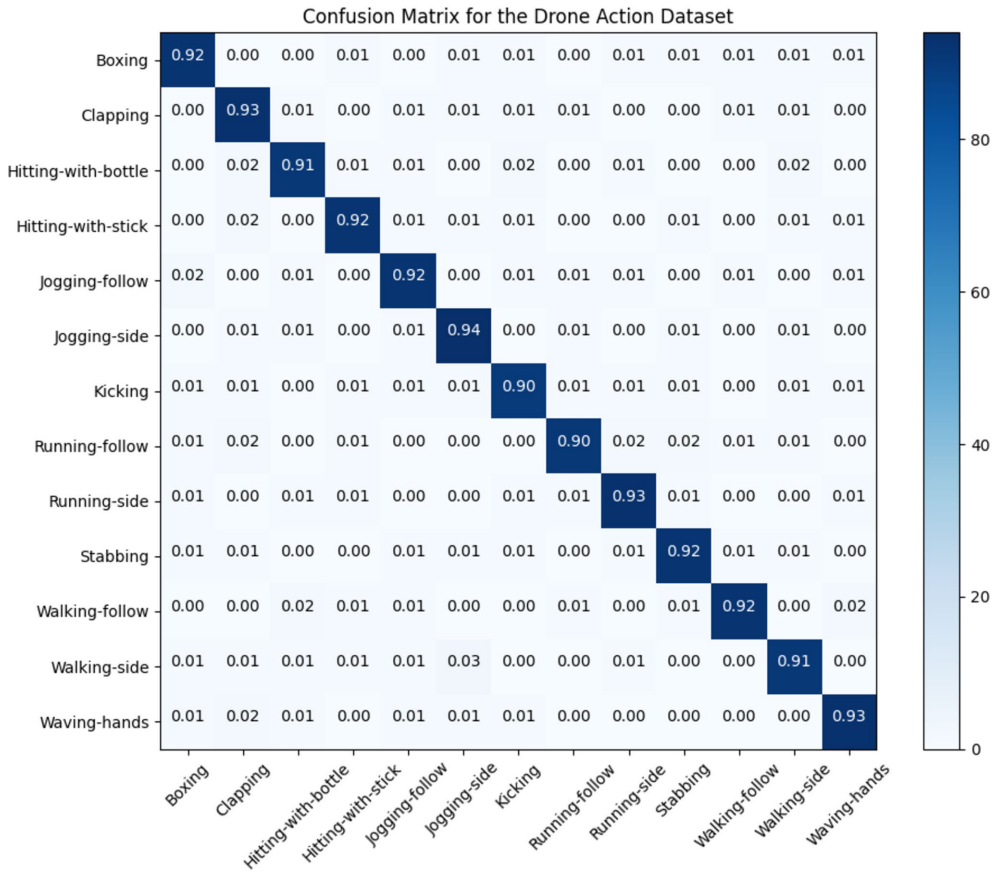


FIGURE 14
Confusion matrix for Drone Action dataset.

TABLE 2 Performance evaluation of the proposed system over UAV-Human dataset.

Classes	Accuracy	Precision	Recall
Sitting	0.68	0.68	0.74
Standing	0.71	0.71	0.69
Applaud	0.69	0.69	0.71
Wave-hands	0.68	0.68	0.67
Run	0.67	0.67	0.66
Walk	0.68	0.68	0.66
Salute	0.67	0.67	0.65
Thumbs-up	0.68	0.68	0.68
Average	**0.68**	**0.67**	**0.66**

Bold value indicates the results of my system.

TABLE 3 Performance evaluation of the proposed system over UCF dataset.

Classes	Accuracy	Precision	Recall
Boxing	0.75	0.77	0.75
Carrying	0.75	0.73	0.74
Clapping	0.75	0.76	0.78
Digging	0.75	0.71	0.77
Jogging	0.75	0.70	0.74
Running	0.75	0.73	0.72
Throwing	0.75	0.77	0.75
Trunk	0.75	0.75	0.71
Walking	0.75	0.78	0.77
Waving	0.75	0.79	0.74
Average	**0.75**	**0.74**	**0.73**

Bold value indicates the results of my system.

comprises a collection of video clips, ranging from 10 to 20 clips per class.

5 Results and analysis

In this section, we performed different experiments for the proposed system. The system is evaluated using different matrices, including confusion matrix, precision and recall.

5.1 Confusion matrices

In this section; we discussed performance analytics of all 3 benchmarks datasets used in the field of unmanned aerial vehicles for human detection and recognition. Figures 12–14 presents' confusion matrix of human interaction

recognition over UAV-Human, UCF and Drone Action datasets, respectively.

5.2 Precision, recall, and F1 score values for locomotion activities

Tables 2–4 presented the comparison of each class with their precision, accuracy and recall values.

Our system get UAV-Human = 0.68, UCF = 0.75, and Drone-Action = 0.83. We recognize that accuracy by itself is not enough to define the reliability of such a system, especially if it is used for applications like surveillance, search and rescue, or working closely with people. Our system is built to address typical issues arising with drone-based object tracking, including complex background, object occlusion and illumination variation. Some of the procedures performed here help to gain higher image's contrast – the objects in the foreground will be more easily detected which will definitely improve recognition in non-ideal conditions. The capacity for withstanding broad variations in the environment is useful in making certain that the system will perform well optimally after implementing it in real field use.

TABLE 4 Performance evaluation of the proposed system over Drone-Action dataset.

Classes	Accuracy	Precision	Recall
Boxing	0.92	0.92	0.92
Clapping	0.92	0.89	0.91
Hitting-w-b	0.92	0.92	0.91
Hitting-w-s	0.92	0.93	0.92
Jogging-f	0.92	0.91	0.92
Jogging-s	0.92	0.91	0.92
Kicking	0.92	0.91	0.90
Running-f	0.92	0.91	0.90
Running-side	0.92	0.91	0.91
Stabbing	0.92	0.91	0.90
Walking-s	0.92	0.92	0.82
Walking-f	0.92	0.91	0.90
waving	0.92	0.91	0.90
Average	**0.92**	**0.91**	**0.90**

Bold value indicates the results of my system.

5.3 Ablation study analysis of propose model components

We perform an ablation study in Table 5 to evaluate our model by systematically removing components one at a time. Every row describes the model with one element omitted and the accuracy on UAV-Human, UCF, and Drone-Action datasets. Table 5 also shows how important each of these elements is for achieving high accuracy.

5.4 Analyzing time complexity and executing time

Understanding time complexity of different processes is critical to the efficiency of the machine learning and computer vision tasks. Time complexity computation helps us identify slow activities within the system and estimate the impacts of certain techniques on run-time. Data preprocessing is critical in enhancing the efficiency of our model functions most importantly in the area of recommendation. Preprocessing Execution Time and Preprocessing Time Complexity of Critical Processes in Our Model (with and without) The empirical results reveal that preprocessing can greatly enhance efficiency as many processes are transformed from linear or quadratic to logarithmic. First, this kind of transition reduces the execution time by almost half and increases the system's throughput, making it beneficial for real-time applications such as action recognition and the field of study. Table 6 shows the computational cost of all steps of given system.

TABLE 5 An ablation experiment evaluating all methods across different datasets.

Experiments	Preprocessing	Human detect	Key-point extraction	JA	RD	FP	PC	KDA	CNN	UAV human	UCF	Drone Action
Full model	✓	✓	✓	✓	✓	✓	✓	✓	✓	68	75	92
Preprocessing	✗	✓	✓	✓	✓	✓	✓	✓	✓	63	70	86
Human detection	✓	✗	✓	✓	✓	✓	✓	✓	✓	60	69	84
Key-point extraction	✓	✓	✗	✓	✓	✓	✓	✓	✓	63	71	87
Without KDA	✓	✓	✓	✓	✓	✓	✓	✗	✓	61	68	84
Without pre + Key points	✗	✓	✓	✗	✓	✓	✓	✓	✓	58	65	81
Without pre + point clouds	✗	✓	✓	✓	✓	✓	✗	✓	✓	62	69	85
Without Pre + KDA	✗	✓	✓	✓	✓	✓	✓	✗	✓	59	66	83

JA = Joint Angle, RD = Relative Distance, FP = Fiducial Points, PC = Point Clouds, KDA = Kernel Discriminant Analysis, CNN = Convolutional Neural Network.

TABLE 6 Processing efficiency analysis and execution time.

Process	Without preprocessing	With preprocessing	Execution time without preprocessing (s)	Execution time with preprocessing (s)	Reduction in time complexity
Preprocessing	N/A	O(n)	N/A	0.1	Enhance efficiency
Human detection	O(n log n)	O(log n)	4.0	1.2	Notable improvement
Key-point extraction	O(n)	O(log n)	2.5	0.8	More efficient extraction
Joint angle	O(n)	O(log n)	2.0	0.6	Faster computations
Relative distance	O(n)	O(log n)	1.8	0.5	Optimized calculations
Fiducial points	O(n)	O(log n)	1.7	0.6	Quicker identification
Point cloud	O(n log n)	O(log n)	5.0	1.5	Improved noise reduction
KDA	$O(n^2)$	O(n log n)	8.0	3,0	Reduced dimensionality
CNN	O(n log n)	O(log n)	10,0	3.5	Enhanced feature processing

TABLE 7 Comparisons of the recognition accuracies between proposed method and other state-of-the-arts methods.

Methods	UAV Human	UCF	Drone Action
Baseline (SGN) (Xu et al., 2022)	0.39	-	-
MSST-RT (Sun et al., 2021)	0.41	-	-
P-CNN (Perera et al., 2019a)	-	-	0.75
SWTF + Pose-Stream (Yadav et al., 2023)	-	-	0.78
CNN (Azmat et al., 2023)	0.44	-	0.90
Proposed system mean accuracy	**0.68**	**0.75**	**0.92**

Bold value indicates the results of my system.

5.5 Comparison

In this experiment, we have compared our proposed method with other popular state-of-the-art methods over all 3 datasets. Table 7 provided a significant improvements in recognition accuracies over other methods.

6 Discussion

Our study addresses the challenge of recognizing human actions in drone-recorded RGB videos, crucial for various applications like video surveillance and sports analysis. We propose a multi-step system: segmenting video frames, preprocessing for quality enhancement, and identifying human bodies using the YOLO algorithm. Key skeletal points are extracted from human silhouettes, including head, shoulders, and joints. This data is optimized using the KDA optimizer and classified using a CNN. Evaluation on benchmark datasets shows promising action recognition accuracies, highlighting the effectiveness of our approach in overcoming complexities in drone-captured RGB videos.

7 Conclusion

The technique proposed in this study brings into the framework a new approach for detecting human actions in drone videos making it easier to identify people's movements and actions. Subsequently and most importantly, the system recognizes human poses and categorizes them with a fair degree of accuracy based on the features selected, thus enabling the users to understand the different movements of the human form in various activities. The integration of Convolutional Neural Networks (CNN) enables our system to focus and identify regional features and variations which enhances its capability of detecting motion dissimilarities in human movement. This is especially important for the type of applications where action recognition is crucial due to the possibility of better interpretation of the performed gestures and interactions in the context of the environment. Besides, as a part of the preprocessing steps which are also integrated into the proposed approach, the quality of the images is enhanced, and the interference from the background is minimized. When making foreground subjects stand out, we boost recognition rates and simplify detection models freed of interferences that disrupt recognition in real-world conditions.

Further, in the future, we plan on incorporating more features and testing our system with more different types of datasets. Expanding the number of scenarios and actions that we teach to our model is our goal to make more flexible the system that we develop in various conditions of operation. This is a continuous work due to our commitment to improve and enhance the method for the detection of the human action in the drone video for better performances in real-world scenarios.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://www.kaggle.com/datasets/

dasmehdixtr/drone-dataset-uav and https://paperswithcode.com/dataset/drone-action.

Author contributions

YA: Methodology, Writing – original draft. NM: Formal analysis, Writing – review & editing. BA: Investigation, Writing – review & editing. TS: Resources, Writing – review & editing. AA: Conceptualization, Writing – review & editing. HR: Project administration, Writing – review & editing. AJ: Supervision, Writing – original draft.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This research was supported by the Deanship of Scientific Research at Najran University, under the Research Group Funding program grant code (NU/PG/SERC/13/30). The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-FFR-2024-231-08". Princess Nourah Bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R440), Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Azmat, U., Alotaibi, S. S., Abdelhaq, M., Alsufyani, N., Shorfuzzaman, M., Jalal, A., et al. (2023). Aerial insights: deep learning-based human action recognition in drone imagery. *IEEE Access* 11, 83946–83961. doi: 10.1109/ACCESS.2023.3302353

Chen, H., Wang, J., Li, J., Qiu, Y., and Zhang, D. (2023). "Small object detection for drone image based on advanced YOLOv7," in Proceedings of the 42nd Chinese control conference (CCC), Tianjin, China, pp. 7453–7458.

Chéron, G., Laptev, I., and Schmid, C. (2015). P-CNN: pose-based CNN features for action recognition. In proceedings of the IEEE international conference on computer vision (ICCV), Dec., pp. 3218–3226.

Guo, L., Cheng, S., Liu, J., Wang, Y., Cai, Y., and Hong, X. (2022). Does social perception data express the spatio-temporal pattern of perceived urban noise? A case study based on 3,137 noise complaints in Fuzhou, China. *Appl. Acoust.* 201:109129. doi: 10.1016/j.apacoust.2022.109129

Hwang, H., Jang, C., Park, G., Cho, J., and Kim, I.-J. (2023). ElderSim: a synthetic data generation platform for human action recognition in eldercare applications. *IEEE Access* 11, 9279–9294. doi: 10.1109/ACCESS.2021.3051842

Jiang, Y., and He, X. (2020). Overview of applications of the sensor Technologies for Construction Machinery. *IEEE Access* 8, 110324–110335. doi: 10.1109/ACCESS.2020.3001968

Kozlov, O. V., Kondratenko, Y. P., and Skakodub, O. S. (2022). Information Technology for Parametric Optimization of fuzzy systems based on hybrid Grey wolf algorithms. *SN Comput. Sci.* 3:463. doi: 10.1007/s42979-022-01333-4

Kozlov, O., Kondratenko, Y., and Skakodub, O. (2024). Intelligent IoT-based control system of the UAV for meteorological measurements. *J. Mobile Multim.* 20, 555–596. doi: 10.13052/jmm1550-4646.2032

Nadeem, A., Jalal, A., and Kim, K. (2020). "Human actions tracking and recognition based on body parts detection via artificial neural network," in Proceedings of the 3rd international conference on advancements in computer science (ICACS), Lahore, Pakistan, pp. 1–6.

Papaioannidis, C., Makrygiannis, D., Mademlis, I., and Pitas, I. (2021). Learning fast and robust gesture recognition. In Proceedings of the 29th European signal processing conference (EUSIPCO), pp. 761–765.

Perera, A. G., Law, Y. W., and Chahl, J. (2019a). Drone-action: an outdoor recorded drone video dataset for action recognition. *Drones* 3:82. doi: 10.3390/drones3040082

Perera, A., Law, Y., and Chahl, J. (2019b). UAV-GESTURE: A dataset for UAV control and gesture recognition. In proceedings of the European conference on computer vision (ECCV). Cham, Switzerland: Springer, p. 11130.

Reddy, N. S., Saketh, M. S., and Dhar, S. (2016). Review of sensor Technology for Mine Safety Monitoring Systems: a holistic approach. In proceedings of the 2016 IEEE first international conference on control, measurement and instrumentation (CMI), Kolkata, India, 429–434.

Sanjay Kumar, Y. R., Raja Venkatesan, P., Ramanathan, M., and Saranya, S. (2024). "Smart surveillance with face recognition and object detection using drones," in Proceedings of the 10th international conference on communication and signal processing (ICCSP), Melmaruvathur, India, pp. 683–686.

Sidenko, I., Trukhov, A., Kondratenko, G., Zhukov, Y., and Kondratenko, Y. (2023). Machine learning for unmanned aerial vehicle routing on rough terrain. *Lecture Notes Data Eng. Commun. Technol.* 181, 626–635. doi: 10.1007/978-3-031-36118-0_56

Skakodub, O., Kozlov, O., and Kondratenko, Y. (2021). Optimization of Linguistic Terms' Shapes and Parameters: Fuzzy Control System of a Quadrotor Drone. 566–571. doi: 10.1109/IDAACS53288.2021.9660926

Shi, Y., Tian, Y., Wang, Y., and Huang, T. (2017). Sequential deep trajectory descriptor for action recognition with three-stream CNN. *Trans. Multi.* 19, 1510–1520. doi: 10.1109/TMM.2017.2666540

Sobhan, S., Islam, S., Valero, M., Shahriar, H., and Ahamed, S. I. (2021). Data analysis methods for health monitoring sensors: a survey. In Proceedings of the 2021 IEEE 45th annual computers, software, and applications conference (COMPSAC), Madrid, Spain, 669–676.

Sun, Y., Shen, Y., and Ma, L. (2021). MSST-RT: multi-stream spatial–temporal relative transformer for skeleton-based action recognition. *Sensors* 21:5339. doi: 10.3390/s21165339

Xu, L., Lan, C., Zeng, W., and Lu, C. (2022). Skeleton-based mutually assisted interacted object localization and human action recognition. *IEEE Trans. Multim.* doi: 10.1109/TMM.2022.3175374

Yadav, S. K., Luthra, A., Pahwa, E., Tiwari, K., Rathore, H., Pandey, H. M., et al. (2023). DroneAttention: sparse weighted temporal attention for drone-camera based activity recognition. *Neural Netw.* 159, 57–69. doi: 10.1016/j.neunet.2022.12.005

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 07 January 2025
DOI 10.3389/fnbot.2024.1460255

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Xu Zhang,
Boston Children's Hospital and Harvard Medical School, United States
Paolo Mercorelli,
Leuphana University Lüneburg, Germany

*CORRESPONDENCE
Zhengjun Zhai
✉ zhaizjun@nwpu.edu.cn

RECEIVED 05 July 2024
ACCEPTED 10 December 2024
PUBLISHED 07 January 2025

CITATION
Shen S, Chen J, Yu G, Zhai Z and Han P (2025) KalmanFormer: using transformer to model the Kalman Gain in Kalman Filters. *Front. Neurorobot.* 18:1460255. doi: 10.3389/fnbot.2024.1460255

KalmanFormer: using transformer to model the Kalman Gain in Kalman Filters

Siyuan Shen[1], Jichen Chen[2], Guanfeng Yu[3], Zhengjun Zhai[1]* and Pujie Han[4]

[1]School of Computer Science, Northwestern Polytechnical University, Xi'an, China, [2]Fourth Technical Department, Xi'an Microelectronics Technology Institute, Xi'an, China, [3]Research Office 16, AVIC Xi'an Aeronautics Computing Technique Research Institute, Xi'an, China, [4]Software Engineering College, Zhengzhou University of Light Industry, Zhengzhou, China

Introduction: Tracking the hidden states of dynamic systems is a fundamental task in signal processing. Recursive Kalman Filters (KF) are widely regarded as an efficient solution for linear and Gaussian systems, offering low computational complexity. However, real-world applications often involve non-linear dynamics, making it challenging for traditional Kalman Filters to achieve accurate state estimation. Additionally, the accurate modeling of system dynamics and noise in practical scenarios is often difficult. To address these limitations, we propose the KalmanFormer, a hybrid model-driven and data-driven state estimator. By leveraging data, the KalmanFormer promotes the performance of state estimation under non-linear conditions and partial information scenarios.

Methods: The proposed KalmanFormer integrates classical Kalman Filter with a Transformer framework. Specifically, it utilizes the Transformer to learn the Kalman Gain directly from data without requiring prior knowledge of noise parameters. The learned Kalman Gain is then incorporated into the standard Kalman Filter workflow, enabling the system to better handle non-linearities and model mismatches. The hybrid approach combines the strengths of data-driven learning and model-driven methodologies to achieve robust state estimation.

Results and discussion: To evaluate the effectiveness of KalmanFormer, we conducted numerical experiments in both synthetic and real-world dataset. The results demonstrate that KalmanFormer outperforms the classical Extended Kalman Filter (EKF) in the same settings. It achieves superior accuracy in tracking hidden states, demonstrating resilience to non-linearities and imprecise system models.

KEYWORDS

Kalman Filter, deep learning, transformer, Kalman Gain, supervised paradigm

1 Introduction

It is the most fundamental task to track the hidden state of a dynamical system by using the noisy measurements in real-time in many fileds, including singal processing (Yadav et al., 2023), navigation (Hu et al., 2003), information fusion (Xu et al., 2004), and automation control (Menner et al., 2023; Mercorelli, 2012a). A large number of algorithms were proposed to stress this issue, such as Bayesian estimation (Coué et al., 2003) and particle filter (Hue et al., 2002).

Kalman Filter (KF) (Kalman, 1960) is also an efficient recursive filter that can track the state of dynamic systems from a series of incomplete measurements with additive white Gaussian noise (AWGN). Low complexity implementation of KF, combined with theoretical foundation, resulted in it quickly becoming the popular method for state estimation problems.

The original Kalman Filters perform well in linear and Gaussian systems. The reality is that many nonlinear phenomena are encountered in real-world multi-sensor systems. Therefore, several variants of Kalman Filters are available to meet the requirements of

nonlinear dynamic systems, including Extended Kalman Filters (Maybeck, 1982) (EKF) and Unscented Kalman Filters (UKF) (Wan and Van Der Merwe, 2001).

There are still limitations associated with the application of EKF and UKF in practical applications. Specifically, the Kalman Filter is a model-based method, and the performance of state estimation heavily depends on model accuracy. Furthermore, the noise covariance matrix is determined by prior process noise and measurement noise, which are assumed to be Additive White Gaussian Noise (AWGN). Additionally, there is no guarantee that the AWGN will accurately reflect the actual performance of the information fusion.

Several variants of Kalman Filters were proposed to overcome the above issue. For example, Huang et al. (2020) introduced a sliding window variational adaptive Kalman filter to simultaneously modify the state estimation and covariance matrix. Yu and Li (2021) presented an adaptive Kalman Filter that concentrated on unknown covariances of both dynamic multiplicative noise and additive noises. Xiong et al. (2020) employed a parallel adaptive Kalman Filter to estimate the attitude of the vehicle based on the Inertial Measurement Unit (IMU). Paolo Mercorelli introduced (Mercorelli, 2012b) a combination of the augmented EKF and EKF for sensorless Valve Control which avoids complicated observation.

Recent years have seen the application of deep learning techniques to multiple real-world applications such as computer vision (Voulodimos et al., 2018) and natural language processing (Otter et al., 2020). Particularly, some Deep Neural Networks (DNNs), such as the Recurrent Neural Network (RNN) (Elman, 1990), Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997), Gated Recurrent Unit (GRU) (Chung et al., 2014), and Transformer (Vaswani et al., 2017), have demonstrated excellent performance when it comes to processing time series data. For example, Xia et al. (2021) presented staked GRU and RNN to predict the payload of electricity. Zhang et al. (2020) applied LSTM to estimate the battery's state of health.

Furthermore, deep learning techniques have been utilized by some researchers to enhance the effects of the Kalman Filters. For example, Rangapuram et al. (2018) introduced RNN to forecast the state space parameters of linear systems. Coskun et al. (2017) utilized LSTM to learn the noisy parameters and motion model of the Kalman filters. EKFNet (Xu and Niu, 2021) used BPTT (Ruder, 2016) to learn the process and measurement noise from the measurement. Bence Zsombor Hadlaczky applied neural networks and EKF to estimate the wing shape (Hadlaczky et al., 2023). Dahal et al. (2024) introduced RobustStateNet, which applied RNN and Kalman Filters to perform ego vehicle state estimation. Zhang et al. (2023) adopted the Transformer to pre-estimate the vehicle mass, thus acting as an observation for EKF. Luttmann and Mercorelli (2021) employed EKF to accelerate the convergence of the learning system.

In this work, we present KalmanFormer, a hybrid data-driven and model state estimator that can be used to perform information fusion in multi-sensor systems. Our KalmanFormer uses a Transformer framework to track the Kalman Gain instead of computing it from the statistic moments.

The structure of this paper is organized as follows: Section 2 introduces the Kalman Filters and Transformer architecture. Section 3 details the methodology of the proposed KalmanFormer. Experiments will be discussed in Section 4. Section 5 concludes the whole paper.

2 Preliminary knowledge

2.1 Kalman Filter

The Kalman Filter algorithm (KF) is a classic algorithm of information fusion technology and is widely used to solve various optimal estimation problems. The classic Kalman Filters are composed of a state transition model and an observation model, which are expressed as follows:

$$\begin{cases} x_k = \mathbf{F}_k x_{k-1} + \mathbf{B_k} u_{k-1} + w_{k-1} \\ z_k = \mathbf{H_k} x_k + v_k \\ w_{k-1} \sim \mathcal{N}(0, \mathbf{Q}_k) \\ v_k \sim N(0, \mathbf{R}_k) \end{cases} \tag{1}$$

where x_k is the state vector of the system, $\mathbf{F}_k$ represents the state transition matrix, $\mathbf{B}_k$ is the control-input model which is applied to the control vector u_{k-1}, and $\mathbf{H_k}$ represents the observation function, which maps the true state space into the observed space. w_{k-1} and v_k are process noise and observation noises respectively. Process noise is assumed to be drawn from a zero multivariate normal distribution $\mathcal{N}$ with covariance $\mathbf{Q_k}$. Observation noise is assumed to be zero mean Gaussian white noise with covariance $\mathbf{R_k}$.

In general, Recursive Kalman Filter can be divided into two steps: Prediction and Updation. The information flow of the Kalman Filter is shown in Figure 1. As shown in Figure 1, the predict step uses the state estimate from the previous timestep to produce an *priori* estimate of the state at the current timestep, which is expressed as follows:

$$\begin{aligned} \hat{x}_{k|k-1} &= \mathbf{F}_k \hat{x}_{k-1|k-1} + \mathbf{B} u_{k-1} \\ \mathbf{P}_{k|k-1} &= \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q_{k-1}} \end{aligned} \tag{2}$$

In the update phase, the innovation between the current a *priori* estimation and the current observation information, is multiplied by the optimal Kalman gain and combined with the previous state estimate to optimal the state estimate. This improved estimate based on the current observation is termed a *posteriori* state estimate, which is summarized as follows:

$$\begin{aligned} \mathbf{K}_k &= \frac{\mathbf{P}_{k|k-1}\mathbf{H}_k^T}{\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + \mathbf{K}_k (z_k - \mathbf{H}_k \hat{x}_{k|k-1}) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \end{aligned} \tag{3}$$

Noise has a significant impact on the performance of a Kalman Filter system, as it directly affects the accuracy of the estimation. A Kalman Filter is designed to optimally combine observations and predictions in the presence of noise, which controls how the Kalman filter weights the model predictions versus the actual observations.

The process noise covariance $\mathbf{Q}$ represents the uncertainty in the model of the system dynamics. Higher values of $\mathbf{Q}$ means we have less reliability on the prediction and place more trust in the observations.

The observation noise covariance $\mathbf{R}$ means the uncertainty in the observations. Higher values of $\mathbf{R}$ suggest more noise in

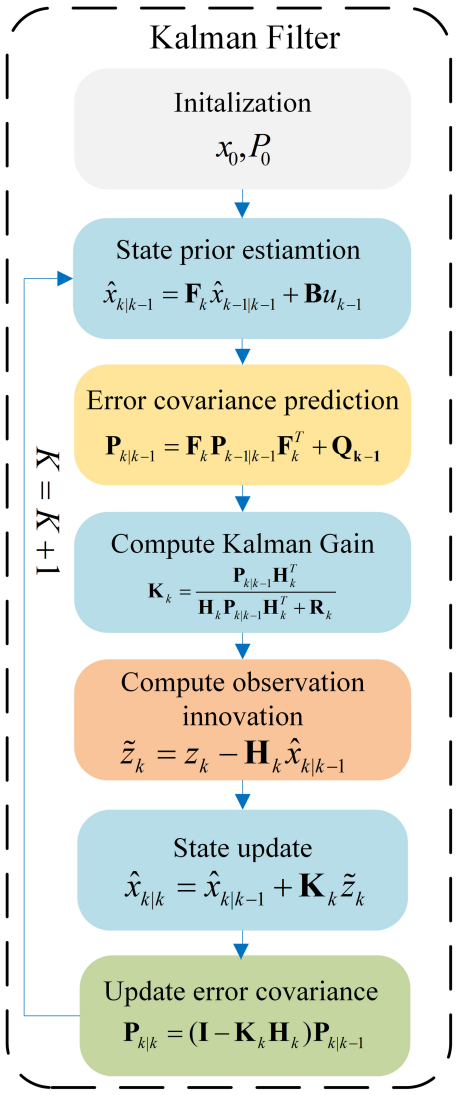


FIGURE 1
Information flow of the Kalman Filter.

the observation, so the Kalman Filter will pay more attention to its predictions.

The following tuning steps are necessary before using the Kalman Filters:

- Set initial state vector $\hat{x}_0$.
- Set initial noise values for **Q** and **R**.
- Tuning the Process Noise Covariance **Q**.
- Tuning the Observation Noise Covariance **R**.
- Test the performance and adjust **Q** and **R**.

Although the noise parameters are tuned before using the Kalman Filters. It is difficult to obtain an accurate state transition model and observation model, especially in the nonlinear occasion, which results in a significant degradation of Kalman Filter performance.

2.2 Extended Kalman Filters

Differentiable nonlinear functions may be used in place of the state transition and observation models in the extended Kalman Filter:

$$\begin{cases} \hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}) + w_{k-1} \\ z_k = h(x_k) + v_k \\ w_{k-1} \sim \mathcal{N}(0, \mathbf{Q}_k) \\ v_{k-1} \sim \mathcal{N}(0, \mathbf{R}_k) \end{cases} \tag{4}$$

Similar to the linear Kalman Filter, x_k is the state vector of the system, w_{k-1} and v_k are process noise and observation noises respectively. Process noise is assumed to be drawn from a zero multivariate normal distribution $\mathcal{N}$ with covariance $\mathbf{Q_k}$. Observation noise is assumed to be zero mean Gaussian white noise with covariance $\mathbf{R_k}$.

Function f is used to predict the state from the previous estimation and function h is applied to produce the predicted measurement form the predicted state. Different from the linear Kalman Filter, the Jacobian of f and h are used to compute the covariance matrix in extended Kalman Filters.

At timestamp k, the Jacobian is evaluated with the current predicted states, thus it can be used in Kalman equations. The prediction procedure of EKF is presented as follows:

$$\begin{aligned} \hat{x}_{k|k-1} &= f(\hat{x}_{k-1|k-1}, u_{k-1}) \\ \mathbf{P}_{k|k-1} &= \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q_{k-1}} \end{aligned} \tag{5}$$

The update procedure of EKF is calculated as follows:

$$\begin{aligned} \mathbf{K}_k &= \frac{\mathbf{P}_{k|k-1}\mathbf{H}_k^T}{\mathbf{H}_k \mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + \mathbf{K}_k(z_k - h(\hat{x}_{k|k-1})) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}_{k|k-1} \end{aligned} \tag{6}$$

where the state transition function and observation model are defined as the following Jacobians:

$$\begin{aligned} \mathbf{F}_k &= \frac{\partial f}{\partial x}\Big|_{\hat{x}_{k-1|k-1}} \\ \mathbf{H}_k &= \frac{\partial h}{\partial x}\Big|_{\hat{x}_{k|k-1}} \end{aligned} \tag{7}$$

2.3 Transformer

2.3.1 Transformer architecture

Transformer (Vaswani et al., 2017) was originally proposed in natural language processing and it has been applied in various sequence-to-sequence tasks. As shown in Figure 2, the Transformer is mainly composed of encoders and decoders with several basic transformer blocks. Transformer blocks inside the encoders and decoders remain in the same structure.

Encoders produce encodings for the input sequence, while the decoders take all the encodings from encoders and use contextual information to generate the prediction results. Each transformer block is composed of a multi-head attention layer, a feed-forward neural network, a skip connection connection, and a layer normalization operation.

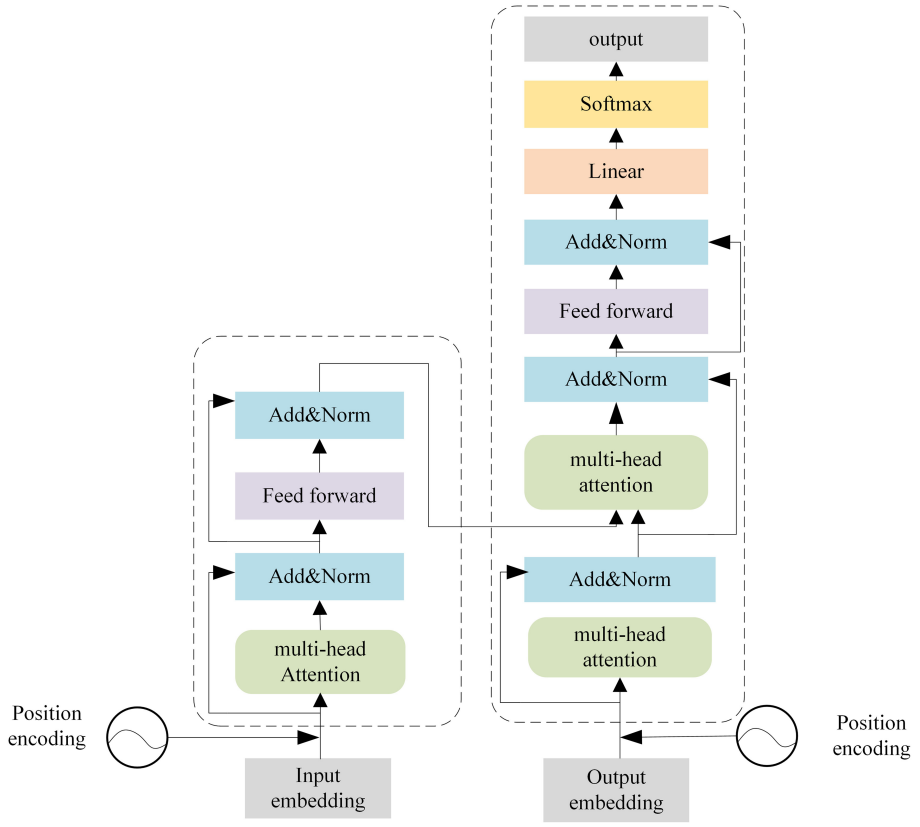


FIGURE 2
Architecture of the Transformer.

2.3.2 Self-attention mechanism

The Self-Attention Mechanism (SAM) is a core component of Transformer architecture, which seeks to emphasize the correlation between the input vector spaces.

As a first step, the input features are transformed into three different vectors using matrix multiplication, which is expressed as follows:

$$\begin{cases} \mathbf{Q} = F_{in}\mathbf{W}_Q \\ \mathbf{K} = F_{in}\mathbf{W}_K \\ \mathbf{V} = F_{in}\mathbf{W}_V \end{cases} \tag{8}$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ are Query matrix, Key matrix, and Value matrix respectively. $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ are used to generate the above-mentioned matrices. After that, attention map between different input vectors is calculated as follows:

- Compute scores between different input vectors with: $\mathbf{QK}^{\mathrm{T}}$.
- Normalize the scores to improve the stability with: $\frac{\mathbf{QK}^{\mathrm{T}}}{\sqrt{d_k}}$.
- Transform the scores into probabilities with softmax function: $\mathrm{softmax}(\frac{\mathbf{QK}^{T}}{\sqrt{d_k}})$.
- Generate the weighted value matrix with: $\mathrm{softmax}(\frac{\mathbf{QK}^{T}}{\sqrt{d_k}}) \cdot \mathbf{V}$.

The above process can be describe with a single function:

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}(\frac{\mathbf{QK}^{\mathrm{T}}}{\sqrt{d_k}})\mathbf{V} \tag{9}$$

where d_k means the dimension of the input. This procedure is shown in Figure 3.

2.3.3 Position encoding

Transformer architecture can't guarantee the order of objects inside the sequence. Therefore, positional encoding is employed to assign a unique representation to each position inside the sequence. Cosine and sine functions are used to produce position encoding for varying frequencies, which is calculated as follows:

$$\begin{aligned} P(k, 2i) &= \sin(\frac{k}{n^{2i/d}}) \\ P(k, 2i+1) &= \cos(\frac{k}{n^{2i/d}}) \end{aligned} \tag{10}$$

where k is the position of an object inside the sequence, d means dimensions of the output embedding space, $P(k, j)$ is position function, n is a predefine scalar, i is used to map column indices.

Using the position encoding, even positions correspond to a sine function and odd positions correspond to cosine functions.

3 Methodology

In this section, we present our KalmanFormer: a hybrid model and data-driven Kalman Filter for estimating the state of dynamic systems. Our KalmanFormer combines the model-based Kalman Filters with Transformer (Vaswani et al., 2017) to tackle model

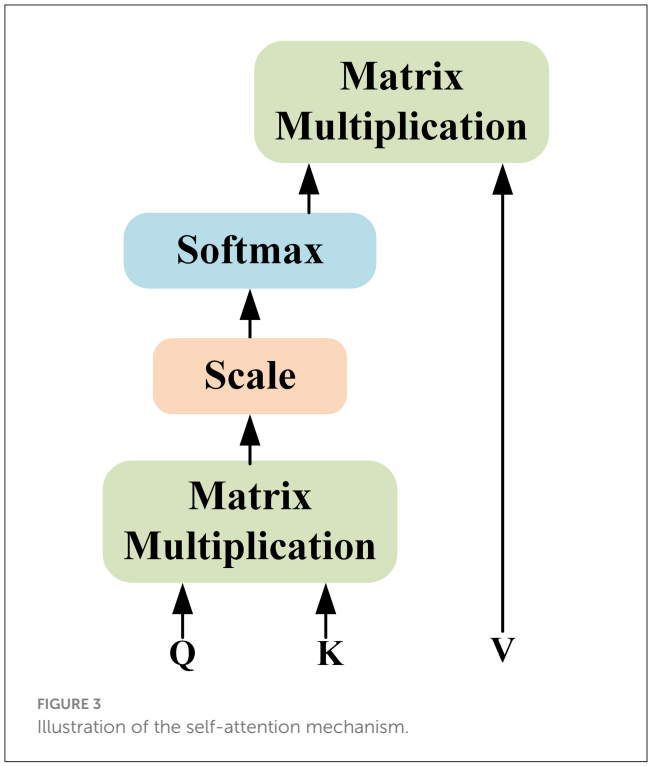


FIGURE 3
Illustration of the self-attention mechanism.

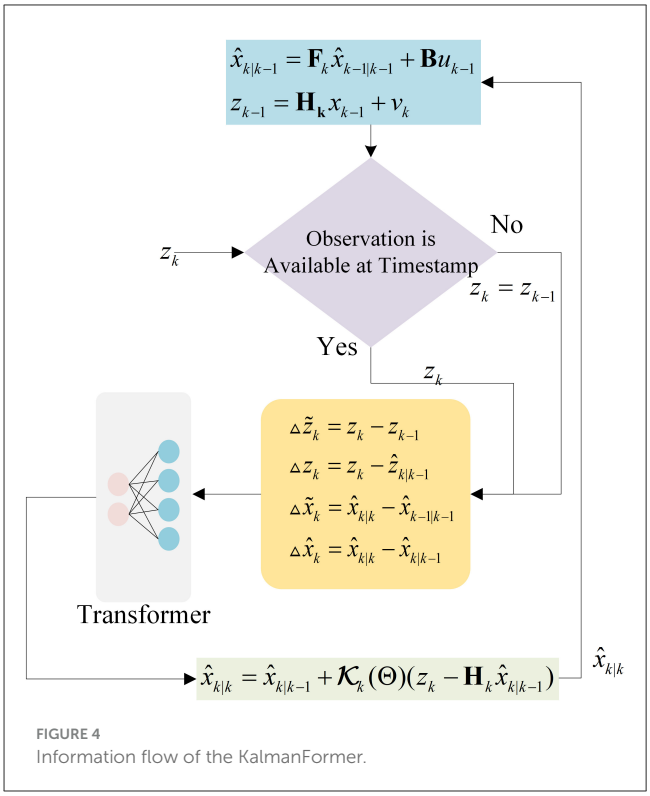


FIGURE 4
Information flow of the KalmanFormer.

mismatch and non-linearities. As a first step, the information flow of our KalmanFormer will be presented. Subsequently, details information about the inputs for our KalmanFormer will be discussed. Following that, the architecture of the KalmanFormer and the training strategy will be introduced at the end of this section.

3.1 Information flow of KalmanFormer

In order to formulate our KalmanFormer, we identify the specific computation process of linear Kalman Filters that are based on unavailable knowledge. To be specific, the state transition model $\mathbf{F_k}$ and observation model $\mathbf{H_k}$ are available (although inaccurate), while the process noise $\mathbf{Q}_k$ and observation noise $\mathbf{R}_k$ are unavailable. As shown in Figure 1, unknown process noise and observation noise are used in Kalman Filters only for the purpose of calculating the Kalman Gain. To this end, we develop the KalmanFormer that tracks the Kalman Gain from the data and combines the learned Kalman Gain into the data flow of the Kalman Filter. The architecture of our KalmanFormer is provided in Figure 4. In the same manner as the model-based Kalman Filters, our KalmanFormer outputs the state estimate through two procedures: Prediction and Update.

1. In the prediction procedure, a *prior* state estimate of the current moment $\hat{x}_{k|k-1}^{-}$ is obtained from the previous *posterior* estimate $\hat{x}_{k-1|k-1}$.
2. In the update procedure, KalmanFormer uses the new observation z_k to compute the current state *posterior* $\hat{x}_{k|k}$ from the previous prior estimation $\hat{x}_{k|k-1}$, which is calculated in Equation 11.

Instead of using the Kalman Gain matrix for the observation-update in the traditional Kalman Filters, KalmanFormer produces the Kalman Gain in a learned manner, denoted by $\mathcal{K}_{\mathcal{K}}(\Theta)$, with the trainable parameters Θ:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \mathcal{K}_{\mathcal{K}}(\Theta)(z_k - \mathbf{H}_k \hat{x}_{k|k-1}) \tag{11}$$

3.2 Input features

The model-based Kalman Filters, including EKF, UKF, and CKF compute the Kalman Gain from the known statistical information. We use a Transformer to model the Kalman Gain in a learned fashion in this paper. To calculate the Kalman Gain, we have to provide the information to a deep neural network to use the information to calculate the Kalman Gain. Inspired by KalmanNet (Revach et al., 2022), we devise the following quantities, which can be used for the input of the KalmanFormer:

- *The observation difference*:$\tilde{z}_k = z_k - z_{k-1}$
- *The innovation difference*: $z_k = z_k - \hat{z}_{k|k-1}$
- *The state evolution difference*: $\tilde{x}_k = \hat{x}_{k|k} - \hat{x}_{k-1|k-1}$, which represents the difference between two consecutive posterior state estimate.
- *The state update difference*: $\hat{x}_k = \hat{x}_{k|k} - \hat{x}_{k|k-1}$, which indicates the difference between the posterior state estimate and the prior state estimate.

Features $\tilde{x}_k$ and z_k indicate the uncertainty of the state estimates, while features z_k and $\hat{x}_k$ characterize the state transition and observation update process. Features z_k and $\tilde{z}_k$ contains the

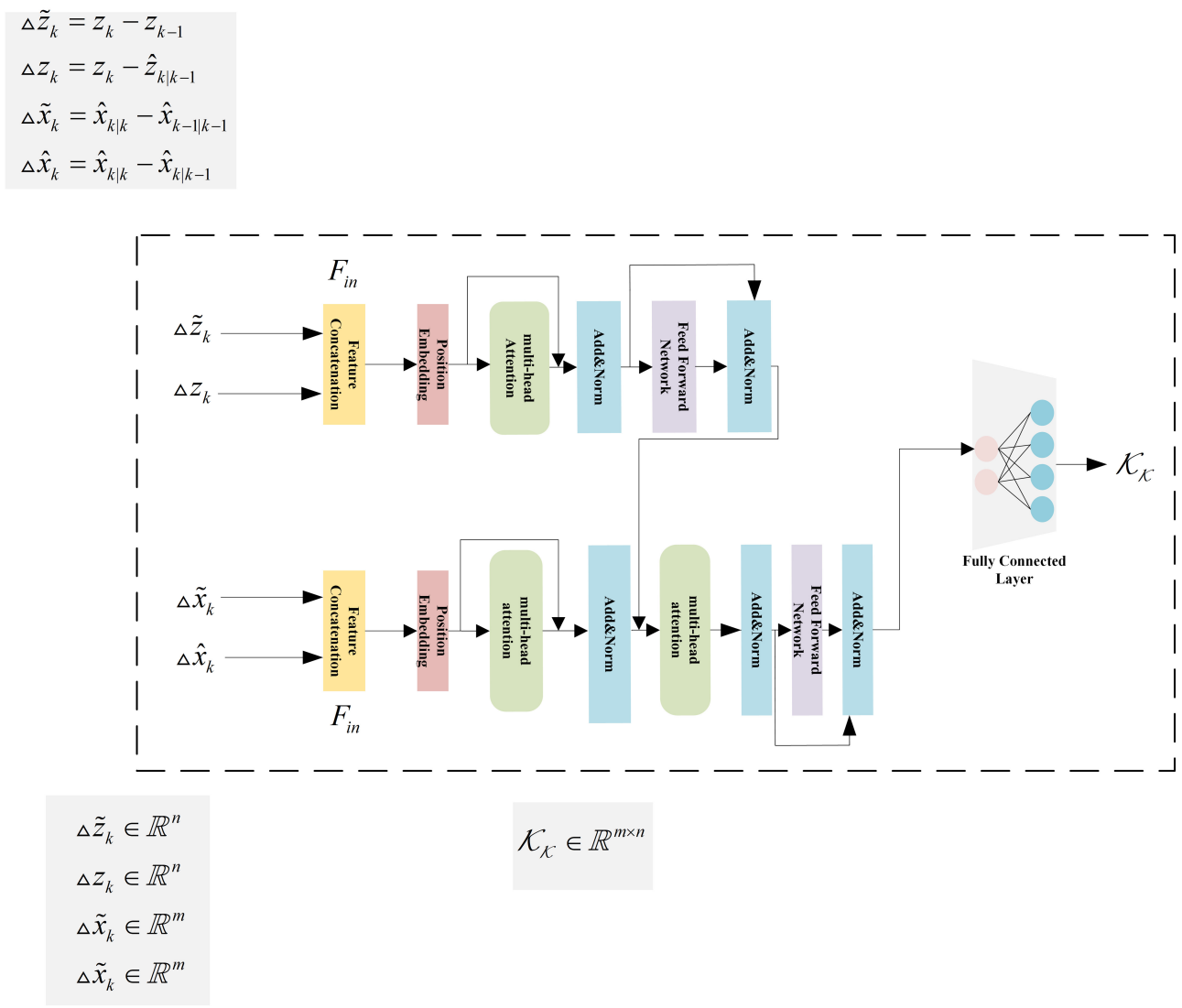


FIGURE 5
The details of the KalmanFormer.

observation information, while features x_k and $\hat{x}_k$ characterize the states information of the system.

3.3 Details of the KalmanFormer

The internal of KalmanFormer uses the features discussed in the previous section to compute the Kalman Gain. As a first step, we will introduce the input features of the Transformer. To be specific, $\tilde{z}_k$, z_k, $\tilde{x}_k$, and $\hat{x}_k$ are used to construct our KalmanFormer. The data flow of the input features inside our KalmanFormer is shown in Figure 5.

Subsequently, the related observation features $\Delta\tilde{z}_k \in \mathbb{R}^n$ and $z_k \in \mathbb{R}^n$ are concatenate together to the input $F_{in} \in \mathbb{R}^{2n}$ of the Transformer encoders. And also, the related state features $\Delta\tilde{x}_k \in \mathbb{R}^m$ and $\Delta\hat{x}_k \in \mathbb{R}^m$ are concatenated together to the input for the Transformer decoders.

We devise three initial matrices $\mathbf{W_Q} \in \mathbb{R}^{2m\times 2m}$, $\mathbf{W_K} \in \mathbb{R}^{2m\times 2m}$, and $\mathbf{W_V} \in \mathbb{R}^{2m\times 2m}$ to generate the **Q**, **K**, and **V** matrices, which is used to produce self-attention score described in Section 2.3.2.

Following is the Add and Norm operation. To be specific, Layer normalization is used to perform Add and Norm operation, which is expressed as follows:

$$LayerNorm(X + attention) \tag{12}$$

Then the feed forward neural network is used to generate output, which is presented as:

$$FFN = \mathrm{ReLU}(\mathrm{X}\mathrm{W}_1 + b_1)\mathrm{W}_2 + b_2 \tag{13}$$

The feed-forward neural network is composed of two layers of the fully connected network. The W_1 and W_2 are the weights for the two layers of network. b_1 and b_2 are the bias. ReLU is the Rectified Linear Unit activate function.

Then the output of the Transformer encoder and the concatenated state input features to produce the learned Kalman Gain.

In our implementation, the input dimension is set to 4, the feed-forward dimension is set to 64, and 2 heads are employed in the Multi-head Self Attention Mechanism (MHSA). Furthermore, we stack the encoder and decoder 2 times to produce the learned Kalman Gain.

The information flow of the KalmanFormer is illustrated in Figure 5.

3.4 Training algorithm

A supervised learning paradigm is used to train the KalmanFormer using the available labeled data. Instead of producing the *posterior* estimate state, our KalmanFormer produces the Kalman Gain. Consequently, we define (Equation 21) to backpropagate the loss of Kalman Gain to train our KalmanFormer:

$$\frac{\partial L}{\partial K_k} = \frac{\partial ||K_k \Delta z_k - \Delta x_k||^2}{\partial K_k} = 2 \cdot (K_k \Delta z_k - \Delta x_k) \cdot \Delta z_k^T \quad (14)$$

where $\Delta x_k = x_k - \hat{x}_{k|k-1}$. The Equation 14 indicates that we can learn the computation of the Kalman Gain by training KalmanFormer end-to-end using the squared-error loss.

In general, the dataset used for training the KalmanFormer consists of N length Ttrajectories. Let T denote the length of i-th training trajectory inside the dataset. The dataset can be expressed by $\mathcal{D} = \{(\mathbf{Z_i}, \mathbf{X_i})\}_1^N$, where

$$Z_i = [z_1^{(i)}, z_2^{(i)} ..., z_T^{(i)}], X_i = [x_0^{(i)}, x_1^{(i)} ..., x_T^{(i)}] \quad (15)$$

The empirical loss function for the i-th trajectory training inside the dataset is defined as follows:

$$l_i(\Theta) = \frac{1}{T_i} \sum_{k=1}^{T_i} ||\Psi_\Theta(\hat{x}_{k-1}^i, z_k^{(i)}) - x_k^{(i)}||^2 + \xi \cdot ||\Theta||^2 \quad (16)$$

where Ψ_Θ represents the output of our KalmanFormer, Θ is the trainable parameters inside the KalmanFormer, and ξ is regularization coefficient. Let $\Delta x_k^{(k)} = x_k^{(k)} - \hat{x}_{k|k-1}^{(k)}$ and $\Delta z_k^{(k)} = z_k^{(k)} - \hat{z}_{k|k-1}^{(k)}$ be the state prediction error and the measurement innovation at timestamp k. The partial derivative of the loss function respective to the Kalman gain matrix is:

$$\frac{\partial l(\Theta)}{\partial K_k(\Theta)} = \frac{1}{L T_k} \sum_{k=1}^{L} \sum_{k=1}^{T_k} \frac{\partial ||\Delta x_k^{(k)} - K_k(\Theta) \Delta z_k^{(k)}||_2^2}{\partial K_k(\Theta)} \quad (17)$$

By plugging into the chain rule:

$$\frac{\partial l(\Theta)}{\partial(\Theta)} = \frac{\partial l(\Theta)}{\partial K_k(\Theta)} \frac{\partial K_k(\Theta)}{\partial(\Theta)} \quad (18)$$

We can adopt a stochastic gradient descent algorithm to optimize Θ by using $\frac{\partial l(\Theta^*)}{\partial(\Theta^*)} = 0$.

4 Numerical experiments

In this section, we design a series of experiments to evaluate the performance of our proposed KalmanFormer and compare it to some other benchmarks. As a first step, we make a brief description of the training setup of our KalmanFormer. Following that, we conduct the simulation experiments including nonlinear cases to evaluate the performance of our proposed method. At the end of this section, IMU and GPS information are employed to investigate the effectiveness of our proposed method.

4.1 Implement details

To be specific, the dimensions of concatenated observation difference and innovation difference are 8, which is the input to the encoder for the transformer. Also, the dimensions of state evolution difference and state update difference are 4, which is the input to the decoders of Transformer. The feed-forward dimension inside the encoder and decoder is 64, and 2 heads are employed in the multi-head attention mechanism. Furthermore, we stack the encoder and decoder 2 times to produce the output. After the output is obtained, a fully connected layer is used to generate the learned Kalman Gain.

Furthermore, we conduct all of our training and validation experiments on the Pytorch (Paszke et al., 2019) platform using a single RTX 3090 GPU card, CUDA11.6, and cuDNN version 8. Furthermore, the Cosine Annealing Schedule is employed to adjust the learning rate in the training procedure, which can be expressed as follows:

$$\eta_t = \eta_{\min} + \frac{1}{2}(\eta_{max} - \eta_{min})(1 + \cos\left(\frac{T_{cur}}{T_{max}}\pi\right)) \quad (19)$$

where η_t represents the learning rate of the current iteration, $\eta_{\min}$ and $\eta_{\max}$ mean the predefined minimum and maximum learning rate respectively. T_{cur} and T_{max} are the current iteration and maximum iterations respectively.

Adam (Kingma and Ba, 2014) optimizer is used to train the KalmanFormer. Different hyper parameters are performed on the simulation and multi-sensor fusion experiments. The specific information about the hyperparameters is shown in Table 1.

4.2 Simulation experiments

In this section, a series of simulation experiments are designed to demonstrate the effort of our proposed KalmanFormer. We make a comparison with EKF and KalmanNet (Revach et al., 2021).

4.2.1 Test metric

Mean Square Error (MSE) is used to evaluate the effect of our proposed KalmanFormer, which is computed as follows:

$$MSE = \frac{1}{N} \sum_{j=1}^{N} \sum_{i=1}^{T} |(x_{est} - x_{true})_i|^2 \quad (20)$$

TABLE 1 Details information about the hyperparameters.

Experiment type	Epochs	Batch size	Learning rate	Weight decay
Simulation	200	30	1e-3	1e-3
Multi-sensor fusion	100	10	1e-3	1e-4

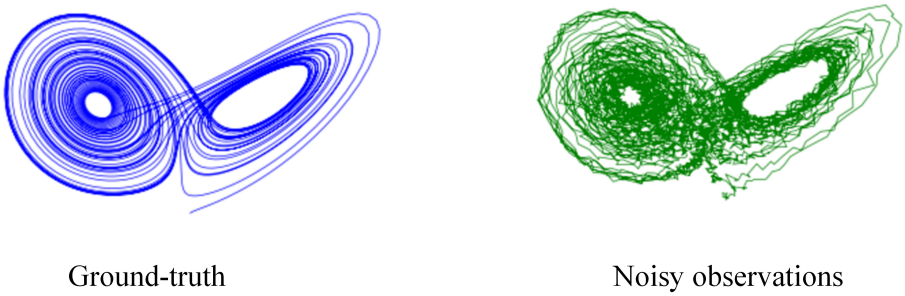


FIGURE 6
Illustration of trajectories.

where x_{est} means the output from our KalmanFormer, x_{true} represents corresponding ground-truth. N means the number of the testing trajectories. T is the length of current trajectory.

4.2.2 Non-linear Lorenz attractors

The Lorenz attractor (Tucker, 1999) describes a non-linear chaotic system used for atmospheric convection. The Lorenz system is expressed by following three differential equations that define the convection rate, the horizontal temperature variation, and the vertical temperature variation of a fluid:

$$\frac{\partial z_1}{\partial t} = 10(z_2 - z_1),\ \frac{\partial z_2}{\partial t} = z_1(28 - z_3) - z_2,\ \frac{\partial z_3}{\partial t} = z_1 z_2 - \frac{8}{3} z_3, \quad (21)$$

In order to generate the simulated trajectories, we run the Lorenz equations described at Equation 21 with a time step of $\Delta t = 0.05$ and add Gaussian noise of standard deviation $\sigma = 0.05$ to the results. The noisy data is considered as the measurements while the decimated data is regarded as the ground truth trajectory for our experiments. The trajectories of ground truth and noisy observations are shown in Figure 6.

Assuming a three-dimensional vector $x = [z_1, z_2, z_3]^{\mathrm{T}} \in \mathbf{R}$, the dynamic matrix $\mathbf{A}(x)$ of the system from Equation 21 is expressed as follows:

$$\mathbf{A}(x) = \begin{bmatrix} -10 & 10 & 0 \\ 28 - z_3 & -1 & 0 \\ z_2 & 0 & -\frac{8}{3} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \quad (22)$$

After that, Taylor expansion is used to obtain the state transition function:

$$\mathbf{F_k}(x_k) = \mathbf{I} + \sum_{j=1}^{J} \frac{(A(x_k)k)^j}{j!} \quad (23)$$

where $\mathbf{I}$ represents the identity matrix and $\mathbf{J}$ means the number of Taylor expansion. We set J = 5 in our experiments. For the measurement model, we set $\mathbf{H} = \mathbf{I}$. For the noise parameters, we set $\mathbf{Q} = \mathrm{q}^2\mathbf{I}$, $\mathbf{R} = r^2\mathbf{I}$, where q=0.8, r=1.

TABLE 2 Origin point information for NED frame.

Latitude origin	42.29322deg
Longitude origin	−83.709657 deg
Altitude origin	270 m

4.3 Multi-sensor information fusion

We further evaluate the effectiveness of the proposed KalmanFormer in multi-sensor fusion. We employ the Michigan NCLT dataset (Carlevaris-Bianco et al., 2016) with different types of sensors to perform our experiments.

The NCLT dataset was obtained from a mobile robot platform equipped with various sensors, including Real Time Kinematic GPS, IMU, Consumer-grade GPS, etc. In our experiments, IMU is employed to provide angular speed information and acceleration information, which is used to design the state transition function. The consumer-grade GPS is applied to provide the observation of the displacement. The Real-Time Kinematic GPS is used to generate a more accurate state of the system, which is used to evaluate the effectiveness of the proposed method.

4.3.1 Coordinates definition

A North-East-Down (NED) frame is employed to describe the robot's pose and position. Furthermore, the fixed origin point of the NED frame is shown in Table 2.

The angular and acceleration information from the IMU is measured in the IMU's reference frame, which closely aligns with the robot's reference coordinate. It is necessary to transform the IMU reading from IMU's frame into a global frame.

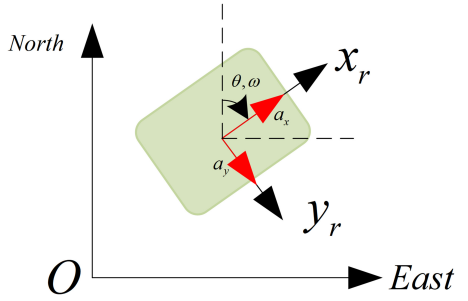


FIGURE 7
Transformation between IMU frame and Global Frame (NED).

As shown in Figure 7, we can obtain the transformation between the IMU frame and the global frame, which is calculated as:

$$\begin{cases} a_{gx} = a_x \cos(-\theta) - a_y \sin(-\theta) \\ a_{gy} = a_x \sin(-\theta) - a_y + \cos(-\theta) \end{cases} \tag{24}$$

4.3.2 State transition model

The state vector of the system in the global coordinate is defined as:

$$x_k = [x, y, v_x, v_y, \theta, \omega] \tag{25}$$

where x_k, y_k represent the position of the robot in the global frame. v_k and v_y represent the velocities. θ and ω mean the heading angle and angular velocities respectively.

In the global coordinate system, the state transition model takes the IMU's readings, including heading θ, angular velocity ω, and the accelerations as the control input. The state transition model (in the NED frame) is then:

$$\hat{x}_{k|k-1} = \mathbf{F_k}(x_{k-1}, u_{k-1}) = \begin{bmatrix} x_{k-1} + v_x \Delta k + \frac{1}{2} a_{gx} \Delta k^2 \\ y_{k-1} + v_y \Delta k + \frac{1}{2} a_{gy} \Delta k^2 \\ v_{x-1} + a_{gx} \Delta k \\ v_{y-1} + a_{gy} \Delta k \\ \theta_k \\ \omega_K \end{bmatrix} \tag{26}$$

4.3.3 Observation model

The GPS observation model produces a prediction of the expected GPS observation based on the predicted state. Here we use the consumer-grade GPS to produce the observation of the displacement. The observation model is expressed as follows:

$$z_{k|k-1} = \mathbf{H_k} \hat{x}_{k|k-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \hat{x}_{k|k-1} \tag{27}$$

4.3.4 Noise setting

The initial process noise $\mathbf{Q}_k$ and measurement noise $\mathbf{R}_k$ matrices of the EKF are expressed in Equations 2, 3. These $\mathbf{Q}_k$ and $\mathbf{R}_k$ matrices are determined using empirical data as well as completing experimental tuning. The initial $\mathbf{Q}_k$ and $\mathbf{R}_k$ matrices used in our experiments are developed as follows:

$$\mathbf{Q}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix} \tag{28}$$

$$\mathbf{R}_k = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix} \tag{29}$$

4.4 Model mismatch

4.4.1 State transition model mismatch

We devise experiments to investigate the robustness of the KalmanFormer when the state transition model is mismatched. This is achieved by using three 3-dimensional rotation matrices:

$$RZ = \begin{bmatrix} \cos(yaw) & -\sin(yaw) & 0 \\ \sin(yaw) & \cos(yaw) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{30}$$

$$RY = \begin{bmatrix} \cos(pitch) & 0 & \sin(pitch) \\ 0 & 1 & 0 \\ -\sin(pitch) & 0 & \cos(pitch) \end{bmatrix} \tag{31}$$

$$RX = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(roll) & -\sin(roll) \\ 0 & \sin(roll) & \cos(roll) \end{bmatrix} \tag{32}$$

$$yaw = roll = pitch = 1^\circ, 5^\circ \tag{33}$$

We evaluate the performance in the condition of model mismatch real-word NCLT datasets. The mismatched state transition real-world is expressed as follows:

$$\mathbf{F}_{real}^{\text{rotated}} = RX \bullet RY \bullet RZ \bullet \begin{bmatrix} 1 & \Delta k & \frac{1}{2}\Delta k^2 \\ 0 & 1 & \Delta k \\ 0 & 0 & 1 \end{bmatrix} \tag{34}$$

where $\mathbf{F}_{real}^{\text{rotated}}$ is the mismatched state transition model for the real-world experiments. In our experiments, the rotation angle is set to 1° and 5° to verify the model performance.

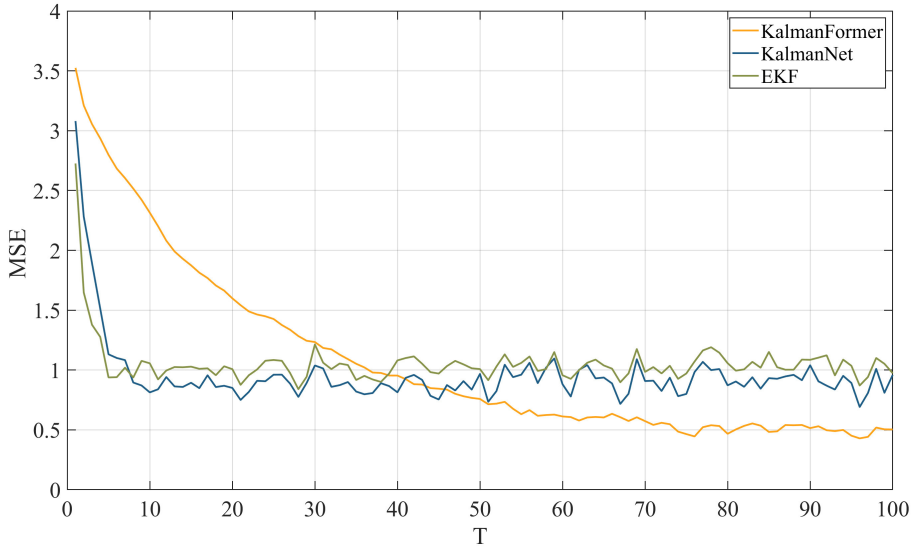


FIGURE 8
MSE comparison results with KalmanNet and EKF on synthetic dataset.

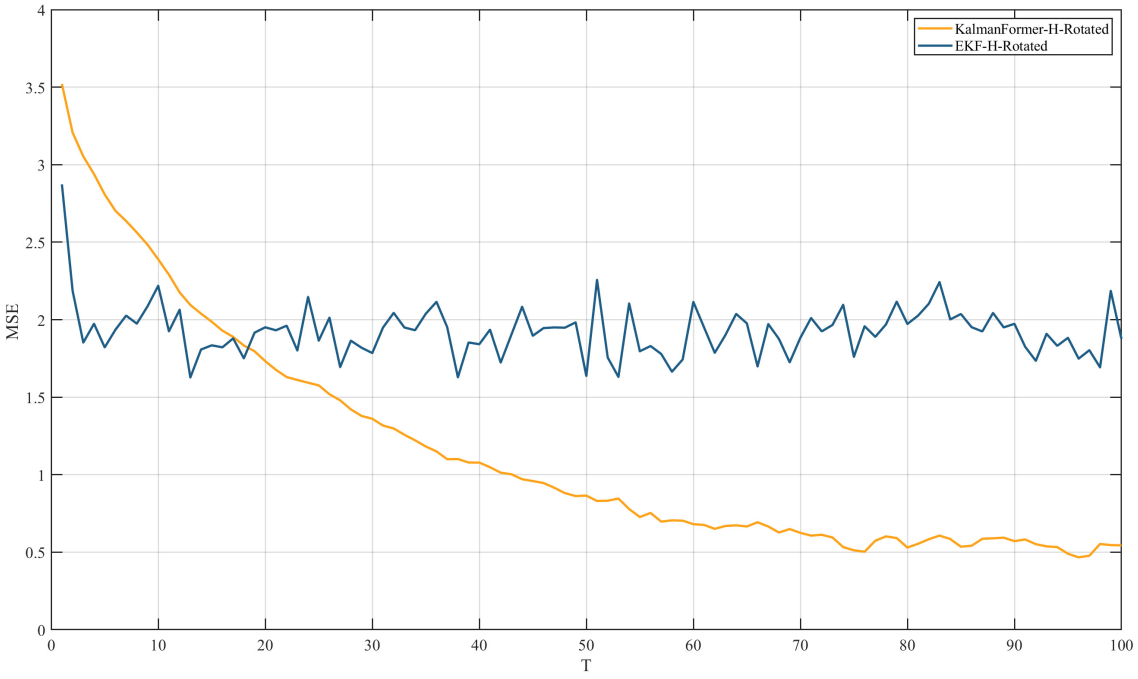


FIGURE 9
MSE comparison results with EKF when the observation is mismatched.

4.4.2 Observation model mismatch

Additionally, we investigate the performance of our proposed KalmanFormer with EKF when the observation function is mismatched. The mismatched observation function is expressed as follows:

$$\mathbf{H}_{rotated} = \mathbf{H} \bullet RX \bullet RY \bullet RZ \tag{35}$$

We set the rotation angle to 10° to validate the effectiveness on the simulation dataset.

Furthermore, we transform the observation in Cartesian coordinates into Spherical coordinates using the equation and compare the performance.

$$\begin{cases} r = \sqrt{z_1^2 + z_2^2 + z_3^2} \\ \theta = \cos^{-1}(\frac{z_3}{r}) \\ \phi = \tan^{-1}(\frac{z_2}{z_1}) \end{cases} \tag{36}$$

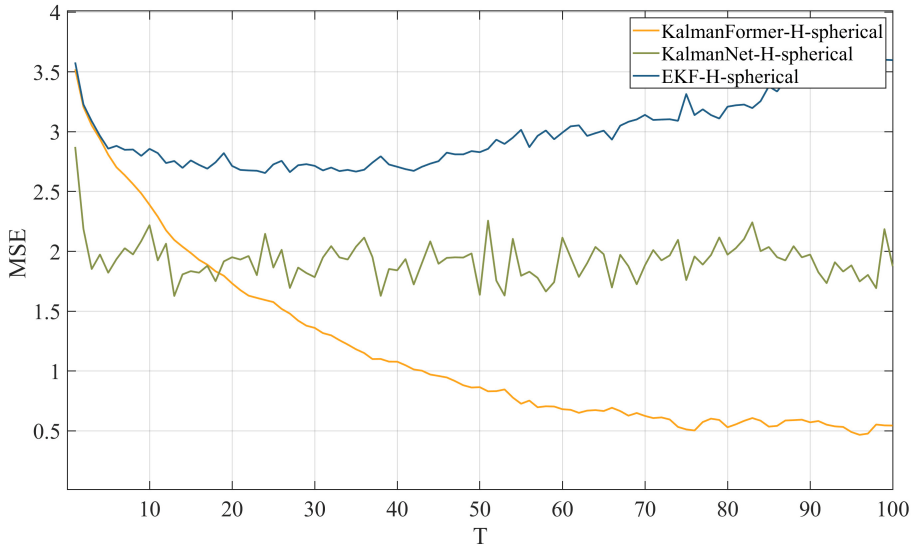


FIGURE 10
MSE comparison results with EKF when the observation is in spherical coordinate.

4.5 Evaluation results

In this section, we will discuss the performance of our proposed KalmanFormer with EKF and KalmanNet for both linear and non-linear systems. Furthermore, we will investigate the performance of the proposed KalmanFormer using the NCLT dataset.

4.5.1 Simulation results

MSE metric is used to demonstrate the effectiveness of the proposed KalmanFormer in non-linear Lorenz attractors. As shown in Figure 8, our KalmanFormer achieves a higher MSE result in the first 30 points of the Test sequence when compared to KalmanNet and EKF. However, after the 40 points, the MSE of our KalmanFormer is much lower than EKF and KalmanNet.

Besides that, Euclidean Distance is used to evaluate the effectiveness of our methodology over the whole test trajectories. Euclidean Distance is expressed as follows:

$$\text{distance} = \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{T} \left(x_{est}^{j} - x_{true}^{j}\right)^{2}} \tag{37}$$

where $x_{true} \in [\, z_1 \;\; z_2 \;\; z_3]^T \in \mathbb{R}$ means the ground truth of the state vector. $x_{\text{est}} \in [\, z_1 \;\; z_2 \;\; z_3]^T \in \mathbb{R}$ represents the estimation from our KalmanFormer. N is the number of the trajectories. T is the length for each trajectory.

Using Equation 37, the distance of our proposed method is 136. While the distance of KalmanNet is 209, which demonstrates the superiority of our KalmanFormer. In conclusion, KalmanFormer achieves more accurate performance on the Simulation Test set.

Additionally, Figure 9 reports the experiment results when the observation model is mismatched.

TABLE 3 The complexity comparison results on simulation experiments.

Method	Parameters	Storage (KB)	Inference time (s)
KalmanFormer	8,081	66	21
KalmanNet	23,928	46	19
EKF	\	\	20

We can observe that the proposed KalmanFormer achieves lower MSE performance than EKF in the same experiment setup when the observation model is disturbed by the rotation matrix.

Figure 10 reports the results when the observation in transformed into spherical coordinates. We can see that our proposed KalmanFormer achieves the best performance compared to KalmanNet and EKF in the condition of the mismatched observation.

Finally, we compare the time complexity of the KalmanFormer compared to EKF and KalmanNet through simulation experiments. Parameters, storage space, and inference time are adopted to verify the computational complexity of the KalmanFormer, KalmanNet, and EKF. The inference time is computed on the simulation experiments. The comparison results are shown in Table 3.

As shown in Table 3, the KalmanNet and the KalmanFormer have similar space demand and the similar running speeds on the simulation experiments. To be specific, the KalmanNet needs 44 KB harddisk space to store while the KalmanFormer 66KB needs disk space. Furthermore, we compare the inference time on the whole dataset. The inference time of the EKF is about 20s while the KalmanFormer runs about 21s. We can conclude that the proposed KalmanFormer has a similar time complexity with the EKF and KalmanNet and it can be further used in real-world applications.

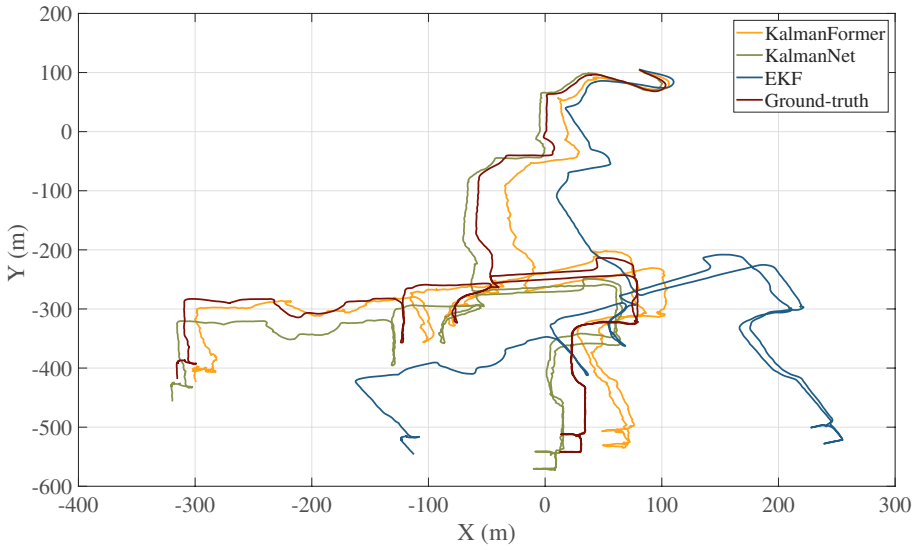


FIGURE 11
Trajectory comparison results with KalmanNet and EKF.

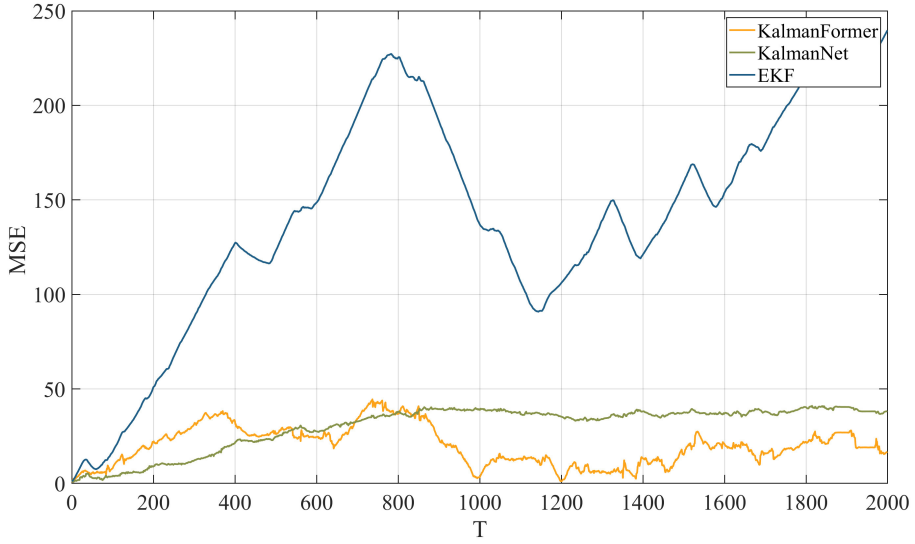


FIGURE 12
MSE performance with KalmanNet and EKF on NCLT dataset.

4.5.2 Multi-sensor information fusion

The trajectory we used for training and validating KalmanFormer and KalmanNet are obtained from the date of 2012-01-22 within the NCLT datasets. Furthermore, a date of 2012-04-29 trajectory is used to test the performance. The sample rate of the training, validation, and test is 1 *HZ*. The trajectory comparison result is shown in Figure 11.

As shown in Figure 11, our KalmanFormer performs better than EKF and KalmanNet. In order to evaluate the property of our KalmanFormer, we make a comparison with EKF and KalmanNet in terms of MSE using the same data in Figure 11. The result is shown in Figure 12. According to Figure 12, our KalmanFormer achieves similar accuracy at the first 500 points of the testing set. However, in the last 1,500 points, our method achieves better performance in MSE compared to KalmanNet.

Additionally, Equation 37 is used to evaluate the validity over the whole test trajectory quantitatively. The distance of KalmanFormer is 19 m, the distance of KalmanNet is 30 m, and the distance of EKF is 316 m, which proves the superiority of the KalmanFormer.

Finally, we investigate the results using the mismatched state transition function with the rotation angles of 1° and 5°. Figure 13 reports the results when the state transition function is mismatched.

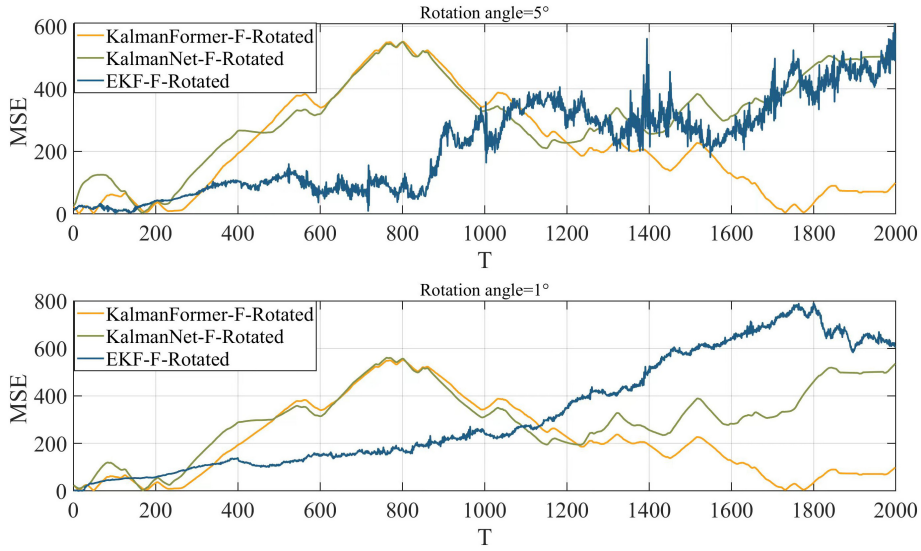


FIGURE 13
MSE performance when the state transition model is mismatched.

We can observe that when the state transition model is disturbed by the rotation matrix with a rotation angle of 1°, our KalmanFormer has a similar performance to the KalmanNet and outperforms the EKF. When the rotation angle is set to 5°, the performance of EKF degrades significantly. And the KalmanFormer outperforms the KalamNet and EKF. Even our KalmanFormer achieves lower MSE than KalmanNet.

5 Conclusion

In this paper, we proposed KalmanFormer, which is a hybrid of data-driven and model-driven implementation of the Kalman Filters. KalmanFormer incorporates a Transformer architecture within the learning process of computing the Kalman Gain (KG) and combines the learned KG into a traditional Kalman Filter. The proposed KalmanFormer uses the Kalman Filter without requiring any prior knowledge of process statistics or measurement noise statistics, even if the system model is mismatched. It has been demonstrated through numerical experiments that KalmanFormer is capable of achieving the minimum MSE when properly trained. It has also been proven that KalmanFormer is more robust to inaccurate knowledge of state space parameters in multi-sensor information fusion.

Data availability statement

The public NCLT dataset is used in this study. Researchers can get them from website of https://robots.engin.umich.edu/nclt/.

Author contributions

SS: Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing. JC: Conceptualization, Validation, Writing – review & editing. GY: Software, Writing – original draft. ZZ: Writing – original draft. PH: Data curation, Methodology, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Carlevaris-Bianco, N., Ushani, A. K., and Eustice, R. M. (2016). University of Michigan north campus long-term vision and Lidar dataset. *Int. J. Rob. Res.* 35, 1023–1035. doi: 10.1177/0278364915614638

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*. doi: 10.48550/arXiv.1412.3555

Coskun, H., Achilles, F., DiPietro, R., Navab, N., and Tombari, F. (2017). "Long short-term memory kalman filters: recurrent neural estimators for pose regularization,"9D in *IEEE International Conference on Computer Vision*, 5524–5532. doi: 10.1109/ICCV.2017.589

Coué, C., Fraichard, T., Bessiere, P., and Mazer, E. (2003). "Using bayesian programming for multi-sensor multi-target tracking in automotive applications,"9D in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)* (IEEE), 2104–2109. doi: 10.1109/ROBOT.2003.1241904

Dahal, P., Mentasti, S., Paparusso, L., Arrigoni, S., and Braghin, F. (2024). Robuststatenet: robust ego vehicle state estimation for autonomous driving. *Rob. Auton. Syst.* 172:104585. doi: 10.1016/j.robot.2023.104585

Elman, J. L. (1990). Finding structure in time. *Cogn. Sci.* 14, 179–211. doi: 10.1207/s15516709cog1402_1

Hadlaczky, B. Z., Friedman, N., Takarics, B., and Vanek, B. (2023). "Wing shape estimation with extended kalman filtering and kalmannet neural network of a flexible wing aircraft," in *Learning for Dynamics and Control Conference* (PMLR), 1429–1440.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9, 1735–1780. doi: 10.1162/neco.1997.9.8.1735

Hu, C., Chen, W., Chen, Y., Liu, D., et al. (2003). Adaptive kalman filtering for vehicle navigation. *J. Global Posit. Syst.* 2, 42–47. doi: 10.5081/jgps.2.1.42

Huang, Y., Zhu, F., Jia, G., and Zhang, Y. (2020). A slide window variational adaptive kalman filter. *IEEE Trans. Circ. Syst.* 67, 3552–3556. doi: 10.1109/TCSII.2020.2995714

Hue, C., Le Cadre, J.-P., and Pérez, P. (2002). Sequential monte carlo methods for multiple target tracking and data fusion. *IEEE Trans. Signal Proc.* 50, 309–325. doi: 10.1109/78.978386

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *J. Basic Eng.* 82, 35–45. doi: 10.1115/1.3662552

Kingma, D. P., and Ba, J. (2014). Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. doi: 10.48550/arXiv.1412.6980

Luttmann, L., and Mercorelli, P. (2021). "Comparison of backpropagation and kalman filter-based training for neural networks," in *2021 25th International Conference on System Theory, Control and Computing (ICSTCC)* (IEEE), 234–241. doi: 10.1109/ICSTCC52150.2021.9607274

Maybeck, P. S. (1982). *Stochastic Models, Estimation, and Control*. London: Academic Press.

Menner, M., Berntorp, K., and Di Cairano, S. (2023). Automated controller calibration by kalman filtering. *IEEE Trans. Control Syst. Technol.* 31, 2350–2364. doi: 10.1109/TCST.2023.3254213

Mercorelli, P. (2012a). "A kalman filter for sensorless control of a hybrid hydraulic piezo actuator using MPC for camless internal combustion engines," in *2012 IEEE International Conference on Control Applications* (IEEE), 980–985. doi: 10.1109/CCA.2012.6402717

Mercorelli, P. (2012b). A two-stage augmented extended kalman filter as an observer for sensorless valve control in camless internal combustion engines. *IEEE Trans. Industr. Electr.* 59, 4236–4247. doi: 10.1109/TIE.2012.2192892

Otter, D. W., Medina, J. R., and Kalita, J. K. (2020). A survey of the usages of deep learning for natural language processing. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 604–624. doi: 10.1109/TNNLS.2020.2979670

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). "Pytorch: an imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 32.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. (2018). "Deep state space models for time series forecasting," in *Advances in Neural Information Processing Systems*, 31.

Revach, G., Shlezinger, N., Ni, X., Escoriza, A. L., Van Sloun, R. J., and Eldar, Y. C. (2022). Kalmannet: Neural network aided kalman filtering for partially known dynamics. *IEEE Trans. Signal Proc.* 70, 1532–1547. doi: 10.1109/TSP.2022.3158588

Revach, G., Shlezinger, N., Van Sloun, R. J., and Eldar, Y. C. (2021). "Kalmannet: data-driven kalman filtering," in *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE), 3905–3909. doi: 10.1109/ICASSP39728.2021.9413750

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*. doi: 10.48550/arXiv.1609.04747

Tucker, W. (1999). The lorenz attractor exists. *Compt. Rendus l'Acad. Sci. Mathem.* 328, 1197–1202. doi: 10.1016/S0764-4442(99)80439-X

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). "Attention is all you need," in *Advances in Neural Information Processing Systems*, 30.

Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E., et al. (2018). Deep learning for computer vision: a brief review. *Comput. Intell. Neurosci.* 2018:7068349. doi: 10.1155/2018/7068349

Wan, E. A., and Van Der Merwe, R. (2001). "The unscented kalman filter," in *Kalman Filtering and Neural Networks*, 221–280. doi: 10.1002/0471221546.ch7

Xia, M., Shao, H., Ma, X., and de Silva, C. W. (2021). A stacked GRU-RNN-based approach for predicting renewable energy and electricity load for smart grid operation. *IEEE Trans. Industr. Inform.* 17, 7050–7059. doi: 10.1109/TII.2021.3056867

Xiong, L., Xia, X., Lu, Y., Liu, W., Gao, L., Song, S., et al. (2020). IMU-based automated vehicle body sideslip angle and attitude estimation aided by gnss using parallel adaptive kalman filters. *IEEE Trans. Vehic. Technol.* 69, 10668–10680. doi: 10.1109/TVT.2020.2983738

Xu, L., and Niu, R. (2021). "Ekfnet: learning system noise statistics from measurement data," in *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE), 4560–4564. doi: 10.1109/ICASSP39728.2021.9415083

Xu, L., Zhang, J. Q., and Yan, Y. (2004). A wavelet-based multisensor data fusion algorithm. *IEEE Trans. Instrum. Meas.* 53, 1539–1545. doi: 10.1109/TIM.2004.834066

Yadav, S., Saha, S. K., and Kar, R. (2023). An application of the kalman filter for EEG/ERP signal enhancement with the autoregressive realisation. *Biomed. Signal Process. Control* 86:105213. doi: 10.1016/j.bspc.2023.105213

Yu, X., and Li, J. (2021). Adaptive kalman filtering for recursive both additive noise and multiplicative noise. *IEEE Trans. Aerosp. Electron. Syst.* 58, 1634–1649. doi: 10.1109/TAES.2021.3117896

Zhang, H., Yang, Z., Xiong, H., Zhu, T., Long, Z., and Wu, W. (2023). Transformer aided adaptive extended kalman filter for autonomous vehicle mass estimation. *Processes* 11:887. doi: 10.3390/pr11030887

Zhang, W., Li, X., and Li, X. (2020). Deep learning-based prognostic approach for lithium-ion batteries with adaptive time-series prediction and on-line validation. *Measurement* 164:108052. doi: 10.1016/j.measurement.2020.108052

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 29 January 2025
DOI 10.3389/fnbot.2024.1490267

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Elishai Ezra Tsur,
Open University of Israel, Israel
Fabio Schittler Neves,
Fraunhofer Institute for Industrial Mathematics (ITWM), Germany

*CORRESPONDENCE
Ugur Akcal
✉ makcal2@illinois.edu
Ivan Georgiev Raikov
✉ iraikov@stanford.edu
Girish Chowdhary
✉ girishc@illinois.edu

RECEIVED 02 September 2024
ACCEPTED 26 December 2024
PUBLISHED 29 January 2025

CITATION
Akcal U, Raikov IG, Gribkova ED, Choudhuri A, Kim SH, Gazzola M, Gillette R, Soltesz I and Chowdhary G (2025) LoCS-Net: Localizing convolutional spiking neural network for fast visual place recognition.
Front. Neurorobot. 18:1490267.
doi: 10.3389/fnbot.2024.1490267

LoCS-Net: Localizing convolutional spiking neural network for fast visual place recognition

Ugur Akcal[1,2,3]*, Ivan Georgiev Raikov[4]*, Ekaterina Dmitrievna Gribkova[3,5], Anwesa Choudhuri[3,6], Seung Hyun Kim[7], Mattia Gazzola[7], Rhanor Gillette[5,8], Ivan Soltesz[4] and Girish Chowdhary[2,3,9]*

[1]The Grainger College of Engineering, Department of Aerospace Engineering, University of Illinois Urbana-Champaign, Urbana, IL, United States, [2]The Grainger College of Engineering, Siebel School of Computing and Data Science, University of Illinois Urbana-Champaign, Urbana, IL, United States, [3]Coordinated Science Laboratory, University of Illinois Urbana-Champaign, Urbana, IL, United States, [4]Department of Neurosurgery, Stanford University, Stanford, CA, United States, [5]Neuroscience Program, Center for Artificial Intelligence Innovation, University of Illinois Urbana-Champaign, Urbana, IL, United States, [6]The Grainger College of Engineering, Department of Electrical and Computer Engineering, University of Illinois Urbana-Champaign, Urbana, IL, United States, [7]The Grainger College of Engineering, Mechanical Science and Engineering, University of Illinois Urbana-Champaign, Urbana, IL, United States, [8]Department of Molecular and Integrative Physiology, University of Illinois Urbana-Champaign, Urbana, IL, United States, [9]The Grainger College of Engineering, College of Agriculture and Consumer Economics, Department of Agricultural and Biological Engineering, University of Illinois Urbana-Champaign, Urbana, IL, United States

Visual place recognition (VPR) is the ability to recognize locations in a physical environment based only on visual inputs. It is a challenging task due to perceptual aliasing, viewpoint and appearance variations and complexity of dynamic scenes. Despite promising demonstrations, many state-of-the-art (SOTA) VPR approaches based on artificial neural networks (ANNs) suffer from computational inefficiency. However, spiking neural networks (SNNs) implemented on neuromorphic hardware are reported to have remarkable potential for more efficient solutions computationally. Still, training SOTA SNNs for VPR is often intractable on large and diverse datasets, and they typically demonstrate poor real-time operation performance. To address these shortcomings, we developed an end-to-end convolutional SNN model for VPR that leverages backpropagation for tractable training. Rate-based approximations of leaky integrate-and-fire (LIF) neurons are employed during training, which are then replaced with spiking LIF neurons during inference. The proposed method significantly outperforms existing SOTA SNNs on challenging datasets like Nordland and Oxford RobotCar, achieving 78.6% precision at 100% recall on the Nordland dataset (compared to 73.0% from the current SOTA) and 45.7% on the Oxford RobotCar dataset (compared to 20.2% from the current SOTA). Our approach offers a simpler training pipeline while yielding significant improvements in both training and inference times compared to SOTA SNNs for VPR. Hardware-in-the-loop tests using Intel's neuromorphic USB form factor, Kapoho Bay, show that our on-chip spiking models for VPR trained via the ANN-to-SNN conversion strategy continue to outperform their SNN counterparts, despite a slight but noticeable decrease in performance when transitioning from off-chip to on-chip, while offering significant energy efficiency. The results highlight the outstanding rapid prototyping and real-world deployment capabilities of this approach, showing it to be a substantial step toward more prevalent SNN-based real-world robotics solutions.

KEYWORDS

spiking neural networks, robotics, visual place recognition, localization, supervised learning, convolutional networks

1 Introduction

Visual place recognition (VPR) refers to the capability of identifying locations within a physical environment solely through visual inputs. It is essential for autonomous navigation of mobile robots, indoor assistive navigation aid, augmented reality, and geolocalization (Lanham, 2018; Reinhardt, 2019; Weyand et al., 2016; Seo et al., 2018; Li et al., 2018; Shan et al., 2015). These applications generally involve complex dynamic scenes, perceptual aliasing, viewpoint and appearance variation, which render VPR extremely challenging.

VPR has been approached via deep learning techniques (Radenović et al., 2018; Chen et al., 2017; Sünderhauf et al., 2015) and through various supervised and self-supervised feature descriptor representations (DeTone et al., 2018; He et al., 2018; McManus et al., 2014). Despite their promise, many of these methods face significant practical challenges (Lynen et al., 2015, 2020). For example, they often rely on large, deep networks with time-consuming training processes and dense feature extraction, ultimately making them computationally expensive, memory-intensive, and energy-demanding. Such limitations significantly reduce the ability for real-world deployment of conventional artificial neural networks (ANNs) on robotic platforms with limited on-board resources (Doan et al., 2019). Spiking neural networks (SNNs) offer an alternative with their remarkable potential for computationally efficient operation when they are implemented on neuromorphic hardware (Davies et al., 2021). However, previous work on SNN models for VPR has suffered from scalability problems that impede their application to data with a large number of locations. In addition, the majority of the aforementioned methods formulate VPR as an image retrieval task (Garg et al., 2021), the solution of which aims for the correct association of given query images with a set of reference images. Such formulation requires the employment of a confusion matrix (a.k.a. distance matrix) (Garg et al., 2022) populated with similarity scores based on the distances between model-specific feature descriptors. A commonly-used similarity metric is the cosine similarity (Naseer et al., 2018), which is reported to be computationally expensive when evaluating high-dimensional feature vectors (Zhang et al., 2021).

These drawbacks have motivated our approach to VPR, described in this paper, in which an SNN model is implemented using an ANN-to-SNN conversion method to enable backpropagation-based training, resulting in fast training and inference times. We employ a smooth rate-based approximation (Hunsberger and Eliasmith, 2015) of the leaky integrate-and-fire (LIF) neurons (Burkitt, 2006) during the training. Once the training session is completed the rate-based units are substituted with the spiking LIF neurons and the resulting spiking network is used for inference.

We formulate VPR as a classification task, where the SNN model predicts place labels that uniquely correspond to the locations in a discretized navigation domain. We evaluate our method with the challenging real-world benchmark datasets Nordland (Olid et al., 2018) and Oxford RobotCar (Maddern et al., 2017, 2020). Our model, the Localizing Convolutional Spiking Neural Network (LoCS-Net), outperforms other SOTA SNN-based VPR methods on both the Nordland (Olid et al., 2018) and the Oxford RobotCar dataset (Maddern et al., 2017, 2020) in terms of precison at 100% recall (P@100%R).

The main contributions of this work are as follows. (a) To the best of our knowledge, LoCS-Net is the first SNN that is trained to perform the VPR task by means of ANN-to-SNN conversion and backpropagation. (b) LoCS-Net is an end-to-end SNN solution. Therefore, LoCS-Net does not require further processing of its outputs for recognizing places. In that sense, LoCS-Net saves all the computation resources that traditional VPR algorithms would typically expend on feature encoding, descriptor matching, computing similarity scores, and storing a distance matrix. (c) We demonstrate that our proposed SNN model yields the fastest training time, the second fastest inference time, and the best VPR performance in P@100%R among its SNN counterparts. This poses LoCS-Net as a significant step toward deployment of SNN-based VPR systems on robotics platforms for real-time localization. (d) We report the challenges we experienced when deploying LoCS-Net on the neuromorphic Loihi chips in detail. We strongly believe that our in-depth discussion on hardware deployment will be useful for the SNN-VPR community.

2 Related work

Task-specific feature descriptors are the very core of traditional VPR systems, which can be grouped into two categories: (1) Local descriptors, (2) Global descriptors. Local descriptors may scan the given images in patches of arbitrary size and stride. These patches are then compared to their immediate neighborhood to determine the distinguishing patterns (Loncomilla et al., 2016). In general, previous VPR work utilizing local descriptors (Johns and Yang, 2011; Kim et al., 2015; Zemene et al., 2018) employs sparse filters that extract so-called key-points (Mikolajczyk and Schmid, 2002; Matas et al., 2004). These key-points can be marked by the descriptions generated through the application of methods including SIFT (Lowe, 1999), RootSIFT (Arandjelović and Zisserman, 2012), SURF (Bay et al., 2006), and BRIEF (Calonder et al., 2011). In this way, the combination of heuristics-based detectors and local descriptors can be used for: (A) Representing images, (B) Comparing two images with respect to their descriptors to determine how similar they are. In addition, local features can be combined with other embeddings (Tsintotas et al., 2022) while leveraging their robustness against the variations in the robot's pose. However, local descriptors can be computationally heavier and more sensitive to illumination changes (Masone and Caputo, 2021). Global descriptors (Oliva and Torralba, 2006; Torralba et al., 2008), on the other hand, do not require a detection phase and directly encode the holistic properties of the input images. Although this might save the global descriptor-based VPR methods (Liu and Zhang, 2012; Schönberger et al., 2018; Revaud et al., 2019; Yin et al., 2019) some compute time, they are more vulnerable to robot pose changes than their local descriptor-based counterparts while being inept at capturing geometric structures (Dube et al., 2020). Yet, global descriptors are reported to be more effective in the case of varying lighting conditions (Lowry et al., 2015). Furthermore,

there are hybrid approaches (Siméoni et al., 2019; Cao et al., 2020; Hausler et al., 2021), which combine the strengths of both approaches.

Deep learning has made key contributions to recent work on VPR. An influential deep-learning-based approach is NetVLAD (Arandjelovic et al., 2016), which is a supervised method for place recognition, based on the Vector of Locally Aggregated Descriptors (VLAD), a technique to construct global image feature representations from local feature descriptors. NetVLAD uses a pre-trained feature extraction network, such as AlexNet (Krizhevsky et al., 2017), to extract the local features, and a loss function that aims to minimize the distance between a baseline input and the most similar image (the positive example), while maximizing the distance between baseline input and the most dissimilar image (the negative example). This loss function is also known as the triplet loss function. Several authors have extended NetVLAD in different directions, and NetVLAD-based methods still perform very competitively (Hausler et al., 2021; Yu et al., 2020).

SNNs have been of interest for various robotics tasks, including not only VPR, but also object detection (Kim et al., 2020), regression (Gehrig et al., 2020), and control of aerial platforms (Vitale et al., 2021) due to their significant potential for computational efficiency (Zhu et al., 2020). Published VPR methods based on SNNs are relatively recent, compared to other robotics research areas. Among them, Hussaini et al. (2022) is reported to be the first high-performance SNN for VPR. There, the authors propose a feed-forward SNN, where the output neuron activations are filtered through a custom softmax layer. Follow-up work by the same authors (Hussaini et al., 2023) introduced a framework where localized spiking neural ensembles are trained to recognize places in particular regions of the environment. They further regularize these networks by removing output from "hyper-active neurons," which exhibit intense spiking activity when provided with input from the regions outside of the ensemble's expertise. This framework yields a significant improvement over its predecessor while demonstrating either superior or competitive VPR performance compared to the traditional methods. A recent study by Hines et al. (2024) presented an SNN model composed of an ensemble of modified BliTNet (Stratton et al., 2022) modules, each tuned to specific regions within the navigation domain. During training, spike forcing is utilized to encode locations uniquely, which are later identified by monitoring the output neuron with the highest spike amplitude. The authors report remarkable improvements in both training and inference times, alongside achieving superior or comparable VPR performance compared to earlier SNN models. However, training of these SNN approaches do not scale with the increasing volume of training data. In addition, heuristics such as the assignment of neural ensembles to spatial regions, nearest neighbor search in the similarity matrix, and the regularization process further complicate the training process and the computational efficiency of the model. In contrast to these previous SNN-based approaches, we propose an end-to-end solution that is much easier to train and to deploy without requiring heuristic training.

3 LoCS-Net model for visual place recognition

Here, we begin with an overview of the task formulation and the architecture of LoCS-Net in Section 3.1. Section 3.2 formally poses the VPR problem as a classification task. Then, in Section 3.3, we walk through the LoCS-Net pipeline and its key design choices. Moreover, Section 3.3 provides a summary of the ANN-to-SNN conversion paradigm while elaborating on its use for the present work. We would like to refer the readers to the supplementary information and to the figshare repository of our code for further implementation details: https://figshare.com/s/c159a8680a261ced28b2.

3.1 Overview

Figure 1 depicts the overall architecture of LoCS-Net. The input to the model is a set of images sampled along a trajectory that traverses a bounded navigation domain. The domain is discretized by means of a uniform grid (orange lines in Figure 1) and each image is assigned an integer *place* label based on the tile traversed at the time of sampling the image. In this manner, we define the VPR task as a classification problem as discussed in Section 3.2.

Each layer in the LoCS-Net model consists of LIF neurons (Burkitt, 2006). In order to train the model, these neurons are converted to rate-based approximations of LIF units (Hunsberger and Eliasmith, 2015). Rate-based LIF approximations are continuous differentiable representations of the LIF activation function. The LIF activation function describes the time evolution of the neuron's membrane potential, and it is discontinuous: when the membrane potential reaches a threshold value, it is reset back to a pre-determined state. The rate-based approximation is a continuous function that describes the neuron's firing rate as a function of its input, enabling the use of back-propagation algorithms for training. However, this doesn't prevent the substitution of the approximate LIF neurons with the original ones for inference after the training is complete. A number of authors have reported successful applications (Rueckauer et al., 2017; Hu et al., 2021; Patel et al., 2019) of ANN-to-SNN conversion.

3.2 VPR as a classification task

A common practice in approaching the VPR task is to pose it as an image retrieval problem where the goal is to compute and store descriptors that would effectively encode both the set of query images and the collection of reference images to match (Lajoie and Beltrame, 2022). The encoding process is followed by an image retrieval scheme, which is based on comparing query embeddings (z_q) to the database of reference descriptors (z_r) with respect to the customized similarity metrics. Nevertheless, computation of the descriptors is numerically expensive. In contrast, we formulate the VPR task as a classification problem in order to bypass the encoding phase of the images. We designed the LoCS-Net so that it would uniquely map the given input images to the mutually

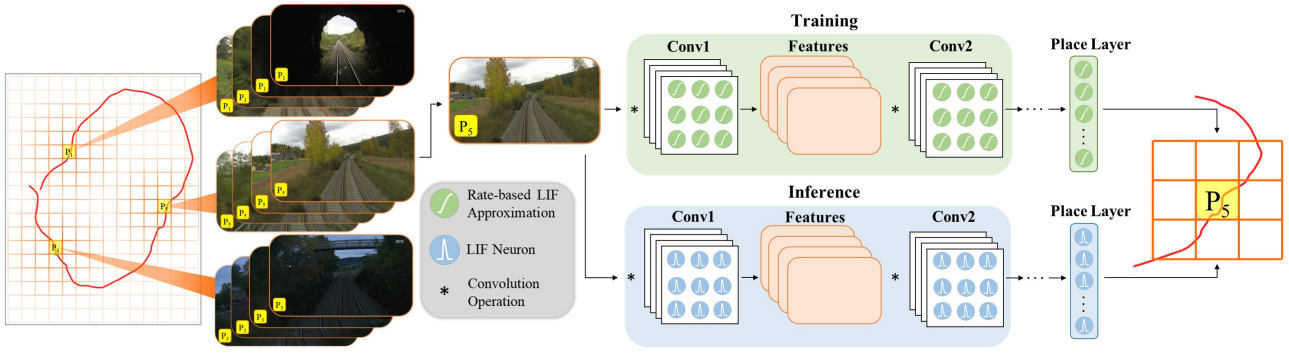


FIGURE 1
LoCS-Net VPR system: a convolutional network of rate-based LIF neurons (Hunsberger and Eliasmith, 2015) is trained over a set of annotated images sampled over a trajectory (the red curve) traversing a finite discretized (orange grid bounded by gray lines) navigation domain. The VPR task is formulated as a classification problem where each tile of the grid ($P_1, P_2, P_3, \ldots$) corresponds to a distinct location. After training, the LIF approximations are substituted with the spiking LIF neurons (Burkitt, 2006) for the inference step.

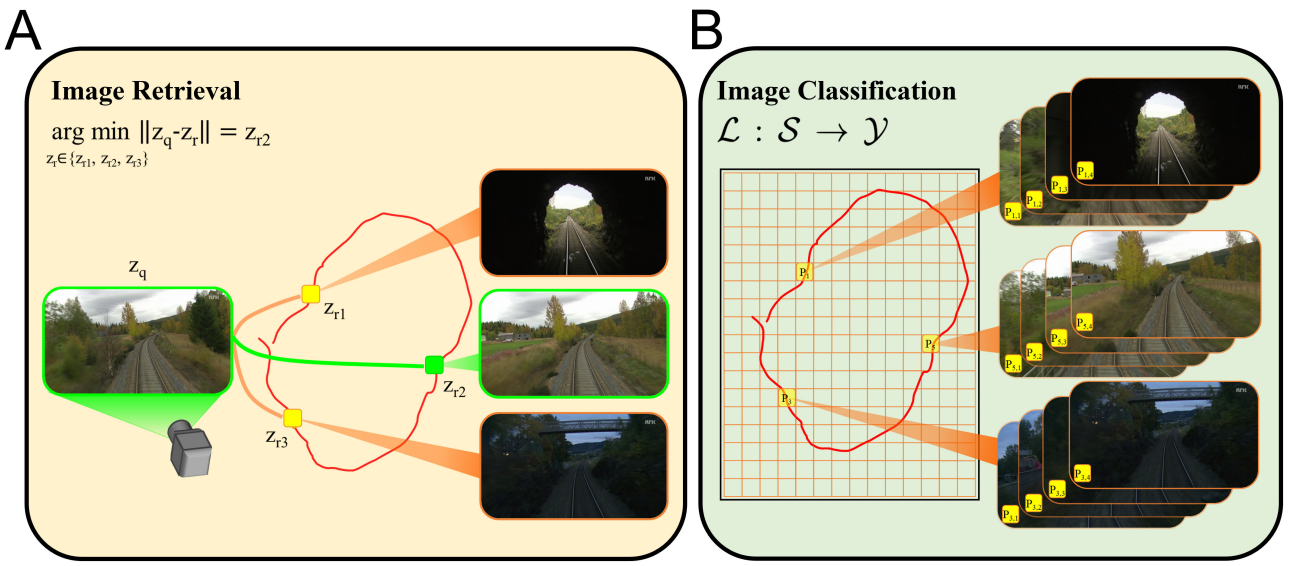


FIGURE 2
VPR can be posed as image retrieval task or image classification problem. For both formulations we consider a set of images collected over a trajectory (the red curves) traversing a finite navigation domain. **(A)** A popular VPR solution is based on generating descriptors for query (z_q) and reference images (z_r), which are then compared to each other in terms of a distance (or similarity) metric $\|\cdot\|$ in order to retrieve the reference image corresponding to the correct place. For instance, z_{r2} represents the most similar reference image to the given query image. **(B)** In contrast, the image classification formulation of the VPR task requires an arbitrary discretization of the navigation domain to define the classes P_i (the places where $i \in \mathcal{Y}$) that annotate the images $s \in \mathcal{S}$. Then, a classifier $\mathcal{L}$ is trained to map images $s \in \mathcal{S}$ to the correct place labels $i \in \mathcal{Y}$. The image annotation $P_{i,j}$ denotes the j^{th} image associated with the class P_i.

exclusive classes, which are the distinct places, as discussed in Sections 3.1, 3.3.

Figure 2 illustrates how our work formulates VPR differently compared to the image retrieval VPR formulation. We first discretize the navigation domain by using a uniform rectangular grid (Figure 2B, the orange lines). Here, each tile of the grid defines a distinct place P_i, $i = 1, 2, 3, \cdots$. We would like to note that the navigation domain can be any physical environment with points described by spatial coordinates. Although we use a uniform rectangular grid to discretize the top-down view of the domain of interest, our approach is flexible with respect to the definition of places, and permits 3-D as well as 2-D discretization. As one of many ways to generate the training and test data, we sample images over numerous trajectories traversing the discretized navigation domain. Suppose that an image $s \in \mathcal{S}$ is sampled at the time instant when the camera is in the region represented by tile P_5. Then, this image would be annotated by the place label $5 \in \mathcal{Y}$. Namely, the image s belongs to the class represented by the tile P_5. Thus, given a query image, our goal is to train a spiking neural network model that would correctly infer the associated place labels. Hence, we pose the VPR task as an image classification problem in this fashion. We

now formally describe the VPR task as a classification problem as follows.

Consider a set of images, $\mathcal{S} = \{s \in \mathbb{R}^{C\times H\times W} | \, X(s) \in \mathcal{D}\}$, where C is the number of color channels, H and W are the height and width of the images in pixels and $\mathcal{D}$ is a pre-determined finite horizontal navigation domain. Here, $X : \mathcal{S} \to \mathcal{D}$ is a function that maps the images $s \in \mathcal{S}$ to the planar spatial coordinates $[x_s, y_s]^{\mathrm{T}} \in \mathcal{D} = \{[d_1, d_2]^{\mathrm{T}} \in \mathbb{R}^2 | \, x_{\min} \leq d_1 \leq x_{\max} \wedge y_{\min} \leq d_2 \leq y_{\max}\}$ where $x_{\min}$, $x_{\max}$, $y_{\min}$, and $y_{\max}$ are the bounds of $\mathcal{D}$. $X(s)$ describes the in-plane spatial state of the camera with respect to a local frame of choice when $s \in \mathcal{S}$ is sampled. The set $\mathcal{Y} = \{i \in \mathbb{N} | \, i \leq N_P\}$ contains the place labels that annotate $s \in \mathcal{S}$ where N_P is the number of assumed places. Each $y \in \mathcal{Y}$ corresponds to a $P_y \subset \mathcal{D}$ such that $P_y \cap P_i \equiv \emptyset$, $y \neq i \wedge i \in \mathcal{Y}$. We formulate the VPR task as an image classification problem, where each class is assumed to be mutually exclusive. That is, each image belongs exactly to one class. Our goal is to design a mapping $\mathcal{L} : \mathcal{S} \to \mathcal{Y}$ that correctly predicts the place label $y \in \mathcal{Y}$ of any given $s \in \mathcal{S}$. One should note that the approach we describe here is different than the image retrieval formulation as we want $\mathcal{L}$ to predict the place labels instead of directly associating the input images with the reference images.

3.3 Localizing convolutional spiking neural network

The design of LoCS-Net is defined mainly by two ideas: (1) Discretization of the given finite navigation domain, (2) Leveraging the back-propagation algorithm by adopting the ANN-to-SNN conversion paradigm. We now walk through the details of these ideas together with the architecture of LoCS-Net and its building blocks, LIF neurons.

3.3.1 The LIF neuron model

Unlike standard artificial neurons, which are defined by time-independent differentiable non-linear transfer functions with continuous outputs, spiking neurons have time-dependent dynamics that aim to capture the information processing in the biological neural systems by emitting discrete pulses (Burkitt, 2006). Equation 1 describes the dynamics of an LIF neuron.

$$C_m \frac{dv(t)}{dt} = -\frac{C_m}{\tau_m}\left[v(t) - v_0\right] + I_s(t) + I_{inj}(t) \tag{1}$$

where C_m is the membrane capacitance, τ_m is the passive membrane time constant, and v_0 is the resting potential. Above formulation considers a resetting scalar state variable, the membrane potential $v(t)$, which will be reinitialized at $v(t) = v_{reset}$ after reaching a threshold, $v(t) = v_{th}$. Whenever the re-initialization happens at time $t = t_{spike}$, the output of the LIF neuron ($o(t)$) will be an impulse signal of unity. We name this a spike event. One can express a spike event of an LIF neuron by Equation 2, which incorporates Dirac's delta function centered at the time of re-initialization.

$$o(t_{spike}) = \delta\left[v(t_{spike}) - v_{th}\right] \tag{2}$$

The right hand side of Equation 1 includes three terms: (1) An exponential decay term (a.k.a the passive membrane leak), (2) $I_s(t)$, the sum of incoming synaptic currents, which are mostly unit impulses filtered through a first order delay and/or multiplied by some scalar, and finally (3) An injection term, $I_{inj}(t)$, that describes the input currents other than synaptic currents. This can be some bias representing the background noise in the corresponding neural system, or just some external input.

Solving the sub-threshold dynamics described by Equation 1 for the firing rate $\rho[I_s(t)]$ of an LIF neuron and assuming $I_{inj}(t) = 0$ for all $t \geq 0$ yields the following.

$$T_{spike} = -\tau_m \log\left(1 - \frac{(v_{th} - v_{reset})\frac{C_m}{\tau_m}}{(v_0 - v_{reset})\frac{C_m}{\tau_m} + I_s(t)}\right) \tag{3}$$

$$\rho[I_s(t)] = \begin{cases} 0 & \text{if } I_s(t) \leq I_{th} \\ \dfrac{1}{T_{ref} + T_{spike}} & \text{if } I_s(t) > I_{th} \end{cases} ; \; I_{th} = (v_{th} - v_0)\frac{C_m}{\tau_m} \tag{4}$$

T_{ref} is the refractory period, which is the time it takes a neuron to start accepting input currents after a spike event. T_{spike} is the time it takes a neuron to reach v_{th} from v_{reset} after a spike event at some $t = t'$ given $v_{th} < I_s(t) = c \in \mathbb{R}$, $t' < t \leq t' + T_{spike}$. Equations 3, 4 describe the response curve of an LIF neuron, which has a discontinuous and unbounded derivative ($\partial\rho/\partial I_s$) at $I_s = (v_{th} - v_0)C_m/\tau_m$. However, one can modify (Equation 4) as described by Hunsberger and Eliasmith (2015) in order to obtain a smooth rate-based LIF approximation.

$$\begin{aligned} \rho'\left[I_s(t)\right] &= \left\{T_{ref} + \tau_m \log\left(1 + \frac{v_{th}}{\Theta\left[I_s(t) - v_{th}\right]}\right)\right\}^{-1}; \\ \Theta(x) &= \gamma \log\left(1 + e^{x/\gamma}\right) \end{aligned} \tag{5}$$

where γ is the smoothing factor of choice.

3.3.2 ANN-to-SNN conversion

Due to the discontinuities introduced by discrete spike events, the conventional gradient-descent training techniques need to be modified for spiking neural networks. Various approximation methods have been developed to overcome these discontinuities (Neftci et al., 2019). One such method is based on the utilization of the rate-based approximations, a.k.a. the tuning curves. Given a loss function, the main idea is to build a network of differentiable rate-based approximation units and solve for the synaptic weights by using an arbitrary version of gradient descent. Once the solution is obtained, the approximation units can be substituted with LIF neurons to use the resulting spiking network during inference as shown in Figure 3. We utilized NengoDL Rasmussen (2018) to implement the aforementioned ANN to SNN conversion methodology. We employed the standard sparse categorical cross entropy as our loss function.

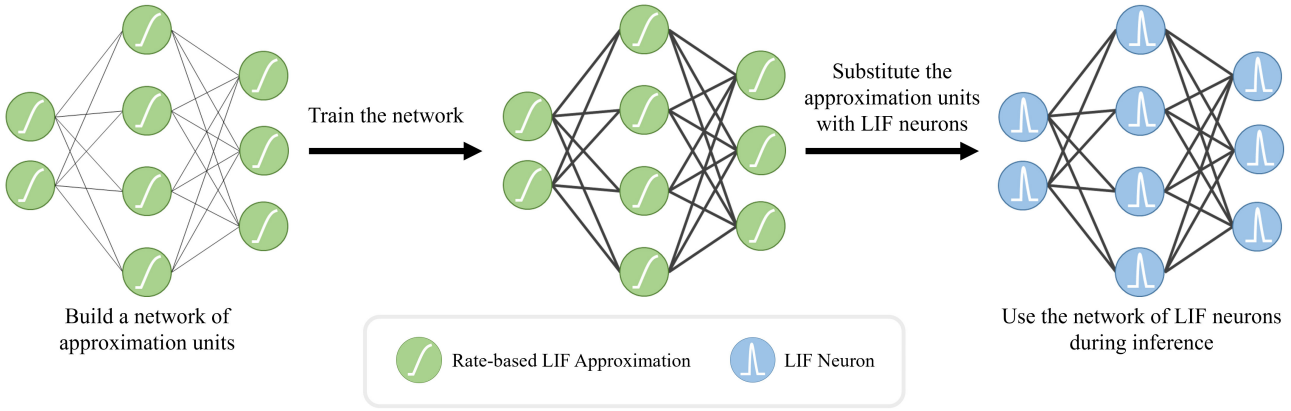


FIGURE 3
ANN-to-SNN conversion work-flow: we first employ rate-based approximations of the LIF neurons to train our network, since the discontinuous spike event outputs of the original LIF neurons prevents the training of the network through the back-propagation algorithm. After completing the training of this interim network, we substitute the LIF approximations with the original ones while keeping the network topology and the trained weights (bold black lines) the same.

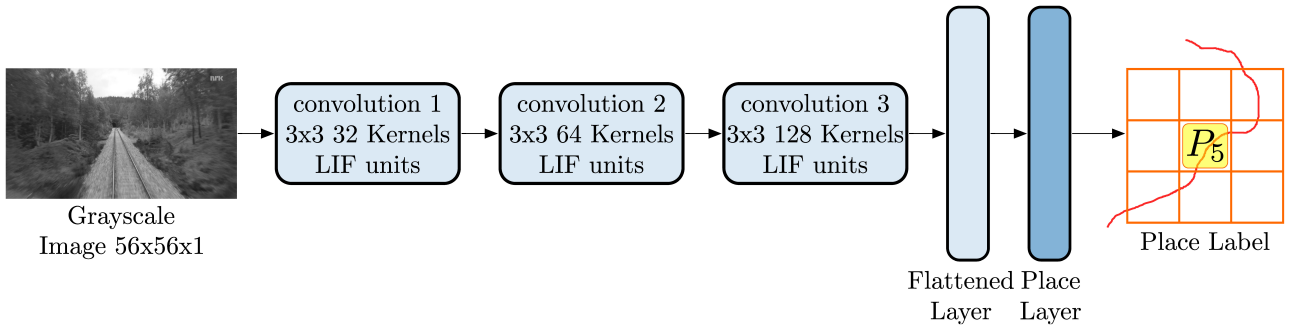


FIGURE 4
The LoCS-Net architecture consists of 3 convolutional layers followed by a fully connected output layer, known as the *place layer*. The units within this layer correspond to unique locations within the environment. LoCS-Net accepts 56 × 56 pixel grayscale images as inputs, using them to predict the associated places from which the input images were sampled.

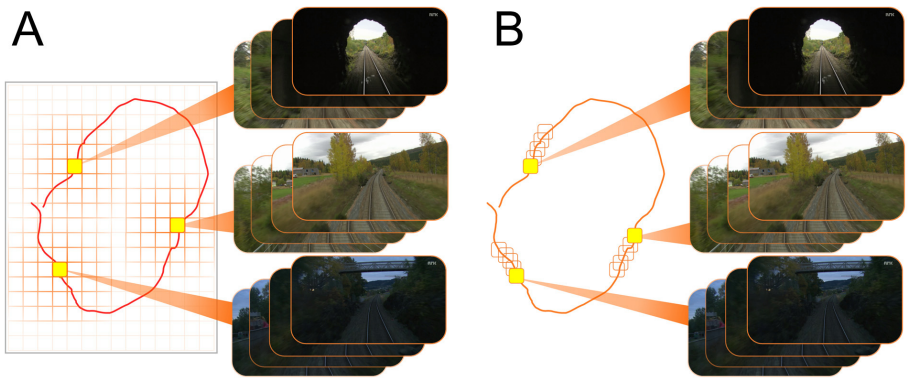


FIGURE 5
Annotating images: **(A)** top-down navigation domain is discretized by defining a grid of arbitrary resolution. Each tile of of the grid annotates the images sampled within its boundaries. **(B)** Images are sampled over a traverse at a pre-determined frequency while each image is corresponding to a unique place. For instance, if the first image is sampled at time $t = 0\,s$, then the second and the third image will be sampled at $t = T\,s$ and $t = 2T\,s$, respectively.

TABLE 1 LoCS-Net training and test data specifications.

Specifications \ dataset	Nordland	Oxford RobotCar (ORC)
Train size [# of images]	6,144	32,475
Test size [# of images]	3,072	17,055
# of labels	3,072	2,500
# of unique labels	3,072	185

3.3.3 LoCS-Net architecture

As depicted in Figure 4, LoCS-Net is composed of a sequence of 3 convolutional layers followed by a fully connected output layer, also known as the "place layer," the units of which correspond to distinct places in the environment. Inputs to LoCS-Net are grayscale images of 56 × 56 pixels. The number of neurons in the place layer is set to be the number of possible places (N_P) as explained in Section 3.2. We considered 50 × 50 grid for Oxford RobotCar (ORC) data in our principal experiments. Note that for training, we employ the smooth rate-based approximated LIF units while maintaining the same architecture illustrated in Figure 4. We use sparse categorical cross entropy as the loss function during training. For inference, we replace the approximated LIF units of the trained network with spiking LIF neurons, keeping both the weights and the architecture unchanged. For further details of the network structure and the corresponding hyper-parameters, we refer the readers to the supplementary information and to the repository of the current work's code at https://figshare.com/s/c159a8680a261ced28b2.

4 Experiments

4.1 Datasets and evaluation metrics

We evaluate our proposed approach on the challenging Nordland (Olid et al., 2018) and ORC data (Maddern et al., 2017, 2020) following prior work (Hussaini et al., 2023). For the Nordland data experiments, we trained LoCS-Net using the spring and fall traverses and tested it with the summer traverse. For the ORC data experiments, we trained LoCS-Net on the sun (2015-08-12-15-04-18) and rain (2015-10-29-12-18-17) traverses, and tested its performance on the dusk (2014-11-21-16-07-03) traverse. We followed the Nordland data processing directions in Hussaini et al. (2023) for the same training and test data. We obtained 3,072 Nordland data (Olid et al., 2018) places, and 2,500 ORC data (Maddern et al., 2017, 2020) places (set by our grid definition) while considering the complete sun, rain, and dusk traverses used in Hussaini et al. (2023).

Although our discretization of the ORC domain yields a total of 2,500 possible places, the trajectories traversed in that dataset cover a much smaller number of labels. Some of the ORC data places are either occasionally visited or not visited at all. This is because the trajectories were generated by a vehicle traversing the road network, making it impossible to visit all parts of the spatial domain. Therefore, we filter out places that do not contain a minimum number (10) of unique training images. We also bound the number of unique instances per place from above (maximum 700) as the training of the baseline SNN models are getting infeasible due to increasing size of the data. Table 1 provides the training and the test data specifications yielded by our data pre-processing pipeline. We would like to note that LoCS-Net can still be trained and be tested on the full ORC data in a matter of minutes.

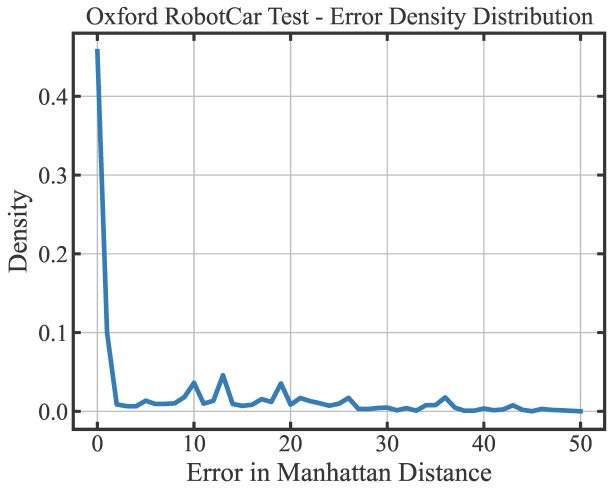


FIGURE 8
Prediction error distribution of LoCS-Net over the ORC dataset: 55.7% of the place predictions of LoCS-Net are within 1-Manhattan Distance of the ground truth labels. Approximately 40% of the LoCS-Net place predictions fall beyond 5-Manhattan Distance of the ground truth labels.

We employ standard VPR performance metrics, including the precision-recall curves, area-under-the-precision-recall curves (AUC-PR or AUC) (Cieslewski and Scaramuzza, 2017; Camara and Přeučil, 2019), and recall-at-N (R@N) curves (Perronnin et al., 2010; Uy and Lee, 2018) in order to assess the performance of our model.

4.2 Experimental set-up

We adopt two annotation methods as the Nordland (Olid et al., 2018) and the ORC data (Maddern et al., 2017, 2020) were structured in different ways. Figure 5A describes the labeling process of the ORC images (Maddern et al., 2017, 2020). As it is shown, we first encapsulated the top-down projection of the path within a rectangular region. Then, we discretize this region to obtain grid tiles, each of which represents a distinct place. These tiles annotate the images sampled within its boundaries.

To label the Nordland images (Olid et al., 2018) we followed the annotation method defined in Hussaini et al. (2023). As depicted in Figure 5B, we sample images over a traverse at a pre-determined frequency (every 8th image) while each image is corresponding to a unique place.

TABLE 2 VPR performance comparison in terms Precision at 100% Recall (P@100%R), area-under-the-precision-recall curves (AUC), mean inference time (MIT), mean training time (MIT), and effective energy consumed per inference.

Method	Approach	Nordland					ORC				
		P@100%R	AUC	MIT [ms]	MTT [min]	Effective energy per inference [J]	P@100%R	AUC	MIT [ms]	MTT [min]	Effective energy per inference [J]
LoCS-Net on GPU (ours)	SNN	**78.6%**	**0.980**	25	**1**	2.545	**45.7%**	**0.702**	10	**3.5**	1.095
LoCS-Net on NUC (ours)	SNN	**78.6%**	**0.980**	796	-	13.183	**45.7%**	**0.702**	371	-	5.871
LoCS-Net on Loihi (ours)	SNN	71.1%	0.761	288	-	**0.060**	41.0%	0.653	147	-	**0.032**
VPRTempo on GPU (Hines et al., 2024)	SNN	73.0%	0.975	**8**	15	0.079	20.2%	0.435	**5**	54	0.053
Ensemble SNNs on CPU (Hussaini et al., 2023)	SNN	66.9%	0.975	408	725	3.405	17.6%	0.485	290	3,408	3.051
WNA (Hussaini et al., 2022)	SNN	0.3%	0.005	-	-	-	4.0%	0.042	-	-	-
MixVPR (Ali-Bey et al., 2023)	ANN	**94.6%**	-	29	3	0.907	**87.7%**	-	14	**6**	0.578
Conv-AP (Ali-bey et al., 2022)	ANN	91.3%	-	27	**2**	0.847	84.6	-	18	8	0.632
EigenPlaces (Berton et al., 2023)	ANN	80.2%	-	57	5	6.443	71.5%	-	31	17	3.335
CosPlace (Berton et al., 2022)	ANN	75.3%	-	60	6	6.842	71.0%	-	35	17	3.524
AP-GeM (Revaud et al., 2019)	ANN	65.1%	-	95	9	10.512	60.7%	-	54	27	5.376
NetVLAD (Arandjelovic et al., 2016)	ANN	51.4%	-	107	10	12.641	43.8%	-	62	29	5.496

Bold values indicate the best performance metrics for SNN- and ANN-based approaches on individual datasets.

10.3389/fnbot.2024.1490267

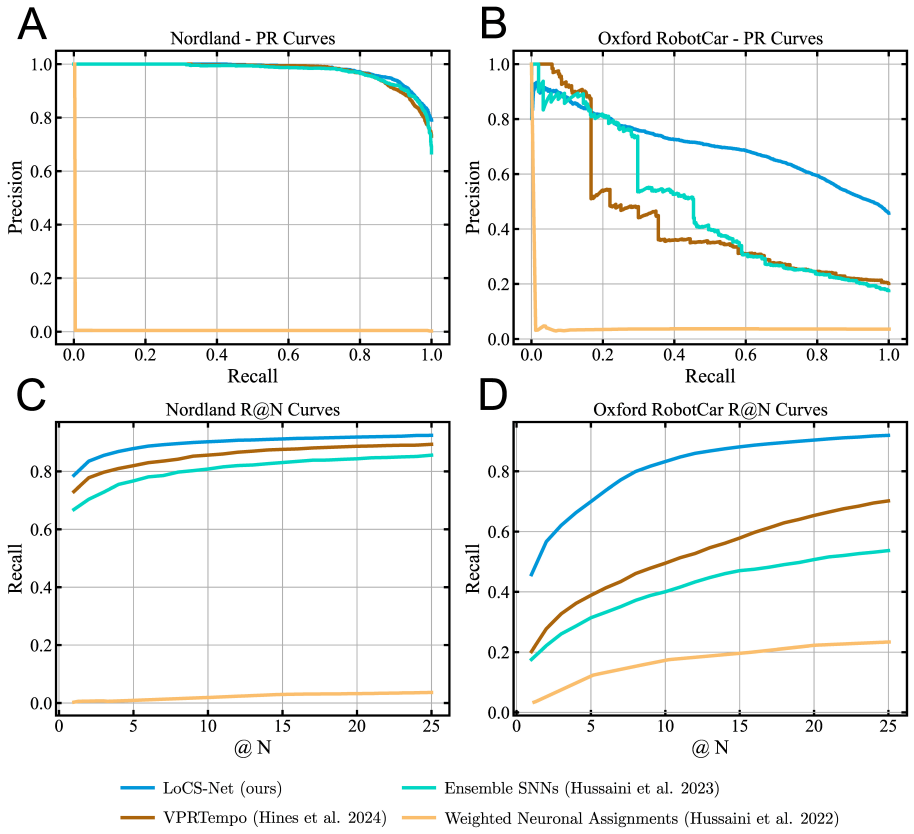


FIGURE 6
Precision-Recall and Recall @ N curves for the baseline SNN-based VPR methods and LoCS-Net: The blue, brown, cyan, and orange curves correspond to LoCS-Net, VPRTempo (Hines et al., 2024), Ensemble SNNs (Hussaini et al., 2023), and Weighted Neuronal Assignments (Hussaini et al., 2022), respectively. These figures demonstrate that LoCS-Net yields the best SNN-based VPR performance on both datasets. **(A)** PR curves obtained from the experiments on the Nordland dataset. **(B)** PR curves obtained from the experiments on the ORC datasets. **(C)** The R@N curves obtained from the experiments on the Nordland dataset. **(D)** The R@N curves obtained from the experiments on the ORC dataset.

4.3 Quantitative results

We conducted several performance comparisons of LoCS-Net with the current SOTA SNN methods, Ensemble SNNs (Hussaini et al., 2023), VPRTempo (Hines et al., 2024), and Weighted Assignment SNN (WNA) (Hussaini et al., 2022). In order to save computational resources, we did not train and test WNA ourselves. Instead, in Table 2, we listed the performance metrics published in Table 1 of Hussaini et al. (2023). We also included additional performance comparisons of LoCS-Net to a set of ANN-based SOTA VPR techniques such as AP-GeM (Revaud et al., 2019), NetVLAD (Arandjelovic et al., 2016), MixVPR (Ali-Bey et al., 2023), Conv-AP (Ali-bey et al., 2022), EigenPlaces (Berton et al., 2023), and CosPlace (Berton et al., 2022). We utilized the benchmark tool developed by Berton et al. (2023) in order to perform these additional comparisons.

Table 2 and Figure 6 summarize the VPR performance of LoCS-Net along with the reference methods. We observe that LoCS-Net outperformed all the SNN-based methods on both the Nordland (Olid et al., 2018) and ORC dataset (Maddern et al., 2017, 2020) by a large margin (78.6% and 45.7% respectively) in terms of P@100%R. LoCS-Net took much less time to train as reported in Table 2, which highlights LoCS-Net's compatibility for rapid prototyping and real-world deployment. Although it falls short of top-performing ANNs such as MixVPR and Conv-AP, LoCS-Net's strengths lie in energy efficiency and training time. While its GPU-based energy usage (2.545J) sits between that of ANNs like EigenPlaces (1.283J) and AP-GeM (5.376J), deploying LoCS-Net on neuromorphic hardware (Loihi) drastically reduces energy consumption, reaching just 0.032J per inference.

Moreover, Figures 6C, D present the Recall @ N curves obtained from the evaluations of the methods on the Nordland (Olid et al., 2018) and ORC datasets (Maddern et al., 2017, 2020). LoCS-Net consistently yields the best Recall @ N performance compared to SNN methods on both datasets. These results indicate good scalability of the LoCS-Net model across thousands of locations, while maintaining computationally efficient inference, as illustrated by Table 2.

We conduct a sensitivity analysis of LoCS-Net with respect to the number of neurons used for signal representation, the maximum firing rate, and the synaptic smoothing factor. We note that the nominal LoCS-Net does not include synaptic filters in order to avoid the additional complexity imposed by temporal dynamics during training, as capturing precise synaptic dynamics is not our primary objective. Figure 7 presents the P@100%R sensitivity analysis of LoCS-Net on the Nordland Figure 7A and

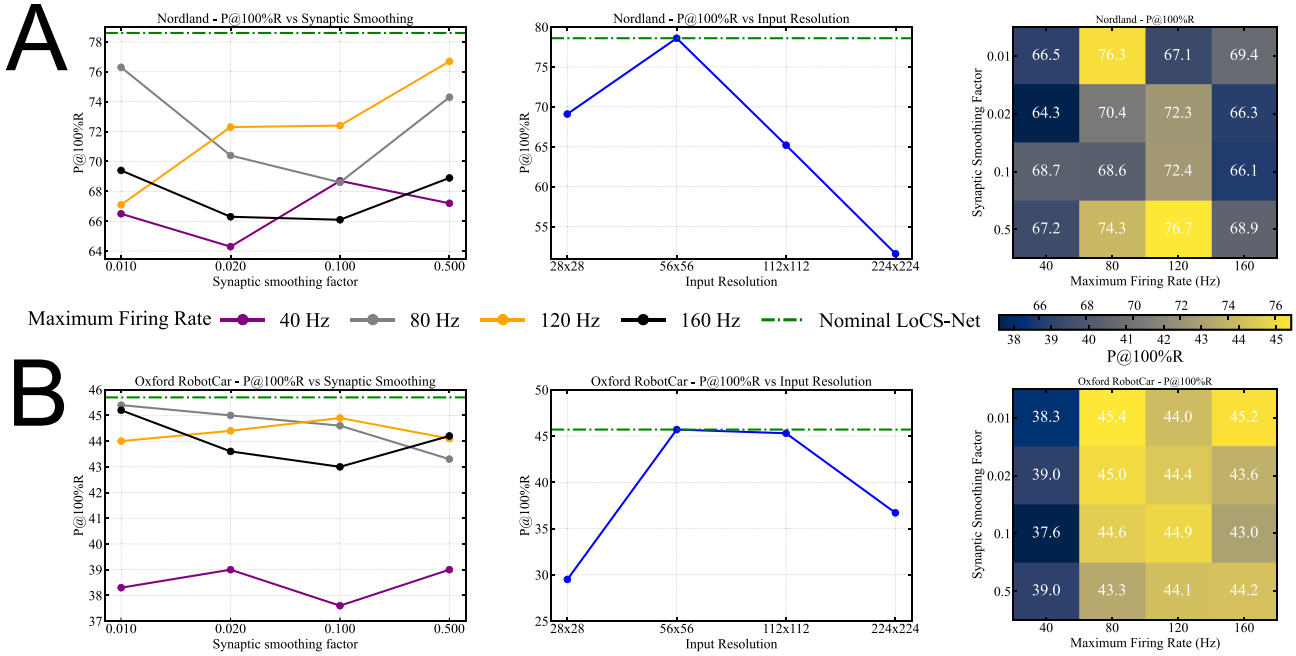


FIGURE 7
VPR performance in P@100%R sensitivity of LoCS-Net on Nordland **(A)** and ORC **(B)** datasets with respect to synaptic smoothing factor, maximum firing rate, and input layer resolution: The remaining neuronal and training parameters of the nominal LoCS-Net model (see Supplementary Tables S1, S2) are kept the same. Curves in the synaptic smoothing sensitivity plots (on the left) are color-coded, indicating model instances with different maximum firing rates with a nominal input resolution of 56 × 56. Plots in the middle depict the sensitivity with respect to only input resolution. Dashed green curves represent the performance of the nominal LoCS-Net model instance. Synaptic smoothing factor-maximum firing rate tables (on the right) illustrate the same data presented in the plots on the left. These tables contain the P@100%R values in white text while exhibiting the corresponding color codes at the same time. The P@100%R values are obtained by averaging the simulation results of five models, each initiated with a different seed for synaptic weights. The results suggest that LoCS-Net is sensitive to the synaptic smoothing factor, maximum firing rate, and input layer resolution, especially on Nordland data.

ORC Figure 7B datasets, focusing on the synaptic smoothing factor, maximum firing rate, and input layer resolution. All other neuronal and training parameters of the nominal LoCS-Net model remain unchanged. In Figures 7A, B, the synaptic smoothing sensitivity plots on the left use color-coded curves to represent different maximum firing rates for a nominal input resolution of 56 × 56. The middle plots isolate the effect of input resolution on model sensitivity, with dashed green curves showing the performance of the nominal LoCS-Net configuration. On the right, tables summarize the combined influence of the variances in synaptic smoothing factor and maximum firing rate, providing the same information as the left-hand plots. The P@100%R values of Figure 7 are computed as averages across simulations of five models initialized with different seeds per parameter set. The findings highlight that LoCS-Net is sensitive to variations in synaptic smoothing factor, maximum firing rate, and input layer resolution, with sensitivity being particularly evident on the Nordland dataset.

We further seek to understand the distribution of LoCS-Net's prediction errors on the ORC dataset. We quantify the prediction error in in terms of Manhattan Distance, as illustrated in Figure 8. 55.7% of the place predictions are within 1-Manhattan Distance of the ground truth labels. Yet, approximately 40% of the LoCS-Net place predictions fall beyond 5-Manhattan Distance of the ground truth labels. We did not perform the same analysis for Nordland data as it doesn't utilize a grid-based labeling structure as the Oxford RobotCar data.

4.4 Neuromorphic hardware deployment

We deployed the trained LoCS-Net on Kapoho Bay, a USB form that hosts 2 of Intel's neuromorphic Loihi chips (Davies et al., 2021). We utilized NengoLoihi (DeWolf et al., 2020) to deploy LoCS-Net on the Loihi chips. The hardware supports up to 260M trainable synaptic connections with 260k neurons; however, the network structure must be sufficiently tuned to fully utilize the hardware due to its architecture.

After the network parameters are trained, neurons and connections must be distributed between two chips, with 128 neuromorphic cores in each. Each core is designated to handle 1,024 neurons at a time, and the number of core-to-core connections is restricted to about 4,000 synapses due to the limited synapse memory. Therefore, networks with large input/output connections must be partitioned across several cores, and the biases are removed from the convolutional layers to reduce inter-core communication. These strategies have be followed to avoid under-utilization of the cores. As a result, our hardware-deployed network architecture contains fewer trainable parameters with sparse connections due to the above constraints, which may result in a slight decrease in performance. Additionally, as the Kapoho Bay is optimized for mobile deployment and energy efficiency, the device handles spike-timing with 8-bit accuracy, which defines its quantization limit. Training the simulated network without accounting for these hardware specifications might lead

to performance drop during on-chip inference. To minimize such discrepancies, the regularization parameter of the training is tuned to adjust the magnitude of the network weights. Here, we note that the hardware limitations mentioned above may be resolved in future versions of neuromorphic chips.

To examine the energy-saving benefits of neuromorphic hardware, we measured the average energy consumption per inference of LoCS-Net when deployed on Loihi, Intel NUC7i7BNH (a small-form-factor PC suitable for mobile robotics), and GPU (NVIDIA RTX 3060). We utilized pyJoules (Belgaid et al., 2019) to measure the average energy consumption on the GPU and NUC, while employing a standard off-the-shelf USB tester to observe the power drawn by Kapoho Bay. For each type of hardware hosting LoCS-Net, we first measured the idle power and then the power drawn under load while LoCS-Net was operational. We then subtracted the idle power from the load (or total) power to obtain the closest estimate of LoCS-Net's effective energy consumption, which we list in Table 2. Moreover, we report the total inference energy values in Figure 9.

5 Discussion

We observe a noticeable performance drop of the on-chip LoCS-Net, while achieving at least an order of magnitude improvement in energy efficiency on both datasets compared to CPU and GPU deployments. We believe that the gradient mismatch between the LIF neurons (Burkitt, 2006) and their rate approximations (Hunsberger and Eliasmith, 2015) significantly contribute to the reduced performance of LoCS-Net in this case, as also mentioned by Che et al. (2022). LoCS-Net outperforms all SNN-based methods on both datasets in terms of area-under-the-precision-recall curves, while demonstrating the second fastest inference as shown in Table 2. VPRTempo turns out to be the fastest (in terms of inference time) and the most energy-efficient simulated SNN, coming close after on-chip LoCS-Net in terms of energy consumption per inference. However, it fails to exhibit robustness against dynamic scenes, noise, variance in viewpoint, and lighting conditions in the ORC dataset.

We observe relatively poor performance of our method on the ORC dataset (Maddern et al., 2017, 2020). In addition to ANN-to-SNN conversion losses, we hypothesize that the more dynamic scene content of the ORC images (Maddern et al., 2017, 2020) and the substantial noise levels in a significant portion of the test ORC images impede better VPR performance of LoCS-Net.

As reported in Table 2, LoCS-Net consumes 0.06 J per inference when processing the Nordland images, approximately 1/40th of the energy consumed by the GPU and about 1/220th of the energy consumed by the CPU of the NUC. Similarly, Loihi chips demonstrate the greatest energy efficiency (0.032 J/inference vs. 5.871 J/inference on NUC and 1.095 J/inference on GPU) by a large margin when processing the ORC images. We consistently observe the total energy consumption of neuromorphic chips does not scale intuitively with respect to the size of the neural network in terms of the number of trainable parameters. Instead, the inference energy cost appears to be more related to the communication time between the integrated CPU and the Loihi chip. This includes the time required to generate and to send the spike signals through the input layer of LoCS-Net using the integrated CPU within the Loihi device, and to decode the output signal back to numerical data. This observation suggests that a significant restriction of energy efficient neuromorphic computation involves data conversion during encoding and decoding, which must be managed by traditional CPU architecture. The communication bottleneck also affects the total inference time. Due to the communication delay, there is a challenge in optimally including the spike-conversion stage in-between the sensing and input neurons, as well as between the output layer and the actuator. Unfortunately, this spike-conversion step scales linearly with the data size and the resolution we aim to represent. However, testing the proposed method on alternative neuromorphic hardware designs (Hazan and Ezra Tsur, 2022; Halaly and Ezra Tsur, 2023) might offer even greater power efficiency and yield faster inference times.

The overall performance of the SOTA ANN methods proved superior to that of their SNN competitors in terms of precision at 100% recall on both the Nordland and ORC datasets. As shown in Table 2, while these SOTA ANN techniques outperform their SNN competitors, they also require significantly longer time to generate descriptors (loosely corresponding to training time) and to compute reference-query matches (corresponding to inference time) compared to LoCS-Net. On-chip LoCS-Net remains the most energy-efficient VPR method, with the fastest training time by a large margin.

We must also note that the SOTA ANN approaches included in our comparison studies use pre-trained networks (e.g., ResNet50 and ResNet101 backbones), which are subsequently fine-tuned for the VPR tasks. In contrast, LoCS-Net is trained solely on data from the navigation domain of interest. In this sense, comparing our network to these SOTA ANN techniques may not be entirely fair, as they benefit from cumulative training over a much larger dataset. We would like to emphasize that our work focuses on the SNN domain, and LoCS-Net significantly advances the state of the art in SNN-based VPR techniques.

We may further analyze the performance of LoCS-Net by investigating the spiking activity in the convolutional layers of the model. Figures 10, 11 depict representative samples of Nordland and ORC training and test images. Compared to the ORC images, Nordland test and training instances are much more visually aligned. ORC test images, on the other hand, are extremely challenging due to intense variance in lighting, appearance, viewpoint, and noise. Some of these test instances are impossible to recognize by a human observer. We believe that these characteristics of the ORC data significantly contribute to LoCS-Net's reduced performance on this dataset.

We examined the activities generated by a set of randomly chosen images that were either correctly or incorrectly labeled by LoCS-Net. The images shown in this section are representative samples for both mislabeled and correctly labeled images from the Nordland and Oxford RobotCar datasets. We observed similar spiking activity patterns in all of the images we randomly picked as in those demonstrated in Figures 10, 11.

Figures 10, 11 depict the spiking unit activations of the first convolutional layer in the model, when presented images

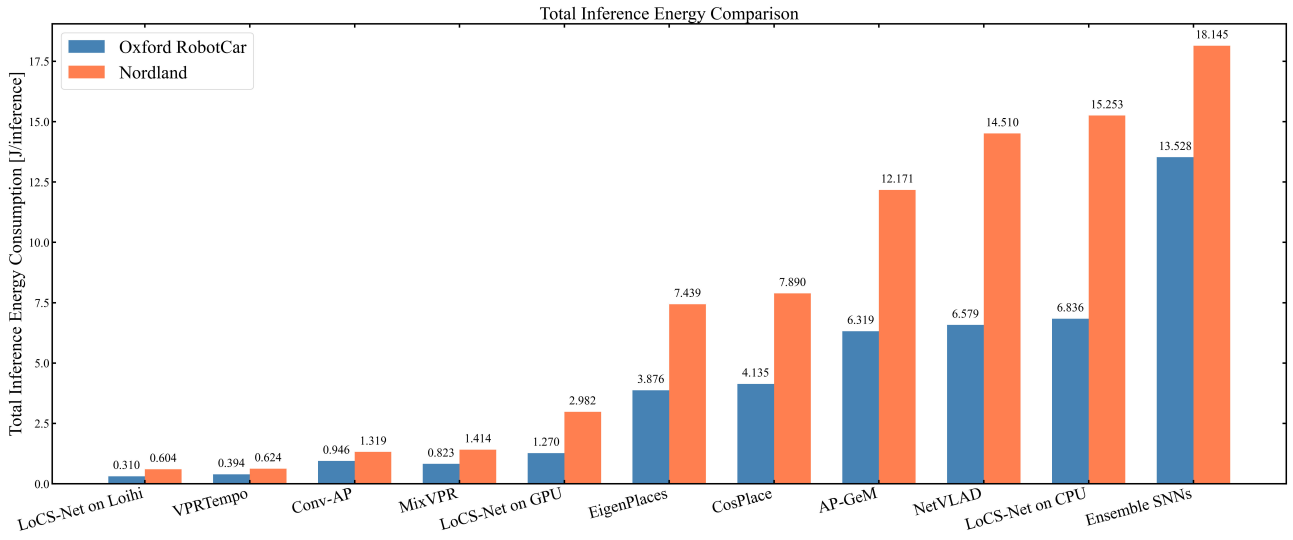


FIGURE 9
Total energy consumption (in Joules) per inference for different VPR methods: Orange bars represent energy consumption on Nordland data, while blue bars correspond to ORC data. The LoCS-Net model deployed on Loihi achieves the lowest energy consumption per inference, closely followed by its on-GPU SNN competitor, VPRTempo.

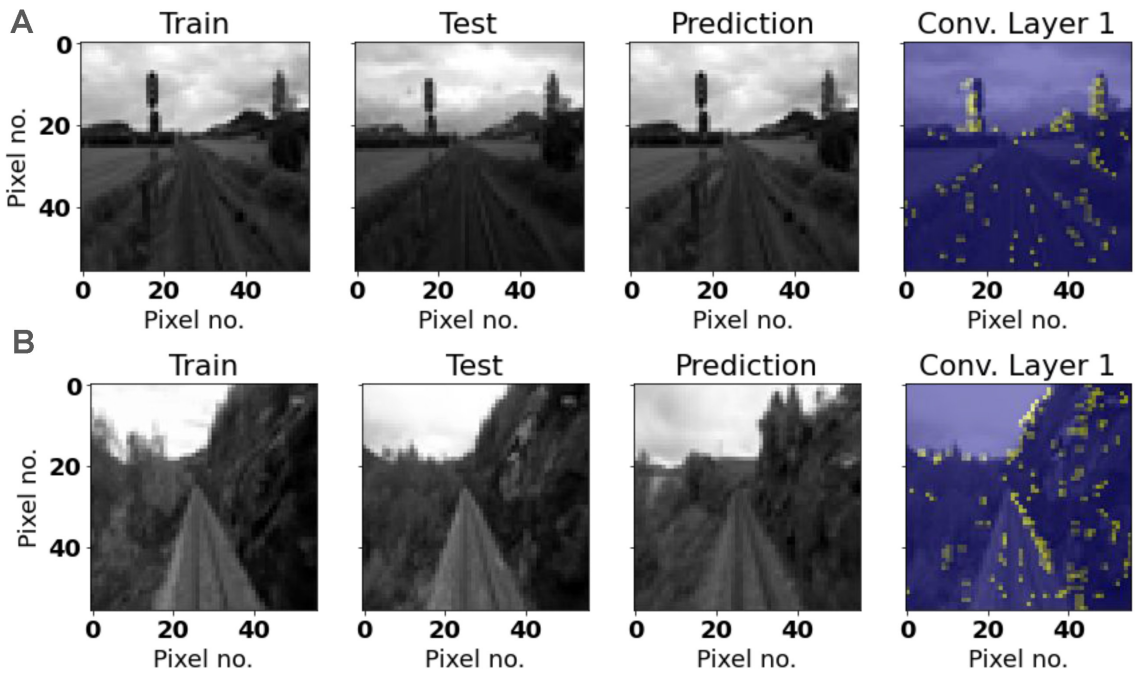


FIGURE 10
Representative samples of Nordland training and test images: Yellow regions represent the associated active neurons. **(A)** An instance of correct prediction. Representative convolutional layer activation appears to represent features in the input image. **(B)** An instance of incorrect prediction. Although the activation of the convolution layer appears to represent features of the image, the prediction made is for a similar, but incorrect class label.

from the Nordland and ORC datasets. In both figures, the correct predictions are associated with spiking activity that is clustered over large features in the input image that could potentially help to distinguish the input image from others. When we examine the spiking activities generated by the images that are mislabeled by LoCS-Net, we observe matching spiking patterns with the activities generated by the training image for the correct class. This implies that LoCS-Net struggles to distinguish images marginally different from each other.

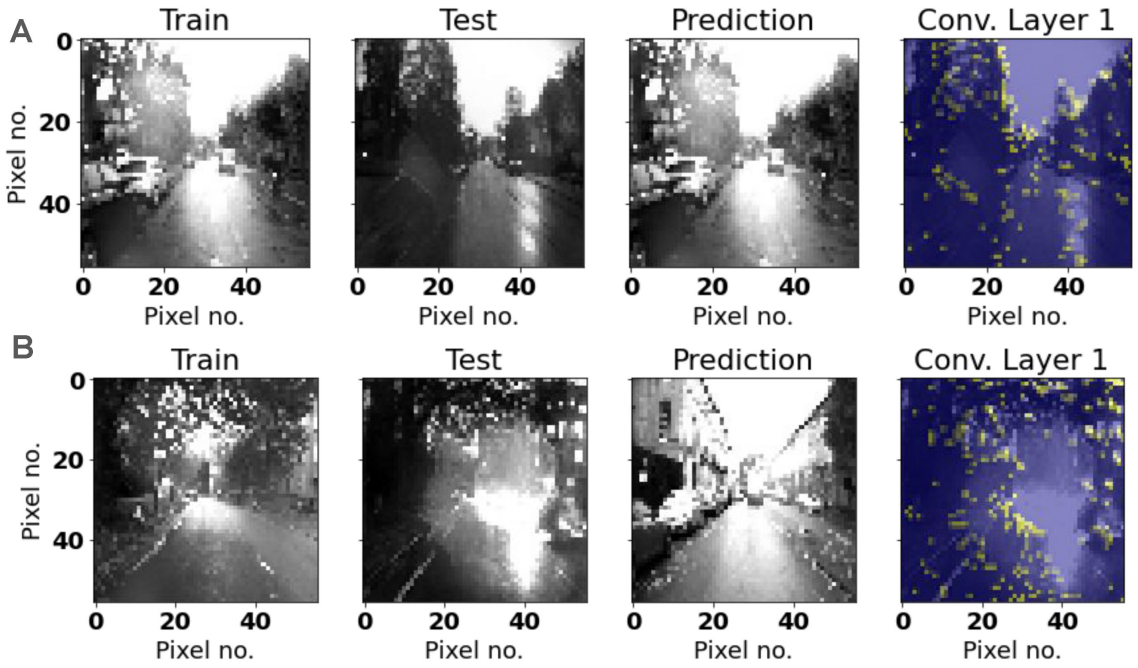


FIGURE 11
Representative samples of Oxford Robot Car training and test images: Yellow regions represent the associated active neurons. **(A)** An instance of correct prediction. Representative convolutional layer activation appears to represent features in the input image. **(B)** An instance of incorrect prediction. A possible water droplet on the lens causes significant image distortion that results in incorrect place prediction.

6 Conclusion

In this work, we formulate visual place recognition as a classification problem and develop LoCS-Net, a convolutional SNN to solve VPR tasks with challenging real-world datasets. Our approach leverages ANN-to-SNN conversion and back-propagation for tractable training, by using rate-based approximations of leaky integrate-and-fire (LIF) neurons. The proposed method substantially surpasses existing state-of-the-art SNNs on challenging datasets such as Nordland and ORC, achieving 78.6% precision at 100% recall on the Nordland dataset (compared to the current SOTA's 73.0%) and 45.7% on the Oxford RobotCar dataset (compared to the current SOTA's 20.2%). Our approach simplifies the training pipeline, delivering the fastest training time and the second fastest inference time among SOTA SNNs for VPR. Hardware-in-the-loop evaluations using Intel's neuromorphic USB device, Kapoho Bay, demonstrate that our on-chip spiking models for VPR-trained through the ANN-to-SNN conversion strategy-continue to outperform their SNN counterparts, despite a slight performance drop when transitioning from off-chip to on-chip, while still offering significant energy savings. These results emphasize the LoCS-Net's exceptional rapid prototyping and deployment capabilities, marking a significant advance toward more widespread adoption of SNN-based solutions in real-world robotics.

The SOTA ANN methods over-shadowed their SNN counterparts in terms of precision at 100% recall on both the Nordland and ORC datasets. However, as detailed in Table 2, these ANN techniques come with notable trade-offs, requiring longer times for descriptor generation and inference relative to LoCS-Net. Among the evaluated methods, the on-chip implementation of LoCS-Net stands out as the most energy-efficient VPR solution, achieving the shortest training time by a considerable margin.

We would like to emphasize that this manuscript proposes LoCS-Net as a environment-specific VPR solution. In that sense, providing a long-term general spatial memory as a global VPR solution is beyond the capabilities of the current work. In addition, LoCS-Net's performance is sensitive to the definition of places, which in turn may require the implementation of domain-specific discretization techniques to maximize the performance of LoCS-Net over different navigation environments. As discussed in Section 4.3, LoCS-Net doesn't perform as well over the Oxford RobotCar (Maddern et al., 2017, 2020) data as compared with the Nordland (Olid et al., 2018) dataset. This might be due to the LIF neuron approximation errors as well as the significantly varying lighting and road conditions of the ORC traverses (Maddern et al., 2017, 2020), which suggests the lack of robustness to such dynamic scenes. Nevertheless, we emprically show that LoCS-Net is much better than its SNN competitiors at handling such challenges.

Data availability statement

The computational model and data preparation scripts generated for this study can be found in the FigShare repository at https://figshare.com/s/c159a8680a261ced28b2. This includes all necessary code to reproduce the results presented in this paper.

Please direct further inquiries and questions about the model implementation to the corresponding authors.

Author contributions

UA: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. IR: Data curation, Formal analysis, Software, Validation, Visualization, Writing – original draft, Writing – review & editing, Investigation. EG: Visualization, Writing – review & editing, Formal analysis, Validation, Data curation. AC: Formal analysis, Validation, Visualization, Writing – review & editing. SK: Data curation, Software, Validation, Writing – original draft, Writing – review & editing. MG: Resources, Supervision, Writing – review & editing. RG: Supervision, Writing – review & editing. IS: Resources, Supervision, Writing – review & editing. GC: Funding acquisition, Project administration, Resources, Supervision, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Office of Naval Research, United States [grant number N00014-19-1-2373].

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2024.1490267/full#supplementary-material

References

Ali-bey, A., Chaib-draa, B., and Giguere, P. (2022). GSV-cities: toward appropriate supervised visual place recognition. *Neurocomputing* 513, 194–203. doi: 10.1016/j.neucom.2022.09.127

Ali-Bey, A., Chaib-Draa, B., and Giguere, P. (2023). "Mixvpr: feature mixing for visual place recognition," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2998–3007. doi: 10.1109/WACV56688.2023.00301

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). "Netvlad: CNN architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5297–5307. doi: 10.1109/CVPR.2016.572

Arandjelović, R., and Zisserman, A. (2012). "Three things everyone should know to improve object retrieval," in *2012 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE), 2911–2918. doi: 10.1109/CVPR.2012.6248018

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). "SURF: speeded up robust features," in *Computer Vision – ECCV 2006. ECCV 2006* (Berlin, Heidelberg: Springer), 404–417. doi: 10.1007/11744023_32

Belgaid, M. C., Rouvoy, R., and Seinturier, L. (2019). *pyJoules: Python library that measures python code snippets* [computer software]. Available at: https://github.com/powerapi-ng/pyJoules (accessed January 6, 2025).

Berton, G., Masone, C., and Caputo, B. (2022). "Rethinking visual geo-localization for large-scale applications," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4878–4888. doi: 10.1109/CVPR52688.2022.00483

Berton, G., Trivigno, G., Caputo, B., and Masone, C. (2023). "Eigenplaces: training viewpoint robust models for visual place recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 11080–11090. doi: 10.1109/ICCV51070.2023.01017

Burkitt, A. N. (2006). A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biol. Cyber.* 95, 1–19. doi: 10.1007/s00422-006-0068-6

Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C., and Fua, P. (2011). Brief: computing a local binary descriptor very fast. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 1281–1298. doi: 10.1109/TPAMI.2011.222

Camara, L. G., and Přeučil, L. (2019). "Spatio-semantic convnet-based visual place recognition," in *2019 European Conference on Mobile Robots (ECMR)* (IEEE), 1–8. doi: 10.1109/ECMR.2019.8870948

Cao, B., Araujo, A., and Sim, J. (2020). "Unifying deep local and global features for image search," in *Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16* (Springer), 726–743. doi: 10.1007/978-3-030-58565-5_43

Che, K., Leng, L., Zhang, K., Zhang, J., Meng, Q., Cheng, J., et al. (2022). "Differentiable hierarchical and surrogate gradient search for spiking neural networks," in *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, 24975–24990.

Chen, Z., Jacobson, A., Sünderhauf, N., Upcroft, B., Liu, L., Shen, C., et al. (2017). "Deep learning features at scale for visual place recognition," in *2017 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 3223–3230. doi: 10.1109/ICRA.2017.7989366

Cieslewski, T., and Scaramuzza, D. (2017). "Efficient decentralized visual place recognition from full-image descriptors," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)* (IEEE), 78–82. doi: 10.1109/MRS.2017.8250934

Davies, M., Wild, A., Orchard, G., Sandamirskaya, Y., Guerra, G. A. F., Joshi, P., et al. (2021). Advancing neuromorphic computing with loihi: a survey of results and outlook. *Proc. IEEE* 109, 911–934. doi: 10.1109/JPROC.2021.3067593

DeTone, D., Malisiewicz, T., and Rabinovich, A. (2018). "Superpoint: self-supervised interest point detection and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 224–236. doi: 10.1109/CVPRW.2018.00060

DeWolf, T., Jaworski, P., and Eliasmith, C. (2020). Nengo and low-power ai hardware for robust, embedded neurorobotics. *Front. Neurorobot.* 14:568359. doi: 10.3389/fnbot.2020.568359

Doan, A.-D., Latif, Y., Chin, T.-J., Liu, Y., Do, T.-T., and Reid, I. (2019). "Scalable place recognition under appearance change for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9319–9328. doi: 10.1109/ICCV.2019.00941

Dube, R., Cramariuc, A., Dugas, D., Sommer, H., Dymczyk, M., Nieto, J., et al. (2020). Segmap: segment-based mapping and localization using data-driven descriptors. *Int. J. Rob. Res.* 39, 339–355. doi: 10.1177/0278364919863090

Garg, S., Fischer, T., and Milford, M. (2021). "Where is your place, visual place recognition?" in *Proceedings of the Thirtieth International Joint Conference on Artificial*

Intelligence (IJCAI-21) (International Joint Conferences on Artificial Intelligence), 4416–4425. doi: 10.24963/ijcai.2021/603

Garg, S., Suenderhauf, N., and Milford, M. (2022). Semantic-geometric visual place recognition: a new perspective for reconciling opposing views. *Int. J. Rob. Res.* 41, 573–598. doi: 10.1177/0278364919839761

Gehrig, M., Shrestha, S. B., Mouritzen, D., and Scaramuzza, D. (2020). "Event-based angular velocity regression with spiking networks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 4195–4202. doi: 10.1109/ICRA40945.2020.9197133

Halaly, R., and Ezra Tsur, E. (2023). Autonomous driving controllers with neuromorphic spiking neural networks. *Front. Neurorobot.* 17:1234962. doi: 10.3389/fnbot.2023.1234962

Hausler, S., Garg, S., Xu, M., Milford, M., and Fischer, T. (2021). "Patch-netvlad: multi-scale fusion of locally-global descriptors for place recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14141–14152. doi: 10.1109/CVPR46437.2021.01392

Hazan, A., and Ezra Tsur, E. (2022). Neuromorphic neural engineering framework-inspired online continuous learning with analog circuitry. *Appl. Sci.* 12:4528. doi: 10.3390/app12094528

He, K., Lu, Y., and Sclaroff, S. (2018). "Local descriptors optimized for average precision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 596–605. doi: 10.1109/CVPR.2018.00069

Hines, A. D., Stratton, P. G., Milford, M., and Fischer, T. (2024). "Vprtempo: a fast temporally encoded spiking neural network for visual place recognition," in *2024 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 10200–10207. doi: 10.1109/ICRA57147.2024.10610918

Hu, Y., Tang, H., and Pan, G. (2021). Spiking deep residual networks. *IEEE Trans. Neur. Netw. Learn. Syst.* 34, 5200–5205. doi: 10.1109/TNNLS.2021.3119238

Hunsberger, E., and Eliasmith, C. (2015). Spiking deep networks with lif neurons. *arXiv preprint arXiv:1510.08829*.

Hussaini, S., Milford, M., and Fischer, T. (2022). Spiking neural networks for visual place recognition via weighted neuronal assignments. *IEEE Robot. Autom. Lett.* 7, 4094–4101. doi: 10.1109/LRA.2022.3149030

Hussaini, S., Milford, M., and Fischer, T. (2023). "Ensembles of compact, region-specific regularized spiking neural networks for scalable place recognition," in *2023 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 4200–4207. doi: 10.1109/ICRA48891.2023.10160749

Johns, E., and Yang, G.-Z. (2011). "From images to scenes: compressing an image cluster into a single scene model for place recognition," in *2011 International Conference on Computer Vision* (IEEE), 874–881. doi: 10.1109/ICCV.2011.6126328

Kim, H. J., Dunn, E., and Frahm, J.-M. (2015). "Predicting good features for image geo-localization using per-bundle vlad," in *Proceedings of the IEEE International Conference on Computer Vision*, 1170–1178. doi: 10.1109/ICCV.2015.139

Kim, S., Park, S., Na, B., and Yoon, S. (2020). "Spiking-yolo: spiking neural network for energy-efficient object detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 11270–11277. doi: 10.1609/aaai.v34i07.6787

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 84–90. doi: 10.1145/3065386

Lajoie, P.-Y., and Beltrame, G. (2022). Self-supervised domain calibration and uncertainty estimation for place recognition. *IEEE Robot. Autom. Lett.* 8, 792–799. doi: 10.1109/LRA.2022.3232033

Lanham, M. (2018). *Learn ARCore-Fundamentals of Google ARCore: Learn to build augmented reality apps for Android, Unity, and the web with Google ARCore 1.0*. Birmingham: Packt Publishing Ltd.

Li, B., Munoz, J. P., Rong, X., Chen, Q., Xiao, J., Tian, Y., et al. (2018). Vision-based mobile indoor assistive navigation aid for blind people. *IEEE Trans. Mobile Comput.* 18, 702–714. doi: 10.1109/TMC.2018.2842751

Liu, Y., and Zhang, H. (2012). "Visual loop closure detection with a compact image descriptor," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE), 1051–1056. doi: 10.1109/IROS.2012.6386145

Loncomilla, P., Ruiz-del Solar, J., and Martínez, L. (2016). Object recognition using local invariant features for robotic applications: a survey. *Pattern Recognit.* 60, 499–514. doi: 10.1016/j.patcog.2016.05.021

Lowe, D. G. (1999). Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision* (IEEE), 1150–1157. doi: 10.1109/ICCV.1999.790410

Lowry, S., Sünderhauf, N., Newman, P., Leonard, J. J., Cox, D., Corke, P., et al. (2015). Visual place recognition: a survey. *IEEE Trans. Robot.* 32, 1–19. doi: 10.1109/TRO.2015.2496823

Lynen, S., Sattler, T., Bosse, M., Hesch, J. A., Pollefeys, M., and Siegwart, R. (2015). "Get out of my lab: large-scale, real-time visual-inertial localization," in *Robotics: Science and Systems*, 1. doi: 10.15607/RSS.2015.XI.037

Lynen, S., Zeisl, B., Aiger, D., Bosse, M., Hesch, J., Pollefeys, M., et al. (2020). Large-scale, real-time visual-inertial localization revisited. *Int. J. Rob. Res.* 39, 1061–1084. doi: 10.1177/0278364920931151

Maddern, W., Pascoe, G., Gadd, M., Barnes, D., Yeomans, B., and Newman, P. (2020). Real-time kinematic ground truth for the oxford robotcar dataset. *arXiv preprint arXiv:2002.10152*.

Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). "1 Year, 1000km: the Oxford RobotCar dataset. *Int. J. Robot. Res.* 36, 3–15. doi: 10.1177/0278364916679498

Masone, C., and Caputo, B. (2021). A survey on deep visual place recognition. *IEEE Access* 9, 19516–19547. doi: 10.1109/ACCESS.2021.3054937

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image Vis. Comput.* 22, 761–767. doi: 10.1016/j.imavis.2004.02.006

McManus, C., Upcroft, B., and Newman, P. (2014). Scene signatures: localised and point-less features for localisation. *Robotics* 10, 1–9. doi: 10.15607/RSS.2014.X.023

Mikolajczyk, K., and Schmid, C. (2002). "An affine invariant interest point detector," in *Computer Vision-ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part I 7* (Springer), 128–142. doi: 10.1007/3-540-47969-4_9

Naseer, T., Burgard, W., and Stachniss, C. (2018). Robust visual localization across seasons. *IEEE Trans. Robot.* 34, 289–302. doi: 10.1109/TRO.2017.2788045

Neftci, E. O., Mostafa, H., and Zenke, F. (2019). Surrogate gradient learning in spiking neural networks: bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Proc. Mag.* 36, 51–63. doi: 10.1109/MSP.2019.2931595

Olid, D., Fácil, J. M., and Civera, J. (2018). "Single-view place recognition under seasonal changes," in *PPNIV Workshop at IROS 2018*.

Oliva, A., and Torralba, A. (2006). Building the gist of a scene: the role of global image features in recognition. *Prog. Brain Res.* 155, 23–36. doi: 10.1016/S0079-6123(06)55002-2

Patel, D., Hazan, H., Saunders, D. J., Siegelmann, H. T., and Kozma, R. (2019). Improved robustness of reinforcement learning policies upon conversion to spiking neuronal network platforms applied to atari breakout game. *Neural Netw.* 120, 108–115. doi: 10.1016/j.neunet.2019.08.009

Perronnin, F., Liu, Y., Sánchez, J., and Poirier, H. (2010). "Large-scale image retrieval with compressed fisher vectors," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE), 3384–3391. doi: 10.1109/CVPR.2010.5540009

Radenović, F., Tolias, G., and Chum, O. (2018). Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 1655–1668. doi: 10.1109/TPAMI.2018.2846566

Rasmussen, D. (2018). NengoDL: Combining deep learning and neuromorphic modelling methods. *Neuroinformatics* 17, 611–628. doi: 10.1007/s12021-019-09424-z

Reinhardt, T. (2019). *Using Global Localization to Improve Navigation*. Available at: https://ai.googleblog.com/2019/02/using-global-localization-to-improve.html (accessed May 15, 2023).

Revaud, J., Almazán, J., Rezende, R. S., and de Souza, C. R. (2019). "Learning with average precision: training image retrieval with a listwise loss," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5107–5116. doi: 10.1109/ICCV.2019.00521

Rueckauer, B., Lungu, I.-A., Hu, Y., Pfeiffer, M., and Liu, S.-C. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Front. Neurosci.* 11:682. doi: 10.3389/fnins.2017.00682

Schönberger, J. L., Pollefeys, M., Geiger, A., and Sattler, T. (2018). "Semantic visual localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6896–6906. doi: 10.1109/CVPR.2018.00721

Seo, P. H., Weyand, T., Sim, J., and Han, B. (2018). "Cplanet: enhancing image geolocalization by combinatorial partitioning of maps," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 536–551. doi: 10.1007/978-3-030-01249-6_33

Shan, M., Wang, F., Lin, F., Gao, Z., Tang, Y. Z., and Chen, B. M. (2015). "Google map aided visual navigation for uavs in gps-denied environment," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (IEEE), 114–119. doi: 10.1109/ROBIO.2015.7418753

Siméoni, O., Avrithis, Y., and Chum, O. (2019). "Local features and visual words emerge in activations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11651–11660. doi: 10.1109/CVPR.2019.01192

Stratton, P. G., Wabnitz, A., Essam, C., Cheung, A., and Hamilton, T. J. (2022). Making a spiking net work: robust brain-like unsupervised machine learning. *arXiv preprint arXiv:2208.01204*.

Sünderhauf, N., Shirazi, S., Dayoub, F., Upcroft, B., and Milford, M. (2015). "On the performance of convnet features for place recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE), 4297–4304. doi: 10.1109/IROS.2015.7353986

Torralba, A., Fergus, R., and Weiss, Y. (2008). "Small codes and large image databases for recognition," in *2008 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE), 1–8. doi: 10.1109/CVPR.2008.4587633

Tsintotas, K. A., Bampis, L., and Gasteratos, A. (2022). The revisiting problem in simultaneous localization and mapping: a survey on visual loop closure detection. *IEEE Trans. Intell. Transp. Syst.* 23, 19929–19953. doi: 10.1109/TITS.2022.3175656

Uy, M. A., and Lee, G. H. (2018). "Point-netvlad: deep point cloud based retrieval for large-scale place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4470–4479. doi: 10.1109/CVPR.2018.00470

Vitale, A., Renner, A., Nauer, C., Scaramuzza, D., and Sandamirskaya, Y. (2021). "Event-driven vision and control for uavs on a neuromorphic chip," in *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE), 103–109. doi: 10.1109/ICRA48506.2021.9560881

Weyand, T., Kostrikov, I., and Philbin, J. (2016). "Planet-photo geolocation with convolutional neural networks," in *Computer Vision-ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII 14* (Springer), 37–55. doi: 10.1007/978-3-319-46484-8_3

Yin, P., Srivatsan, R. A., Chen, Y., Li, X., Zhang, H., Xu, L., et al. (2019). "Mrs-VPR: a multi-resolution sampling based global visual place recognition method," in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE), 7137–7142. doi: 10.1109/ICRA.2019.8793853

Yu, J., Zhu, C., Zhang, J., Huang, Q., and Tao, D. (2020). Spatial pyramid-enhanced netvlad with weighted triplet loss for place recognition. *IEEE Trans. Neur. Netw. Learn. Syst.* 31, 661–674. doi: 10.1109/TNNLS.2019.2908982

Zemene, E., Tesfaye, Y. T., Idrees, H., Prati, A., Pelillo, M., and Shah, M. (2018). Large-scale image geo-localization using dominant sets. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 148–161. doi: 10.1109/TPAMI.2017.2787132

Zhang, X., Wang, L., and Su, Y. (2021). Visual place recognition: a survey from deep learning perspective. *Pattern Recognit.* 113:107760. doi: 10.1016/j.patcog.2020.107760

Zhu, L., Mangan, M., and Webb, B. (2020). "Spatio-temporal memory for navigation in a mushroom body model," in *Biomimetic and Biohybrid Systems: 9th International Conference, Living Machines 2020, Freiburg, Germany, July 28–30, 2020, Proceedings 9* (Springer), 415–426. doi: 10.1007/978-3-030-64313-3_39

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 05 February 2025
DOI 10.3389/fnbot.2025.1478758

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Gang Chen,
Beijing University of Posts and Telecommunications (BUPT), China
Wenxin Mu,
Kunming University of Science and Technology, China

*CORRESPONDENCE
Ling-Li Zeng
✉ zengphd@nudt.edu.cn

RECEIVED 10 August 2024
ACCEPTED 20 January 2025
PUBLISHED 05 February 2025

CITATION
Zhao C, Yu Y, Ye Z, Tian Z, Zhang Y and Zeng L-L (2025) Universal slip detection of robotic hand with tactile sensing.
Front. Neurorobot. 19:1478758.
doi: 10.3389/fnbot.2025.1478758

Universal slip detection of robotic hand with tactile sensing

Chuangri Zhao, Yang Yu, Zeqi Ye, Ziyang Tian, Yifan Zhang and Ling-Li Zeng*

College of Intelligence Science and Technology, National University of Defense Technology, Changsha, China

Slip detection is to recognize whether an object remains stable during grasping, which can significantly enhance manipulation dexterity. In this study, we explore slip detection for five-finger robotic hands being capable of performing various grasp types, and detect slippage across all five fingers as a whole rather than concentrating on individual fingertips. First, we constructed a dataset collected during the grasping of common objects from daily life across six grasp types, comprising more than 200 k data points. Second, according to the principle of deep double descent, we designed a lightweight universal slip detection convolutional network for different grasp types (USDConvNet-DG) to classify grasp states (no-touch, slipping, and stable grasp). By combining frequency with time domain features, the network achieves a computation time of only 1.26 ms and an average accuracy of over 97% on both the validation and test datasets, demonstrating strong generalization capabilities. Furthermore, we validated the proposed USDConvNet-DG in real-time grasp force adjustment in real-world scenarios, showing that it can effectively improve the stability and reliability of robotic manipulation.

KEYWORDS

slip detection, five-finger robotic hand, deep learning, 3-axial force tactile sensor, grasp types

1 Introduction

The importance of tactile feedback has been emphasized by studies in human motor control, which show that stable object manipulation is difficult without this sensory input (Johansson and Vallbo, 1979). Tactile perception plays a crucial role in human object grasping. When slippage occurs, humans can promptly adjust their grip force and strategy to prevent the object from falling. This ability significantly enhances the flexibility and stability of object manipulation by the human hand (Johansson and Flanagan, 2009).

With the increasing application of robots in unstructured environments, they are required to perform more flexible manipulation tasks and achieve stable grasping, similar to humans (Chen et al., 2018). Although the accuracy and resolution of artificial tactile sensors still fall short of human tactile capabilities, they still play a significant role in improving grasping stability in robotic systems (Grover et al., 2022). They provide essential information about the interaction between the hand and the object, enabling quicker and more accurate slip detection than vision-based methods alone (Johansson and Westling, 1984; Westling and Johansson, 1984). Robots equipped with reliable tactile sensing can significantly improve their dexterous manipulation capabilities and achieve stable grasping of common objects (Cui et al., 2020). One of the most important dexterous robot manipulation tasks using the sense of touch is to detect or predict sliding while grasping a manipulated object. Slip detection is essential for ensuring stable robotic grasping, which is crucial for preventing objects from slipping or falling during manipulation. Detecting slip allows robotic systems to adjust their grasp strategies and

forces in real-time, ensuring that objects remain securely held (Yan et al., 2022; James and Lepora, 2021).

However, there are still some challenges. On the one hand, as sensor arrays become increasingly dense and sensing dimensions expand, traditional methods struggle to construct suitable models for detecting slippage. On the other hand, while previous research has made notable progress in slip detection for two\three-fingered robotic grippers (Chen et al., 2018; Romeo and Zollo, 2020), slip detection for five-fingered dexterous hands presents unique challenges because the complexity of grasp types that five-fingered hands can perform, as well as the need for algorithms that can generalize across a variety of object shapes, sizes, and materials.

In this study, we present a solution to the problem of slip detection in five-fingered robotic hands Five-finger robotic hand can perform a wide range of grasp types, each with unique contact dynamics, making it challenging to develop a one-size-fits-all solution. To address this challenge, we propose a Universal Slip Detection Framework for Different Grasp Types (USDFrame-DG), designed to handle the complexities associated with various grasp types and object properties. In summary, the main contributions of this work are as follows:

(1) According to the reference document (Feix et al., 2016), six common and significantly different grasp types were selected, as shown in Figure 1. A large amount of grasp state data (no-touch, slip, no-slip) was collected during these six grasp types. The 16 objects used for grasping, as shown in Figure 2, are made from materials commonly found in daily life, such as plastic, steel, and wood.

(2) A novel universal slip detection framework (USDFrame-DG) was proposed, focusing on efficiently collecting large-scale datasets and combing the frequency with time domains to achieve improved recognition performance.

(3) To validate which network architecture is better suited to address this problem, we compared four classic classification methods: Support Vector Machine (SVM), Long Short-Term Memory (LSTM) network, Residual Neural Network (ResNet), and Transformer. According to the results of the comparison, a lightweight and efficient USDConvNet-DG was designed, achieving more than 97% accuracy on both the validation and test sets. This capability highlights the universality and generalization of the proposed framework.

(4) We evaluated the performance of different methods, the contribution of various grasp types, and the performance of USDConvNet-DG trained with different numbers of grasp types. Additionally, we developed a physical demonstration system to showcase network's ability to detect slip in real-time, as shown in Figure 1. Furthermore, we increased the object's weight after achieving a stable grasp to verify whether the system can adjust the grasping force in real-time. Video demonstrations have been uploaded to the GitHub repository and are available at https://github.com/sunshine486/show.

2 Related works

Existing methods for detecting slippage during grasping can be divided into two categories: (1) analysis-based methods and (2) learning-based methods. Analysis-based methods typically identify grasp states using two key features: frequency and friction. Learning-based methods usually involve collecting data on slip and no-slip states to train a classification model.

These are some representative works based on changes in friction force. The first, proposed by Claudio Melchiorri, detects slippage by comparing the ratio of friction force to grasp force with the coefficient of friction (Melchiorri, 2000). The second, introduced by Beccai et al. (2008), utilizes friction cones to achieve slip detection, but with a delay exceeding 20 ms both methods operate on similar principles. Another approach, proposed by Song and Liu, employs the Break-Away Friction Ratio (BF-ratio) to predict slippage during the grasping process (Song et al., 2013). Although this method completes the prediction within just 4.2 ms, it requires 5–7 s to determine the

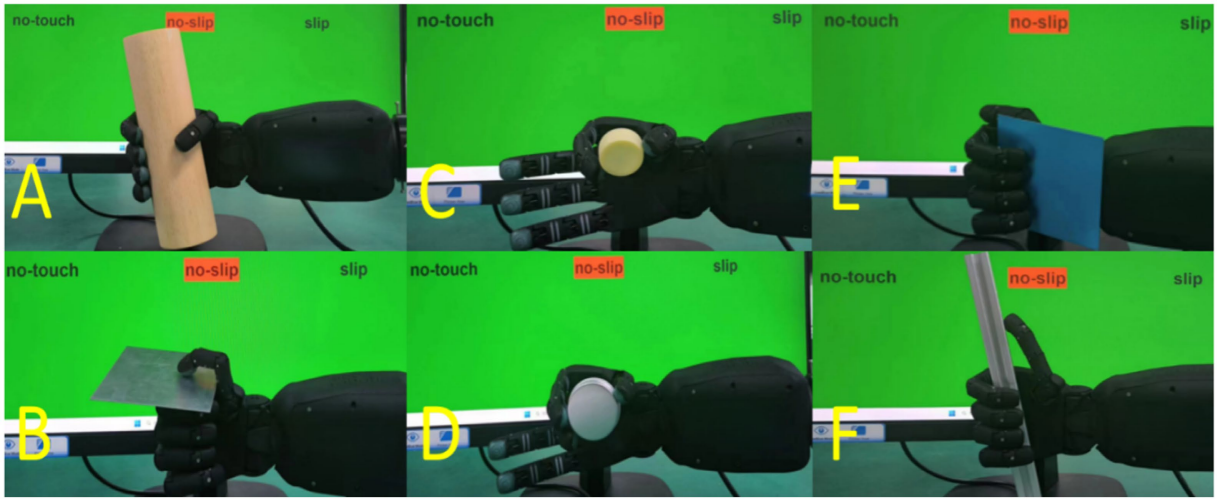


FIGURE 1
Grasp types and grasp state visualization. USDConvNet-DG trained for application in real-world scenarios.

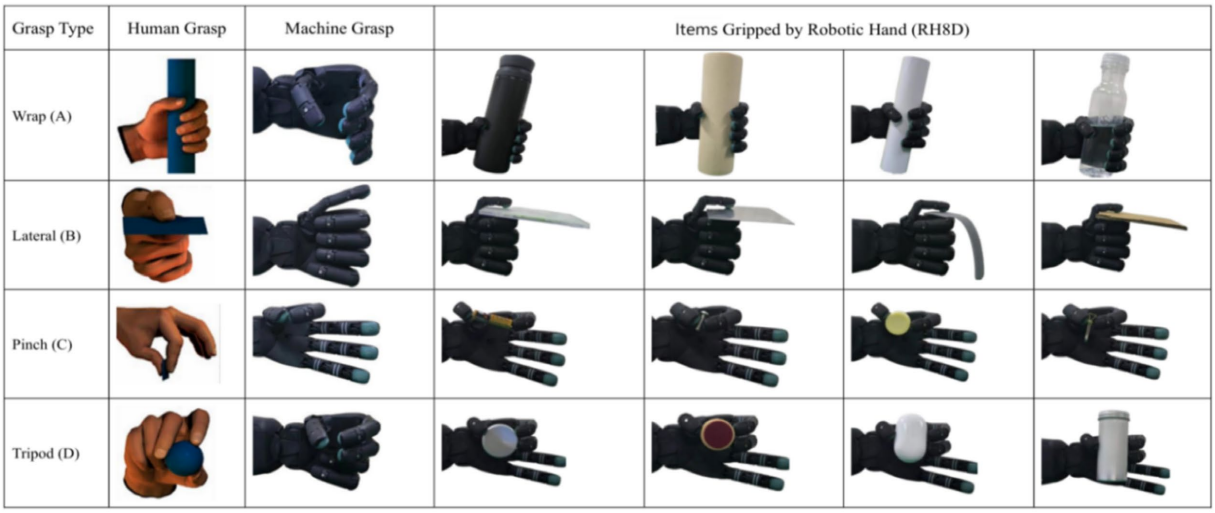


FIGURE 2
Grasp types and items for training the model. The first column represents the grasp type names. The second column indicates the grasping type for a human hand. The third column represents the corresponding grasping posture for the robotic hand. Columns 4, 5, 6, and 7 depict the items grasped for the corresponding grasp types.

friction coefficient through haptic surface exploration and has been validated in only three scenarios.

The signal spectra of slipping and non-slipping states are significantly different (Zhang et al., 2016). Specifically, when the grasp is stable, the signal primarily consists of low-frequency components; however, during slipping, the signal shifts to higher frequencies. Holweg et al. (1996) noted that the normal forces measured by the tactile sensor fluctuate at a certain frequency during slip due to the elasticity of rubber. Techniques such as Discrete Wavelet Transform (DWT) (Shensa, 1992) and Fast Fourier Transform (FFT) (Duhamel and Vetterli, 1990) have been employed to detect slip vibrations during robotic grasping. DWT is typically used for filtering, followed by a manually defined threshold to distinguish between slip and no-slip states (Zhang et al., 2016; Deng et al., 2017), making it more suitable for analysis-based methods. Zeng et al. (2022) utilized Discrete Wavelet Transform (DWT) to extract high-frequency signals, which were then compared against predefined thresholds to achieve slip detection. Similarly, Yang and Wu (2021) divided the slipping process into two phases: the initial slip phase and the slip suppression phase, with detection thresholds estimated separately for each phase. Both studies were conducted using a prosthetic hand. It is worth noting that Romeo et al. achieved slip detection at the hardware level using filters and on–off circuits (Romeo et al., 2021), which provides higher integration. However, adjusting thresholds and filters requires replacing components such as inductors and capacitors, making it challenging for non-technical users.

Analysis-based methods for slip detection generally rely on single touch areas, which overlook the spatial characteristics of different fingertips and the variations in touch areas caused by different grasp types. The slip detection performance of these methods is highly dependent on specific touch conditions. Consequently, parameters such as thresholds and filters lack generalization when applied to new contact scenarios introduced by a wide range of objects (Cui et al., 2024). Moreover, manually setting these parameters is time-consuming and cumbersome, requiring a certain level of engineering expertise.

In learning-based methods, slip detection is commonly formulated as a binary classification problem (slip/non-slip). With the rapid advancements in machine learning and the growing diversity of tactile sensors, machine learning techniques have been increasingly applied to slip detection, resulting in impressive outcomes.

In the field of machine-learning-based slip detection, the work of James and Lepora (2021) is particularly noteworthy. They utilized a sensor array to calculate the rate of change of pin positions per frame and compared three distinct binary classifiers (Threshold Classifier, SVMs, and Logistic Regression), achieving promising results in real-world scenarios.

In previous research, most studies are based on two-finger grippers and use LSTM network. Zhang et al. (2018) developed a novel optical-based tactile sensor (FingerVision), and proposed a sliding classification framework based on ConvLSTM (Convolutional Long Short-Term Memory) networks. Begalinova et al. (2022) employed an LSTM model trained on low-cost tactile sensors and evaluated the model using a two-finger gripper. Xie et al. (2023) employs LSTM networks for sliding detection and found that robotic grasping with slip detection has a success rate nearly 15% higher than grasping without slip detection. Fiedler et al. (2023) utilizes sliding detection based on a two-finger gripper to achieve grasping of textile objects. Yan et al. (2022) employed multimodal machine learning, combining visual and tactile information using a convolutional neural network-temporal convolutional neural network (CNN-TCN), achieving a detection accuracy of 88.7% for sliding detection with a two-finger gripper. James et al. (2018) used the TacTip sensor and Support Vector Machine (SVM) algorithm to classify sliding and stationary states, achieving an accuracy of 99.88%. However, this result was obtained only in structured environments, and the actual performance was not tested.

In addition, there are some studies based on five-finger robotic hands, but they have only achieved slip detection for a single grasp type. Zapata-Impata et al. (2019) utilized ConvLSTM to detect the direction of object sliding on the fingertip. The sensors used in the

papers are BioTac, which is very expensive. Mi et al. (2021) propose two novel methods based on Graph Convolution Network (GCN) for robotic stability classification. Grover et al. (2022) train a temporal convolution neural network (TCN) to detect slip that achieves an accuracy of over 91% on average on validation dataset. These two methods are based on three-finger grippers. Garcia-Garcia et al. (2019) constructed a graph neural network to predict the stability of grasping, but their work was based on three fingers. Deng et al. (2020) utilized sliding detection based on LSTM networks as feedback to control the grasping force.

The above studies demonstrate the effectiveness and robustness of learning-based slip detection methods utilizing tactile sensing. However, there remain several challenges in this field, as outlined below:

(1) Traditional analysis-based methods require manual adjustment of thresholds and filters, which is not only time-consuming and cumbersome but also demands a certain level of engineering expertise.
(2) Tactile sensors are becoming denser arrays, capable of perceiving multi-dimensional forces and more diverse sensing modalities. Analysis-based methods struggle to construct suitable mathematical models to handle this complexity.
(3) Previous learning-based studies have primarily focused on grippers or two−/three-finger robotic hand platforms, which are limited to a single grasping style. In contrast, five-finger dexterous hands are capable of performing a wide range of grasp types, making slip detection significantly more complex. As shown in Table 1, models trained solely on state data from a single grasp type exhibit poor performance in detecting slips for other grasp types, indicating a lack of generalization capability.
(4) Slip detection for five-finger robotic hands usually detect slippage in individual fingertip regions. This study treats across all five fingers as a whole for slip detection. However, this approach lacks sufficient datasets and requires further exploration of suitable network architectures.

In this study, we focus on universal slip detection for different grasp types. Inspired by prior work and integrating analysis-based and learning-based methods, we propose a novel slip detection framework and network.

3 Method

To achieve universal slip detection across different grasp types, we propose a general slip detection framework: USDFrame-DG, as shown in Figure 3. The framework consists of four key components: Grasp Force Control Module, Data Collection for Six Grasp Types, Data Preprocessing, and Model Training, each of which will be detailed below. Over 200 k data samples covering slipping, stable grasping, and non-touch states were collected to train the models. The dataset for slipping and stable grasping states was collected using various grasp types and everyday objects, ensuring the model's applicability to real-world scenarios.

TABLE 1 Accuracy of USDConvNet-DG trained with varying numbers of grasp types.

Grasp types	1	2	3	4
Training dataset	A, B, C, D	AB, AC, AD, BC, BD, CD	ABC, ABD, ADC, BCD	ABCD
Accuracy	45.1 ± 6.12%	80.7 ± 6.43%	93.7 ± 2.92%	95.7 ± 2.41%

3.1 Hardware setting

The model of five-finger robotic hand used in our experiments is RH8D, designed by Seed Robotics, as shown in Figure 4. Inspired by the human hand, it is capable of performing essential grasp types, featuring tendon-driven mechanisms and underactuated design. The RH8D can be mounted at the end of a six-degree-of-freedom robotic arm and features 19 degrees of freedom, including an opposable thumb and a full spherical wrist joint. It's three-segment fingers are powered by smart actuators housed entirely within the unit, offering payload capabilities (750 g in 3D space and 2.5 kg vertical pull). Inspired by the human hand, the RH8D provides advanced sensing and data acquisition on all actuated joints, including real-time feedback on position, speed, current, and PWM output. Additional features include a palm Time of Flight (ToF) distance sensor, optional capacitive touchpads for enhanced human-robot interaction, and reinforced design elements like Dyneema tendons and magnetic detachment for durability.

The Fingertips Tactile Sensors (FTS) used in this study are 3-axis force sensors designed for precise force measurement, as shown in Figure 4. These sensors measure forces along the X, Y, and Z axes and are optimized for forces in the 0–10 N range, offering a resolution of 1mN. For higher forces (10–30 N), an extended range model is used, with a resolution of 10mN in this range. The sensors operate with a sampling frequency of 50 Hz and have an overload capability of up to 50 N. Additionally, there is a 20mN offset when the sensors do not touch objects. The FTS works via an array of MEMS (Micro-Electromechanical System) sensors, which are highly resistant to magnetic field interference. Noise levels are minimal (on the order of millinewtons), making the sensors well-suited for practical applications. The sensors are pre-calibrated and exhibit linear performance in the typical force range of 10°–30° and beyond. While fast temperature changes may cause slight drift (up to 100mN in extreme conditions), these effects are generally negligible in most scenarios. For more technical details on the sensor specifications and design, we refer readers to the Seed Robotics documentation and related resources.

3.2 Grasp types

The five-finger robotic hand offers a higher degree of freedom compared to two-finger and three-finger grippers, allowing it to generate many more grasp types. Feix et al. (2016) summarized 33 common grasp types used by humans, which can be grouped into six categories. When considering only hand configuration, without taking into account object shape or size, these 33 grasp types can be reduced to 17 more general types. Although the RH8D features 19 degrees of

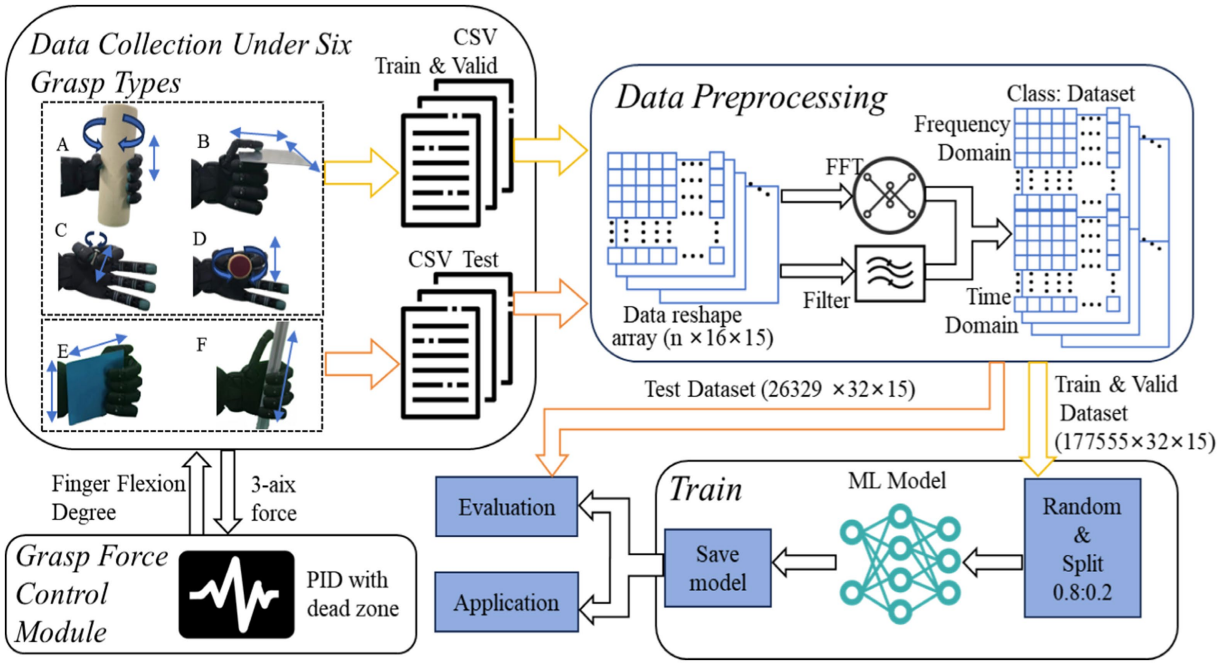


FIGURE 3
USD Frame-DG: universal slip detection framework for different grasp types.

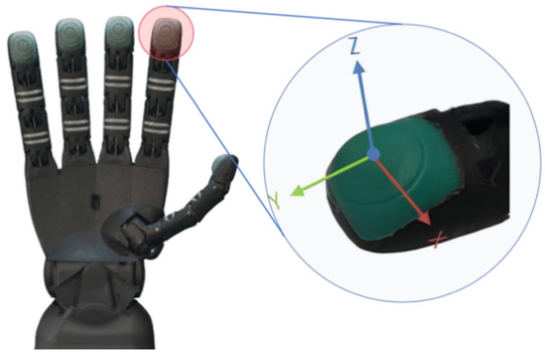


FIGURE 4
RH8D adult robot hand and FTS-3.

freedom, human hands possess 27 degrees of freedom (Agur and Dalley II, 2023), meaning the RH8D is unable to perform all the grasp types like human.

We initially collected slip and non-slip data for one grasp type and used this data to train a USDConvNet-DG model. The recognition accuracy exceeded 96% for the trained grasp type (A), but dropped below 70% for another grasp type (E). Although the model showed some generalization ability, its accuracy was insufficient for adjusting grasp force and strategy. Surprisingly, we discovered that it wasn't necessary to collect sliding data for all 33 grasp types. By gathering data for a few significantly different grasp types, the model could generalize effectively to other grasp types.

In the end, we selected four significantly different grasp types that the robotic hand could perform, as shown in Figure 2. These four types are suited for various scenarios: "Wrap (A)" for grasping long and large objects, "Lateral (B)" for flat objects, "Pinch (C)" for small and delicate objects, and "Tripod (D)" for smaller objects. The remaining two grasp types (E and F) are used to test generalization.

In our experiment, we chose 16 common items to collect grasping data, as shown in Figure 2. The weight of these objects ranged from 10 g to 300 g, and the materials included plastic, metal, wood, paper, and other commonly encountered substances.

3.3 Grasp force control

A PID (Proportional Integral Derivative) controller with a dead zone is used to control the robotic hand's grasping force. The grasping force of each finger can be controlled individually. As shown in the Figure 5. $f_I(i)$ represents the aim grasping force, and $f_p(i)$ represents the synthesis of the three-directional force detected by the FTS, calculated as follows:

$$f_p(i) = \sqrt{f_x^2 + f_y^2 + f_z^2}$$

The difference between $f_I(i)$ and $f_p(i)$ is denoted as $e_r(i)$. To prevent oscillation of the robotic hand during grasping, the range of change in $e_r(i)$ needs to be limited, as shown in the following formula. After multiple tests, setting the threshold *thr* to 150mN was the most suitable.

$$e(i) = \begin{cases} 0.01 e_r(i) & if \ \ e_r(i) < thr \\ e_r(i) & else \end{cases}$$

The value u is obtained from the PID controller, which represents the flexion degree of each finger (range: 0–4,095). The formula is as follows:

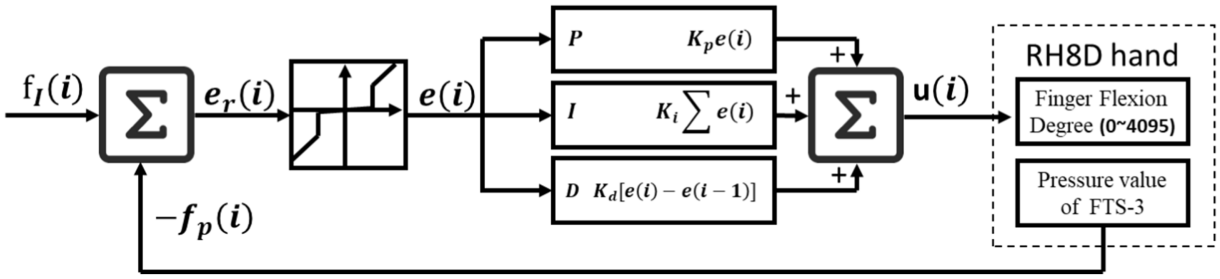


FIGURE 5
Grasp force control module.

$$u(i) = K_p e(i) + K_i \sum e(i) + K_d \left[e(i) - e(i-1) \right]$$

After extensive testing, the most suitable values for K_p, K_i and K_d were found to be 0.4, 0.04, and 0.5, respectively.

3.4 Data collection

In this work, slip detection is treated as a classification problem with three categories: no-touch, no-slip, and slipping. Deep learning methods rely on large amounts of data. It is easy to collect data for the no-touch and no-slip states, but collecting enough data for the slipping state is challenging because it occurs in an instant. Data from the FTS is directly saved as no-touch when no object is being grasped. During stable grasps of six types, the collected data belongs to the no-slip category. We tried two methods to deal with the challenge of collecting slipping data.

The first method involves slowly pulling out the object after the robotic hand has stably grasped it. Approximately 2000 data points can be collected within 5 s when the sampling rate is set to 50 Hz. Although training a network with this data results in high accuracy on the validation set, its performance on the test set and in real-world applications is poor. Through continuous reflection and analysis, we found that the abrupt change in force during slipping is the key feature.

To capture this feature, we proposed another data collection method: after the robotic hand grasps the object, an external force is applied by hand to move the object back and forth quickly in the direction shown in Figure 1. This allows for the rapid collection of a large amount of slipping data, making it possible to use neural network-based classification methods. The final test results demonstrated significant improvements. We believe that this method can also be used to quickly collect a large amount of effective slipping data for robotic hands equipped with other types of sensors. For grasp types A, B, C, and D, the data is used for training and validation, while grasp types E and F are used for testing to evaluate the generalization of the detection model.

3.5 Data preprocessing

Each data record comprises 15 measurements, corresponding to the force components along three axes (X, Y, Z) for each of the five fingers. Noise removal from the dataset is manually performed, with particular attention given to the initial and final segments of the data sequences. To maintain sample balance, excessively long sequences are trimmed. Once processed, the data is ready to construct the training and test sets. The final dataset includes over 200,000 scalar data points sampled at a frequency of 50 Hz.

Since the collected data represents time series information with inherent periodicity and autocorrelation characteristics, training the model using a single data record results in suboptimal performance. Instead, combining multiple adjacent data records into a single sample is more effective, as it enables the system to observe force variations over a period of time, which is crucial for detecting slippage. However, using an excessively long observation period compromises real-time performance. After conducting extensive tests, we found that using a stride of 1 and combining 16 adjacent data records into a 16 × 15 array yields the best practical results. For example, if 2000 data points are collected in one session, the first 16 records form the first array, the second to the 17th records form the second array, the third to the 18th records form the third array, and so on, until the final 16 records form the last array. This structure also facilitates the application of FFT analysis.

In the collected slipping dataset, a small portion of noise is difficult to manually remove, which can significantly affect the trained model. A high-pass filter is used to preprocess the slipping data because the frequency of the slip signal is higher. The calculation formula is as follows:

$$y(i) = 0.2x(i) - 0.8y(i-1)$$

$x(i)$ represents the i-th array. By applying a filtering method, the model's accuracy improved to a certain extent. Since an object generates vibrations during slipping, there is a distinctive spectral distribution in the frequency domain that can be used as a feature for training the model. A Fast Fourier Transform (FFT) is applied individually to each column of the data, resulting in a 16 × 15 matrix. This matrix is then combined with the filtered time-domain matrix, producing a 32 × 15 matrix where the first 16 rows represent the frequency domain, and the last 16 rows represent the time domain.

Labels in a one-hot format are assigned based on the grasping states: [1,0,0] for no-slip, [0,1,0] for slip, and [0,0,1] for no-touch. A total of 177,555 matrices (A, B, C, and D) are randomly divided into training and validation sets in an 80%:20% ratio, while 26,329 matrices (E and F) were reserved for testing, as shown in Table 2. The ratio of

TABLE 2 Class distribution of grasp states across different grasp types.

Grasp type	A	B	C	D	E	F	Total
Slip	15,448	16,512	16,752	16,137	6,643	6,587	78,079
No-slip	15,948	15,538	15,987	16,392	6,513	6,586	76,964
No-touch	48,841						48,841

the three classes—no-touch, slip, and no-slip—is approximately 24%:38%:38%.

3.6 Network architecture

For slip detection, different types of sensors generate different data types, so there is no single model that fits all sensors. To address this, we designed four three-classification algorithms based on four classical models (SVM, LSTM, Residual Convolutional Neural Network and Transformer). These three categories are sliding, non-sliding, and no-touch. Below, we will describe these four models in detail.

Based on the SVM (Cortes and Vapnik, 1995) method: As shown in the Figure 6A, two support vector machines were trained to achieve the three classifications of "no-touch" "no-slip" and "slip." Because both "no slip" and "slip" indicate contact with an object, these two categories belong to "touch." Therefore, the first support vector machine is used to recognize whether there is contact ("touch"), the second support vector machine detects sliding within the "touch" category.

Based on the LSTM (Hochreiter and Schmidhuber, 1997) model, as shown in Figure 6B: Inputting a 16×15 array, it passes through an LSTM network, a flattening layer, two fully connected layers, and finally outputs the probabilities of belonging to each category. The hidden layer dimension and the number of recurrent neural network layers in the LSTM network are both set to 10. During testing, we observed that increasing the size of the LSTM network initially increased the classification accuracy, but then decreased. The best performance was achieved when the hidden layer dimension and the number of recurrent neural network layers reached 10. However, when the number of layers reached 50, the accuracy dropped to 56%.

Based on the ResNet18 (He et al., 2016) model, as shown in Figure 6C: Compared to the standard ResNet18, the number of input channels in the first convolutional layer has been reduced from three to one, and the output dimensions of the final fully connected layer have been adjusted from 1,000 to 3 to match the classification task. The rest of the architecture remains unchanged.

Based on the Transformer (Vaswani et al., 2017) model, as shown in Figure 6D: The input to the model is a 16×15 array. An average pooling layer and a fully connected layer are added after the Transformer. The best performance is achieved when both the encoder and decoder consisting of a single layer.

Although ResNet18 achieved over 99% accuracy on the validation set, its accuracy just reached 70% on the test set, which is unacceptable for practical applications. ResNet18 has over 10 million parameters, which does not match the scale of our training dataset. Therefore, as shown in Figure 7, we designed USDConvNet-DG based on the design principles of ResNet18:

(1) Residual connections: these connections help mitigate the vanishing gradient problem in deep networks, allowing more efficient gradient flow and facilitating the training of deeper architectures.
(2) Hierarchical feature extraction: ResNet18 employs a progressively deeper hierarchical structure, extracting features from lower to higher levels through multiple convolutional layers. Similarly, USDConvNet-DG adopts a block-based design, where each block consists of multiple convolutional operations, enabling finer feature extraction while enhancing the network's representation capacity.
(3) Batch normalization (BN): USDConvNet-DG retains BN layers after each convolution, standardizing data distribution to accelerate convergence, reduce the risk of gradient vanishing, and stabilize the training process for slip detection.
(4) Multi-scale feature integration: ResNet18 integrates multi-scale features through residual blocks and layer-wise feature extraction. USDConvNet-DG combines multi-layer convolution and residual connections to effectively extract multi-scale features across different grasp types and contact states, improving performance in slip detection tasks.

The network takes a 2D input, which is processed by a series of convolutional layers. The first layer is a Conv2d (2D convolution) followed by Batch Normalization and a ReLU activation function being followed by MaxPooling, which reduces the spatial dimensions of the feature map. After multiple tests, we found that four blocks are the most suitable. Each block consists of two convolutional layers (Conv2d) with Batch Normalization. The blocks represent different levels of feature extraction with increasing depth, and contributing to a more complex and rich feature representation. The feature map is then flattened and passed through a fully connected layer (FullConnection), which helps in classification. The final layer outputs one of the three categories: slip, no-slip, or no-touch. USDConvNet-DG achieved a maximum accuracy of 97% on the test set.

3.7 Training

Furthermore, all tactile sensing, slip detection networks, and robotic five-finger hand control algorithms are executed on a PC equipped with an Intel Core i7-12700K processor (3.60 GHz, 12 cores, 20 threads) and an NVIDIA RTX 3080 Ti GPU. The codes are implemented using PyTorch and Python, running on the Windows 11 operating system.

4 Results

This section primarily discusses related test results based on different methods. The result is based on four trained grasping gestures. Overall, the recognition accuracy for the "no-touch" state is higher than other two categories. The classification performance of the method based on USDConvNet-DG is the best, while the performance of SVM method is poorest.

The performance of different models as shown in Table 3, which provides a detailed comparison in terms of accuracy on the validation

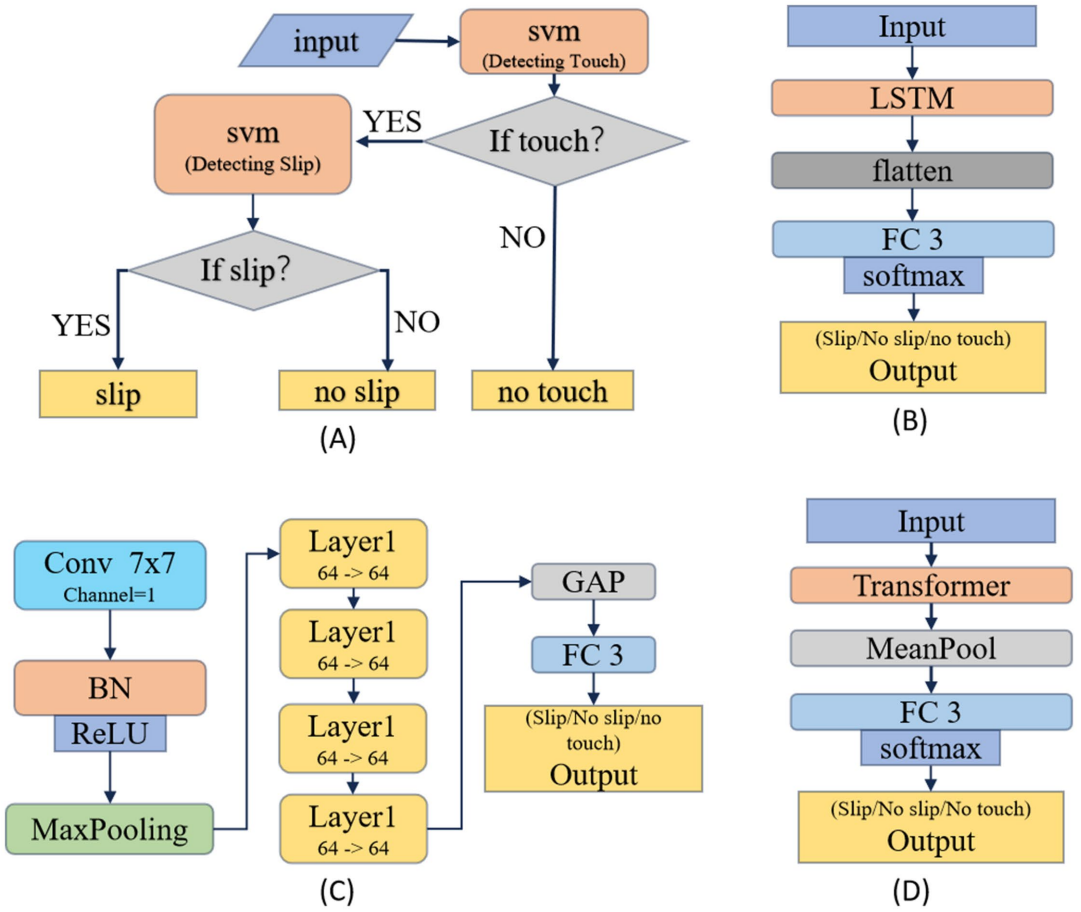


FIGURE 6
Four classical architectures for slip detection models: **(A)** based on SVM model, **(B)** based on LSTM model, **(C)** based on ResNet model, **(D)** based on Transformer model.

dataset (grasp types A, B, C, D) and the test dataset (grasp types E and F), prediction time, and the number of parameters. We selected the highest accuracy from the 24 epochs, then averaged and calculated the standard deviation of the ten accuracy values. Given the high demand for real-time performance in slip detection, we also tested the prediction time of various methods.

The SVM-based classification method had the shortest prediction time, only 0.08 ms, but with low accuracy. When the number of parameters reaches the scale of 10 million, the LSTM and Transformer models achieve approximately 63% accuracy on the validation dataset and 43% on the test dataset, which is about 30% lower than the accuracy of ResNet18. Particularly, the prediction time of the Transformer exceeds 129.55 ms, which is unacceptable for real-time tasks. Additionally, both LSTM and Transformer exhibit slow convergence. The original LSTM lacks residual connections, so multiple LSTM layers can lead to gradient vanishing issues, making it difficult to converge. Moreover, slip detection primarily focuses on local changes in force tactile data, such as short-term high-frequency features. While the self-attention mechanism of the Transformer is applied to capture global long-range dependencies, this capability may not align well with the requirements of slip detection tasks. The complexity of the Transformer may introduce unnecessary computational overhead, whereas convolutional networks are more straightforward and effective for this application.

The ResNet-based classification method has very high accuracy on the validation dataset, but its prediction time is the longest, with over 10 million parameters, making its scale too large to be conveniently integrated into a robotic hand. Thus, we attempted to decrease the number of parameters for ResNet network. We found that the accuracy on the validation dataset decreased by less than 1% when the parameter exceeded 40 k. However, reducing the parameters further resulted in a more pronounced decrease, with accuracy dropping by more than 5%. Specifically, when the parameters are reduced to approximately 2 k, the accuracy on the training dataset decreased by around 4%, but the accuracy on the test dataset improve to 77.38%. These findings suggest that a smaller parameter count may enhance generalization on the test dataset, though it slightly compromises performance on the training and validation datasets. This phenomenon is known as "DEEP DOUBLE DESCENT," which is common in ResNet and convolutional networks (Nakkiran et al., 2021).

The combination of FFT and filtering with USDConvNet-DG yields the best overall results, with over 97% accuracy on validation dataset and test dataset. These results provide stronger evidence that the network demonstrates robust generalization across diverse grasp types, not limited to the initially trained or tested categories. This model maintains a short prediction time (1.26 ms) and the same low

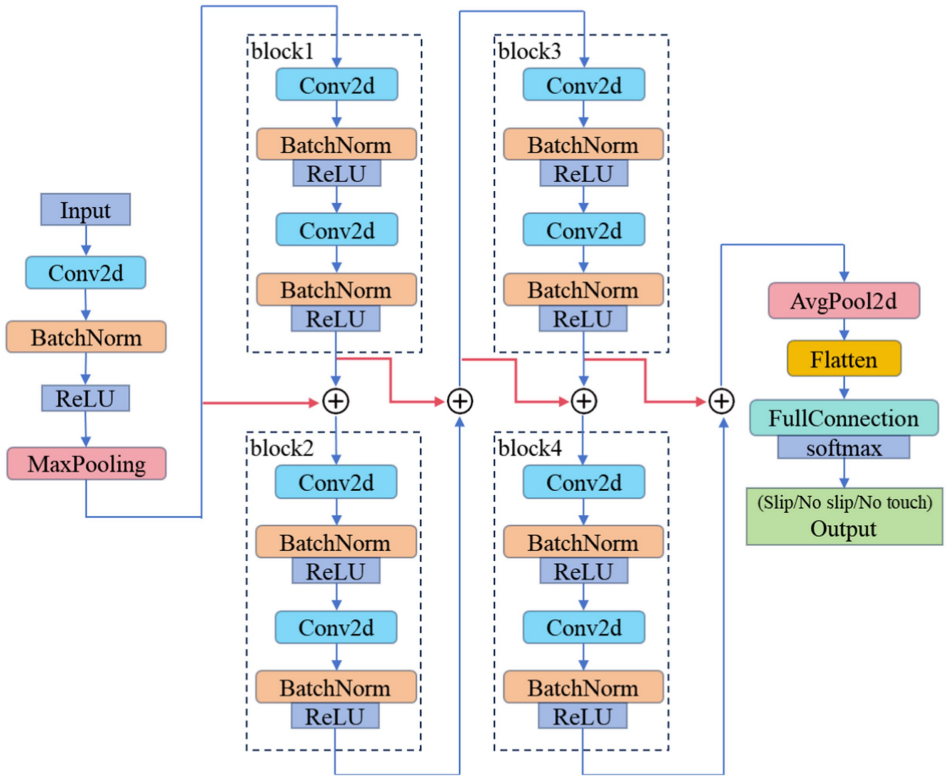


FIGURE 7
USD ConvNet-DG: universal slip detection convolutional network for different grasp types.

TABLE 3 Quantitative comparison of different methods

Model	Accuracy		Prediction time	Parameters
	Validation dataset	Test dataset		
SVM	62.81 ± 1.03%	51.34 ± 3.24%	0.08 ms	<100
LSTM	91.24 ± 1.40%	65.69 ± 4.85%	1.30 ms	10,163
	63.38%	42.85%	12.26 ms	10,630,403
Transformer	96.09 ± 0.89%	68.38 ± 2.35%	1.57 ms	129,499
	63.33%	42.76%	129.55 ms	11,034,156
ResNet	99.67 ± 0.06%	78.35 ± 5.27%	2.66 ms	11,171,779
	99.84 ± 0.06%	75.10 ± 5.50%	2.12 ms	709,155
	99.14 ± 0.20%	72.71 ± 6.72%	1.66 ms	47,499
	95.75 ± 0.67%	77.38 ± 10.54%	1.23 ms	2,305
ResNet18 + FFT + Filter	99.02 ± 0.11%	97.09 ± 1.40%	2.66 ms	11,171,779
USDConvNet-DG	97.07 ± 0.18%	86.46 ± 9.58%	1.26 ms	2,395
USDConvNet-DG + FFT	97.78 ± 0.20%	96.65 ± 1.34%	1.26 ms	2,395
USDConvNet-DG + Filter	97.02 ± 0.32%	89.62 ± 4.25%	1.26 ms	2,395
USDConvNet-DG + FFT + Filter	97.71 ± 0.29%	97.12 ± 1.08%	1.26 ms	2,395

number of parameters (2,395), making it the most effective and efficient model among those tested.

Figure 8 presents the performance of different models on the validation and test dataset. The epoch was set to 24. The six methods are tested ten times, and the accuracy of each epoch was averaged to capture the overall trend. The following observations can be made:

The accuracy of all models is higher on the validation set than on the test set, and exceeds 90%. For validation, LSTM performs worst, and the accuracy of the Transformer gradually increases to around 90% as the number of epochs increases. However, both models only achieve about 60% accuracy on the test dataset, showing weak generalization in this problem and struggling to generalize well to

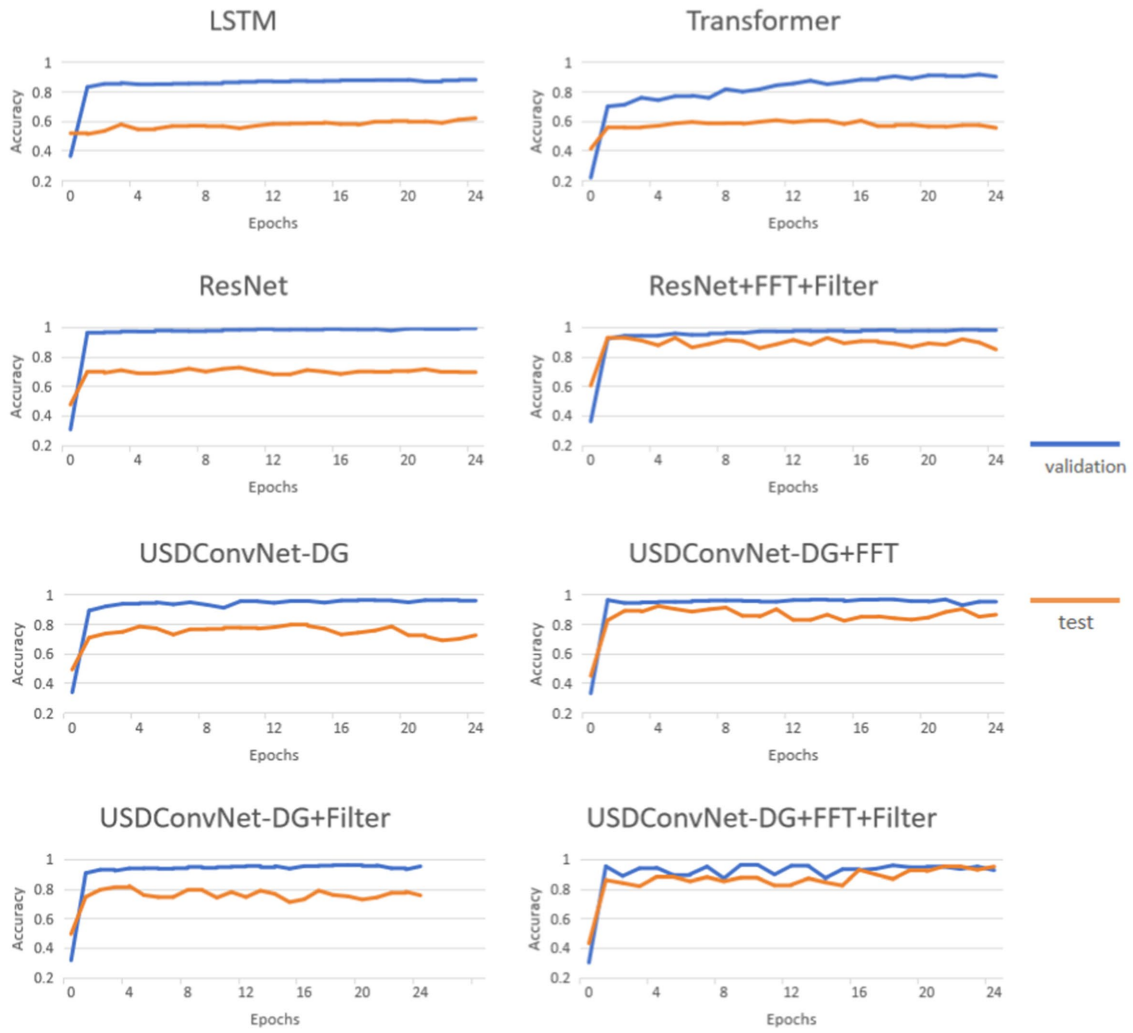


FIGURE 8
Performance of learning-based method during training.

untrained grasp types. ResNet shows the highest accuracy on the validation set, and its accuracy on the test set is about 10% higher. The USDConvNet-DG we propose performs slightly lower than ResNet on the validation set, but it outperforms ResNet on the test set. When applying FFT and filtering to ResNet, the validation accuracy remains consistently high, and the test accuracy improves compared to using ResNet alone.

When FFT is used to preprocess the training data, the test accuracy of USDConvNet-DG improves significantly. The improvement is relatively smaller with filtered data. Overall, combining FFT and filtering with USDConvNet-DG results in the most stable and high accuracy on the test dataset, closely approaching the validation accuracy. This model appears to effectively balance feature extraction and generalization, making it the most robust among the tested configurations.

It is worth exploring whether the data collected from different grasp types contributes differently to universal slip detection. Therefore, we designed a controlled experiment as follows: five of the six grasp types were used to train the model, and the remaining one was used to test the model to obtain the accuracy. The test results for the six grasp types are shown in Table 4. The accuracy is lower when Type A is not included in the training set, indicating that Type A contributes more to universal slip detection.

Table 1 shows the accuracy of USDConvNet-DG with varying numbers of grasp types. The test dataset consists of Grasp Types E and F, and the number of epochs is set to 20. As the grasp type is 1, the model was trained separately on the four training sets (A, B, C, D). The test was repeated five times. Finally, the average and standard deviation of the 20 accuracy results were calculated. When the grasp type is 2, the training dataset is a combination of two grasp types. The model was trained separately on the six training sets (AB, AC, AD, BC, BD, CD), and the accuracy improved significantly. When the grasp type is 3, the training dataset consists of three grasp types. When the grasp type is 4, all four grasp types together form a single training dataset, and the improvement in accuracy is minimal. Overall, with the number of grasp types increases, the accuracy on the test dataset improves.

To test the effectiveness of recognizing tactile events locally (i.e., per fingertip), we trained USDConvNet-DG using individual sensor data (3×16 arrays). Each fingertip was independently detected whether slippage occurred. If any one of the five fingertips detected slippage, the

TABLE 4 The model's accuracy on test datasets with different grasp types.

Model	Accuracy on single dataset					
	Type A	Type B	Type C	Type D	Type E	Type F
LSTM	56.73%	67.73%	65.35%	68.28%	66.31%	64.39%
Transformer	60.89%	69.20%	66.16%	73.91%	70.57%	69.84%
ResNet18	70.98%	84.24%	81.93%	77.02%	78.63%	80.55%
ResNet18 + FFT + Filter	88.90%	96.07%	97.58%	98.13%	97.87%	99.05%
USDConvNet-DG	76.31%	89.47%	87.72%	88.53%	87.41%	88.06%
USDConvNet-DG + FFT	89.17%	96.26%	97.53%	98.27%	97.97%	98.61%
USDConvNet-DG + Filter	77.89%	86.57%	91.23%	89.02%	88.91%	87.20%
USDConvNet-DG + FFT + Filter	89.15%	97.82%	99.15%	98.24%	97.01%	99.84%

system classified the event as slippage; otherwise, it was classified as no slippage. The results showed that the model's accuracy decreased to 93.48% on the training dataset and 80.58% on the test dataset. Additionally, the computation time increased to 5.34 ms because the detection process was repeated five times to evaluate the tactile events for all five fingertips individually. These findings indicate that considering all five fingertips as a whole is more effective than recognizing tactile events locally. Treating the five fingertips as a unified system not only improves the model's accuracy but also reduces computational overhead.

We applied 5-fold cross-validation to measure the accuracy for all six types, where the datasets for all six grasp types and the no-touch state were randomly and evenly divided into six groups. One group was used as the test set, while the other five groups were used for training and validation. The test was repeated five times. The accuracy on the validation set is 97.60%, with a standard deviation of 1.06%. The accuracy on the test set is 97.15%, with a standard deviation of 1.05%. These results provide evidence that the network demonstrates robust generalization across diverse grasp types.

Overall, the USDConvNet-DG model combined with FFT and filtering demonstrates the best generalization on the test set while maintaining high validation accuracy and short computing time, suggesting that this configuration is the most effective for slip detection in this experiment.

Moreover, we designed two groups of physical experiments to test the accuracy and real-time performance of USDConvNet-DG in real-world scenarios. In one group, the grasp state was detected in real-time while external force was applied to the object. In the other group, the grasping force was increased (from 100mN to 700mN) upon slip detection, demonstrating that the force adjustment could be completed with the object slipping by less than 1 cm. However, there were still limitations in accurately detecting minimal contact and slight slippage. For instance, slight slippage around the 6-s mark in Video 1 was not detected, and the contact state was misclassified in Video 3 due to minimal contact. Additionally, a clear delay existed between the end of slip and switching back to the no-slip state, as robotic hand re-established a stable state after detecting slippage. Video demonstrations are available at https://github.com/sunshine486/show.

5 Conclusion

Overall, this work presented a novel framework, USDFrame-DG, that performs slip detection across different grasp types for a five-fingered robotic hand equipped with integrated 3-axis force sensors. The proposed framework achieved this by utilizing a large dataset of various grasp types to train models, enabling it to detect slip across a wide range of untrained grasp types. It is found that the accuracy on the test gradually improve as the number of grasp types in the training set increased. To identify the most suitable network for universal slip detection, we designed three deep networks based on three classic deep learning models. Then, a lightweight network called USDConvNet-DG was designed based on the structure of the best-performing ResNet18. It has fewer parameters, shorter computation time, and no significant drop in accuracy. Using FFT and a digital high-pass filter for data preprocessing facilitated the extraction of spectral features and reduced low-frequency noise, significantly improving recognition accuracy. Physical experiments were conducted to demonstrate that the proposed framework can quickly detect the state of a grasp and adjust grasp force in real-time. These experiments also demonstrated that the ability to detect slip serves as a useful and reliable metric for determining grasp stability. Future research will focus on three aspects: First, we will explore the implementation of our framework on robotic hands with varying numbers of fingers and a diverse range of sensors. Second, the framework can be applied to adjust grasp strategies to achieve grasp stabilization. Third, a robotic hand equipped with slip detection should be capable of grasping unknown objects using minimal force while preventing them from slipping or being dropped.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

CZ: Data curation, Formal analysis, Methodology, Writing – original draft, Writing – review & editing. YY: Formal analysis, Funding acquisition, Methodology, Software, Resources, Writing – original draft, Writing – review & editing. ZY: Data curation, Methodology, Writing – original draft, Writing – review & editing. ZT: Methodology, Writing – original draft, Writing – review & editing. YZ: Formal analysis, Methodology, Software, Writing – original draft, Writing – review & editing. L-LZ: Conceptualization, Funding acquisition, Methodology, Project administration, Resources, Supervision, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research received funding from the STI 2030-Major Projects (Grant No. 2022ZD0208903), the National Natural Science Foundation of China (Grant Nos. 62006239, 61722313, and 62036013) and the Science and Technology Innovation Program of Hunan Province (Grant No. 2023RC1004).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary material for this article can be found online at:https://www.frontiersin.org/articles/10.3389/fnbot.2025.1478758/full#supplementary-material

References

Agur, A. M., and Dalley, A. F. II (2023). Grant's atlas of anatomy. Lippincott Williams & Wilkins.

Beccai, L., Roccella, S., Ascari, L., Valdastri, P., Sieber, A., Carrozza, M. C., et al. (2008). Development and experimental analysis of a soft compliant tactile microsensor for anthropomorphic artificial hand. *IEEE/ASME Trans. Mechatron.* 13, 158–168. doi: 10.1109/TMECH.2008.918483

Begalinova, A., King, R. D., Lennox, B., and Batista-Navarro, R., "Self-supervised learning of object slippage: an LSTM model trained on low-cost tactile sensors," in 2020 Fourth IEEE International Conference on Robotic Computing (IRC), (2022), pp. 191–196

Chen, W., Khamis, H., Birznieks, I., Lepora, N. F., and Redmond, S. J. (2018). Tactile sensors for friction estimation and incipient slip detection—toward dexterous robotic manipulation: a review. *IEEE Sensors J.* 18, 9049–9064. doi: 10.1109/JSEN.2018.2868340

Cortes, C., and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.* 20, 273–297. doi: 10.1007/BF00994018

Cui, S., Wang, S., Wang, R., Zhang, S., and Zhang, C. (2024). Learning-based slip detection for dexterous manipulation using GelStereo sensing. *IEEE Trans. Neural Net. Learn. Syst.* 35, 13691–13700. doi: 10.1109/TNNLS.2023.3270579

Cui, S., Wang, R., Wei, J., Hu, J., and Wang, S. (2020). Self-attention based visual-tactile fusion learning for predicting grasp outcomes. *IEEE Robo. Autom. Lett.* 5, 5827–5834. doi: 10.1109/LRA.2020.3010720

Deng, Z., Jonetzko, Y., Zhang, L., and Zhang, J. (2020). Grasping force control of multi-fingered robotic hands through tactile sensing for object stabilization. *Sensors* 20:1050. doi: 10.3390/s20041050

Deng, H., Zhang, Y., and Duan, X. G. (2017). Wavelet transformation-based fuzzy reflex control for prosthetic hands to prevent slip. *IEEE Trans. Ind. Electron.* 64, 3718–3726. doi: 10.1109/TIE.2016.2643603

Duhamel, P., and Vetterli, M. (1990). Fast Fourier transforms: a tutorial review and a state of the art. *Signal Process.* 19, 259–299. doi: 10.1016/0165-1684(90)90158-U

Feix, T., Romero, J., Schmiedmayer, H. B., Dollar, A. M., and Kragic, D. (2016). The GRASP taxonomy of human grasp types. *IEEE Trans. Human-Machine Syst.* 46, 66–77. doi: 10.1109/THMS.2015.2470657

Fiedler, N., Jonetzko, Y., and Zhang, J., "A multimodal pipeline for grasping fabrics from flat surfaces with tactile slip and fall detection," in 2023 IEEE International Conference on Robotics and Biomimetics (ROBIO), (2023), pp. 1–6

Garcia-Garcia, A., Zapata-Impata, B. S., Orts-Escolano, S., Gil, P., and Garcia-Rodriguez, J., "TactileGCN: a graph convolutional network for predicting grasp stability with tactile sensors," in 2019 International Joint Conference on Neural Networks (IJCNN), (2019), pp. 1–8

Grover, A., Nadeau, P., Grebe, C., and Kelly, J., "Learning to detect slip with barometric tactile sensors and a temporal convolutional neural network," in 2022 International Conference on Robotics and Automation (ICRA), (2022), pp. 570–576.

He, K., Zhang, X., Ren, S., and Sun, J., "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2016), pp. 770–778

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.* 9, 1735–1780. doi: 10.1162/neco.1997.9.8.1735

Holweg, E. G. M., Hoeve, H., Jongkind, W., Marconi, L., Melchiorri, C., and Bonivento, C., "Slip detection by tactile sensors: algorithms and experimental results," in Proceedings of IEEE International Conference on Robotics and Automation, (1996)

James, J. W., and Lepora, N. F. (2021). Slip detection for grasp stabilization with a multifingered tactile robot hand. *IEEE Trans. Robot.* 37, 506–519. doi: 10.1109/TRO.2020.3031245

James, J. W., Pestell, N., and Lepora, N. F. (2018). Slip detection with a biomimetic tactile sensor. *IEEE Robo. Autom. Lett.* 3, 3340–3346. doi: 10.1109/LRA.2018.2852797

Johansson, R. S., and Flanagan, J. R. (2009). Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nat. Rev. Neurosci.* 10, 345–359. doi: 10.1038/nrn2621

Johansson, R. S., and Vallbo, A. B. (1979). Tactile sensibility in the human hand: relative and absolute densities of four types of mechanoreceptive units in glabrous skin. *J. Physiol.* 286, 283–300. doi: 10.1113/jphysiol.1979.sp012619

Johansson, R. S., and Westling, G. (1984). Roles of glabrous skin receptors and sensorimotor memory in automatic control of precision grip when lifting rougher or more slippery objects. *Exp. Brain Res.* 56, 550–564. doi: 10.1007/BF00237997

Melchiorri, C. (2000). Slip detection and control using tactile and force sensors. *IEEE/ASME Trans. Mechatron.* 5, 235–243. doi: 10.1109/3516.868914

Mi, T., Que, D., Fang, S., Zhou, Z., Ye, C., Liu, C., et al., "Tactile grasp stability classification based on graph convolutional networks," in 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR), Xining, China, (2021) 875–880. doi: 10.1109/RCAR52367.2021.9517085

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2021). Deep double descent: where bigger models and more data hurt. *J. Statis. Mech. Theory Experi.* 2021:124003. doi: 10.1088/1742-5468/ac3a74

Romeo, R. A., Lauretti, C., Gentile, C., Guglielmelli, E., and Zollo, L. (2021). Method for automatic slippage detection with tactile sensors embedded in prosthetic hands. *IEEE Trans. Med. Robo. Bionics* 3, 485–497. doi: 10.1109/TMRB.2021.3060032

Romeo, R. A., and Zollo, L. (2020). Methods and sensors for slip detection in robotics: a survey. *IEEE Access* 8, 73027–73050. doi: 10.1109/ACCESS.2020.2987849

Shensa, M. J. (1992). The discrete wavelet transform: wedding the a trous and Mallat algorithms. *IEEE Trans. Signal Process.* 40, 2464–2482. doi: 10.1109/78.157290

Song, X., Liu, H., Althoefer, K., Nanayakkara, T., and Seneviratne, L. D. (2013). Efficient break-away friction ratio and slip prediction based on haptic surface exploration. *IEEE Trans. Robot.* 30, 203–219. doi: 10.1109/TRO.2013.2279630

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al., Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). pp. 6000–6010. doi: 10.5555/3295222.3295349

Westling, G., and Johansson, R. S. (1984). Factors influencing the force control during precision grip. *Exp. Brain Res.* 53, 277–284. doi: 10.1007/BF00238156

Xie, Z., Piriyatharawet, T., and Roberto, C., "Deep learning LSTM-based slip detection for robotic grasping," in IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society, (2023), pp. 1–5

Yan, G., Schmitz, A., Tomo, T. P., Somlor, S., Funabashi, S., and Sugano, S., "Detection of slip from vision and touch," in 2022 International Conference on Robotics and Automation (ICRA), (2022), pp. 3537–3543

Yang, D., and Wu, G. (2021). A multi-threshold-based force regulation policy for prosthetic hand preventing slippage. *IEEE Access* 9, 9600–9609. doi: 10.1109/ACCESS.2021.3049854

Zapata-Impata, B. S., Gil, P., and Torres, F. (2019). Learning Spatio temporal tactile features with a ConvLSTM for the direction of slip detection. *Sensors* 19, 523–539. doi: 10.3390/s19030523

Zeng, B., Liu, H., Song, H., Zhao, Z., Fan, S., Jiang, L., et al. (2022). Design and slip prevention control of a multi-sensory anthropomorphic prosthetic hand. *Indus. Robo. Int. J. Robo. Res. App.* 49, 289–300. doi: 10.1108/IR-07-2021-0133

Zhang, Y., Duan, X. G., Zhong, G., and Deng, H. (2016). Initial slip detection and its application in biomimetic robotic hands. *IEEE Sensors J.* 16, 7073–7080. doi: 10.1109/JSEN.2016.2596840

Zhang, Y., Kan, Z., Tse, Y. A., Yang, Y., and Wang, M. Y. (2018). FingerVision tactile sensor design and slip detection using convolutional LSTM network. *ArXiv* abs/1810.02653. doi: 10.48550/arXiv.1810.02653

frontiers | Frontiers in Neurorobotics

TYPE Original Research
PUBLISHED 13 May 2025
DOI 10.3389/fnbot.2025.1568811

OPEN ACCESS

EDITED BY
Long Jin,
Lanzhou University, China

REVIEWED BY
Bolin Liao,
Jishou University, China
Chang Liao,
Shandong University, China
Shan Rong,
Sehan University, Republic of Korea

*CORRESPONDENCE
Yilin Zhang
✉ 18241116887@163.com

RECEIVED 30 January 2025
ACCEPTED 23 April 2025
PUBLISHED 13 May 2025

CITATION
Zhang Y (2025) TS-Resformer: a model based on multimodal fusion for the classification of music signals.
Front. Neurorobot. 19:1568811.
doi: 10.3389/fnbot.2025.1568811

TS-Resformer: a model based on multimodal fusion for the classification of music signals

Yilin Zhang*

Dalian University of Foreign Languages, International Art College, Dalian, China

The number of music of different genres is increasing year by year, and manual classification is costly and requires professionals in the field of music to manually design features, some of which lack the generality of music genre classification. Deep learning has had a large number of scientific research results in the field of music classification, but the existing deep learning methods still have the problems of insufficient extraction of music feature information, low accuracy rate of music genres, loss of time series information, and slow training. To address the problem that different music durations affect the accuracy of music genre classification, we form a Log Mel spectrum with music audio data of different cut durations. After discarding incomplete audio, we design data enhancement with different slicing durations and verify its effect on accuracy and training time through comparison experiments. Based on this, the audio signal is divided into frames, windowed and short-time Fourier transformed, and then the Log Mel spectrum is obtained by using the Mel filter and logarithmic compression. Aiming at the problems of loss of time information, insufficient feature extraction, and low classification accuracy in music genre classification, firstly, we propose a Res-Transformer model that fuses the residual network with the Transformer coding layer. The model consists of two branches, the left branch is an improved residual network, which enhances the spectral feature extraction ability and network expression ability and realizes the dimensionality reduction; the right branch uses four Transformer coding layers to extract the time-series information of the Log Mel spectrum. The output vectors of the two branches are spliced and input into the classifier to realize music genre classification. Then, to further improve the classification accuracy of the model, we propose the TS-Resformer model based on the Res-Transformer model, combined with different attention mechanisms, and design the time-frequency attention mechanism, which employs different scales of filters to fully extract the low-level music features from the two dimensions of time and frequency as the input to the time-frequency attention mechanism, respectively. Finally, experiments show that the accuracy of this method is 90.23% on the FMA-small dataset, which is an improvement in classification accuracy compared with the classical model.

KEYWORDS

music genre classification, Fourier transform, residual network, transformer, attention mechanism

1 Introduction

With the popularization of the Internet and the development of artificial intelligence, the medium of music is no longer confined to records or tapes, but digital media such as cell phones, computers, mp3, and so on. Digital media and streaming platforms have provided strong support for the music industry, and a huge amount of music tracks have been made

available to more and more people through diversified communication media, allowing them to access their favorite music anytime and anywhere. After 2015, the global music industry has entered the digital era. As of December 2023, China's Internet penetration rate has reached 76.4% (Zhai and Luo, 2025). Thanks to the popularization of the Internet, the music industry has had better development. The global music streaming market size grew from about $15 billion in 2015 to about $43 billion in 2020, with a CAGR of about 23%. On this trend, the global music streaming market size will reach USD 82 billion by 2025. This growth trend is mainly driven by factors such as digitalized music consumption methods and the popularity of mobile internet. At the same time, competition among music streaming platforms is becoming more and more intense, with platforms constantly introducing new features and services to attract more users and increase user retention. In future development, music streaming media should not only provide high-quality audio services to attract more users but also apply artificial intelligence technology to introduce more intelligent recommendation systems and personalized services to improve user experience. There are various styles of music, and different genres have different audiences. Accurate identification of the genre for a new music track is a highly concerning issue, directly affecting the effectiveness of recommendation and user satisfaction. The traditional music genre classification method is mainly realized by manual classification, and the classifiers need to have high professional quality in music. Therefore, traditional music classification has the following drawbacks:

(1) Not suitable for large-scale datasets.
(2) The feature extraction method is complex and requires the designer to have a high level of expertise in music.
(3) The feature extraction method lacks generalization, and different classification tasks need to compute music features separately and individually.

Therefore, there is a need for a music genre classification model with high accuracy, which utilizes the powerful arithmetic power of computers to achieve automated classification. In recent years, neural networks have made impressive achievements in image recognition, optimizing the tedious step of manually designing features in traditional machine learning and providing a new direction for music genre classification. Applying neural networks to the music genre classification task has much room for development, and deep learning technology provides theoretical support for it. The research on music genre classification methods based on deep learning mainly has the following problems:

(1) The current research mainly analyzes the overall accuracy rate of all categories in the dataset, ignoring the lower accuracy rate of individual genre music classification.
(2) Music has a strong time-series relationship, and the serial model architecture design has the problem of losing time-series information.

To address the above problems, we analyze and improve the model, and the main contribution consists of the following aspects:

(1) For data processing, we design a cut-score method to cut music audio into different time durations. Experimentally, we analyze the effect of different time lengths on classification accuracy and training time, select the applicable cut time length for music, and realize data enhancement. The audio signal is converted to Log Mel spectrum by short-time Fourier change, Mel scale filter, and logarithmic compression. Compared with other features, the Log Mel spectrum fully preserves the characteristic information of the musical work by describing the energy intensity of the audio signal and the distribution of information in the time and frequency domains.
(2) In terms of network model design, Res-Transformer, a parallel music genre classification model based on residual network and Transformer coding layer, is proposed, with the following design idea: unlike ordinary audio, music has a strong time-series relationship. Existing serial architectures use RNNs as temporal summarizers to extract time-series information, and their performance depends largely on the results of previous convolutional layers, and the time-series information of the original music is partially lost during the convolution process, resulting in unsatisfactory classification accuracy. To preserve the spatial characteristics and temporal order of the original music samples, the Transformer Encoder is used to extract the time series information of the music directly from the Log Mel spectrum. The RNN network predicts the frequency changes based on the neighboring time steps, while the Transformer's multi-head self-attention layer enables the network to look at multiple previous time steps when predicting the next time step, which is better than the RNN. The effect is superior to RNN. Improve the residual network to enhance music genre feature extraction while avoiding gradient vanishing, add 1×1 convolution kernel in the jump connection to achieve dimensionality reduction, and introduce nonlinear transformations to increase the expressive power of the network.
(3) In terms of model optimization, a parallel music classification model TS-Resformer is proposed based on the fusion of time-frequency and channel attention mechanisms in the residual network and Transformer coding layer. By adding the designed time-frequency attention mechanism in front of the residual module, the convolution adopts the different scales of filters to capture the features from the two dimensions of time and frequency, respectively. Experimenting with different scale convolution kernels for categories with low classification accuracy, the shapes of the two filters were analyzed and determined. After the time-frequency convolutional features are extracted and input into the time-frequency attention mechanism respectively, the feature map with time-frequency weights is formed by feature fusion.

2 Related work

Music is created by the inspiration of human beings, different combinations of instruments and different vocal choruses form a unique musical repertoire, and each song has a unique melody, rhythm, timbre, and other artistic elements, which are far different from ordinary audio. Music Genre Classification (MGC) is one of the branches of Music Information Retrieval (MIR), which is mainly used to classify different music genres.

As a classification task, Music Genre Classification mainly includes three steps:

(1) Data preprocessing, which prepares for the next step of feature extraction by processing the original audio.
(2) Feature extraction, which extracts the features that represent the information of music genres.
(3) Classification, the extracted audio feature vectors are input into the classifier to realize music genre classification.

The results of traditional music classification scientific research are mainly reviewed from two perspectives: feature extraction and classification model.

In terms of feature extraction, traditional music classification extracts handmade features from the original audio signal. Arabi and Lu (2009) introduced chordal features in the process of feature extraction to better represent the characteristics of the music signal and combined them with low-level music features, using a support vector machine as a classifier, and experiments proved that combining low-level music features with high-level music features can improve the classification accuracy rate The paper is a comprehensive study of the classification of chord features in songs. The paper extracts chord features by counting the root notes of chords in each song, which is computationally complex and cannot realize end-to-end music classification. Logan (2000) proposed the Mel-Frequency Cepstral Coefficient (MFCC) and experimentally demonstrated the importance of the MFCC features in the field of audio recognition and the importance of the MFCC features in the field of music retrieval. And experimentally demonstrates the importance of MFCC features in the field of audio identification and its advancement and applicability in the field of music retrieval. Based on MFCC features, Baniya et al. (2014) used wavelet decomposition-based timbre texture (MFCC and other spectral features) and rhythmic content features to improve the performance. The paper retains the features that are useful for classification and discards those that are not effective, but there is a possibility of misclassification in the manual selection of features. Sarkar and Saha (2015) use empirical pattern decomposition to capture local features of different genres, and then compute pitch-based features from the decomposed songs, but the pitch features alone do not adequately characterize the musical features, and the method is only suitable for simple classification of music genres.

In terms of music classification models, Tzanetakis and Cook (2002) realized music classification by several models such as the Gaussian classifier, Gaussian mixture model (Duda et al., 2012) and K-nearest neighbor and the input features are hand-designed rhythmic content, pitch, and timbre features. This paper achieved the automatic classification of music genres earlier but did not design classification models for music features. Xi et al. (2004) extracted inverted spectral domain features of music, performed feature engineering to extract discriminative music features and achieved classification through Hidden Markov Models, but such models need to appropriately adjust the complexity of the model to avoid overfitting. Ali and Siddiqui (2017) used Principal Component Analysis to compare the performance of KNN vs. Support Vector Machines (SVMs) using Principal Component Analysis (PCA), and without dimensionality reduction, KNN and SVMs perform well, and SVMs are more efficient.

In addition, Sturm (2012) analyzed the GTZAN dataset, which is widely used in the field of music genre classification, and experimentally proved that this dataset has the problem of missing labels and errors, and the experimental results obtained on this dataset in the past are not accurate.

Deep learning models can achieve end-to-end learning, i.e., the complete learning process from the original data to the final prediction result, simplifying the design and optimization process of the system and improving the overall efficiency of the model. In recent years neural networks have gained great success in the fields of computer vision as well as speech recognition, and the field of music information retrieval has also begun to widely use deep learning neural networks (Orjesek et al., 2022; Ostermann et al., 2023) to realize end-to-end music genre classification (Solanki and Pandey, 2022; Lin, 2022). For deep learning music classification methods are mainly introduced in terms of feature extraction, network model, and attention mechanism.

In terms of feature extraction, researchers have investigated a variety of audio feature extraction methods. Pelchat and Gelowitz (2020) reviewed some of the machine learning techniques used in the field, utilizing spectrograms generated from song time slices as inputs to a neural network. Spectrograms contain information about multiple musical features, which facilitates model training. Chen et al. (2024) achieved 68.78% accuracy on the FMA-small dataset by using a CNN with residuals in series with bi-GRU, and Mel spectrograms as inputs were better compared to acoustic spectrograms obtained from short-time Fourier transform. Traditional MGC methods only consider audio information or lyrics information, which leads to unsatisfactory recognition results. Li et al. (2023) proposed a multi-modal music genre classification framework that integrates audio information and lyrics information. The framework uses a convolutional neural network to extract audio features from the Mel spectrogram while obtaining a distributed representation of the lyrics. The two modal information is fused through two different strategies, feature level, and decision level, but this approach increases the computation time.

Deep learning network models mainly utilize the strong recognition ability of convolutional networks on image information to achieve music genre classification. Zhang et al. (2016) proposed a CNN model that combines residuals with maximum-minimum pooling to provide higher statistical information for neural networks, but this model does not take into account the time-series information of the music. Choi et al. (2017) proposed a two-dimensional convolutional recurrent CRNN model the convolutional layer extracts features followed by a recurrent and fully connected layer to perform the classification task. This model is popular and many variants have evolved from it. Yang et al. (2020) parallelized CNN and RNN networks to process the inputs, allowing the RNN network to process the original spectrogram instead of processing the output of the CNN but suffered from insufficient feature information extraction. Vaibhavi and Krishna (2021) demonstrated the effectiveness of the CRNN in exploiting the spatial feature extraction capability of CNN and the RNN summarization time of the RNN in the GTZAN dataset. The capability of CNNs and the ability of RNNs to summarize temporal patterns (Verma et al., 2023). Kostrzewa et al. (2021) proposed multiple integrations of CNNs, RNNs, and CRNNs to combine the advantages of different deep neural network architectures, and evaluated this approach on the FMA-small dataset, obtaining an F1 score of 54.9%.

Incorporating the attention mechanism in the model and utilizing the attention mechanism to further improve the accuracy of music classification is one of the more popular research directions in recent

years. Zhuang et al. (2020) proposed a method to achieve music classification using a Transformer classifier without using loop and convolutional structure and achieved 76% accuracy on the GTZAN dataset, which is not ideal. Prabhakar and Lee (2023) utilized a classifier with a graphical attention model to improve image processing, image attention network can automatically learn important regions and features in an image to improve the effectiveness of an image processing task. By weighting different regions in the image, the network can pay more attention to the important information. Khasgiwala and Tailor (2021) learned different levels of self-attention, i.e., learning to find relationships between different parts of the image by Vision Transformer X by retaining the positional information of the features in the image, comparing CNN, RNN-LSTM, Vision Transformer X on FMA. Wen et al. (2024) to solve the problem that the limited sensory field of a convolutional neural network cannot capture the correlation between the time frames of vocalization at any moment and the sound frequencies of all vibrations in the song, applied dual parallel attention to focus on global dependencies in CNN-5, proposed parallel channel attention to constructing the global time-frequency dependencies in the song, and designed the double parallel attention to focus on the global time-frequency dependence in songs. Many models do not effectively design the feature extraction layer of the convolutional structure for the music signal features, and the feature extraction part is relatively simple, which causes the models to neglect the extraction of local features. To solve the above problems, Xie et al. (2024) proposed a model using a one-dimensional res-gate CNN to extract local information of audio sequences. To aggregate the global information of audio feature sequences, the Transformer is applied to the music genre classification model, and the decoder structure of the Transformer is modified according to the task. Zhang et al. (2025) proposed and applied a novel talking face generation framework, termed video portraits transformer (VPT) with controllable blink movements. In the audio-to-landmark stage, the transformer encoder serves as the generator used for predicting whole facial landmarks from given audio and continuous eye aspect ratio (EAR). Zhang et al. (2024) proposed a convolutional dynamically convergent differential neural network (ConvDCDNN) to solve the classification problem of the electroencephalography (EEG) signals. First, a single-layer convolutional neural network is used to replace the preprocessing steps in previous work. Then, focal loss is used to overcome the imbalance in the dataset. After that, a novel automatic dynamic convergence learning (ADCL) algorithm is proposed and proved for training neural networks.

3 Methods

3.1 Music data preprocessing

In Section 3.1, first of all, we introduced and presented the dataset we selected. Then, we perform file integrity checking, data normalization, data augmentation and dataset splitting on the original audio, and then transform the audio file to generate the Log Mel spectrum as the input for the subsequent TS-Resformer model we proposed.

The most commonly used datasets for music genre classification tasks are GTZAN and the Free Music Archive (FMA) dataset. The dataset we use is FMA, a large music audio dataset designed specifically for MIR studies, which is larger and more specialized than the other datasets, providing 106,574 songs from 16,341 artists arranged in a 161-genre hierarchical structure and associated with dedicated metadata. It is categorized into three versions: large, medium, and small. Considering the computational resources, we adopt the small version, which has far more songs than the commonly used GTZAN dataset.

The FMA-small dataset contains a balanced subset of 8,000 songs distributed in 8 genres: Electronic, Experimental, Folk, Hip-Hop, Pop, Rock, Instrumental, and International. Each music clip is saved in map3 format, with a time length of about 30s and a sampling rate of 44,100 Hz. The number of music in each genre in the FMA-small dataset is shown in Figure 1.

To exclude interference from corrupted files as well as to improve classification accuracy, the following data processing steps are performed on the raw audio:

(1) Verify track integrity

Corrupted audio tracks with unsatisfactory sampling rates and unsatisfactory time lengths are discarded to ensure that the audio files left behind can be recognized properly. The processed dataset has 7,992 music tracks.

(2) Data augmentation

Addressing the issues of how varying audio durations affect the accuracy of music genre classification and training time, an approach of segmenting the original audio is adopted. Separate experiments are conducted to select an optimal audio duration suitable for the task of music genre classification. On the other hand, segmenting the original audio can achieve data augmentation, where more training data enhances the model's generalization performance. The model designed for the music genre classification task should be suitable for large-scale datasets. In the FMA-small dataset, each excerpt C of a song's genre has a duration of 30 s. Data augmentation is achieved by segmenting these music excerpts, with each resulting sub-segment c_i having a duration of l seconds and a 50% overlap between two adjacent sub-segments. After segmentation, N sub-segments of l seconds each can be obtained, and each segmented sub-segment carries the same genre information as the original excerpt, defined as per the Equation 1:

$$C = \{c_1, c_2, \cdots, c_n\} \tag{1}$$

For example, slicing a Rock genre music clip in the dataset will result in 4 sub-clips of 12 s duration, each of which is labeled with the Rock genre.

(3) Z-score normalization of raw audio

The Z-Score method calculates the amplitude mean of the audio by taking the absolute values of all the music amplitudes and then summing and dividing them by the number of samples. The Z-Score method normalizes the audio based on the mean and standard deviation of the original audio. Normalization is applied to the original audio to prevent numerical overflow, improve the stability of

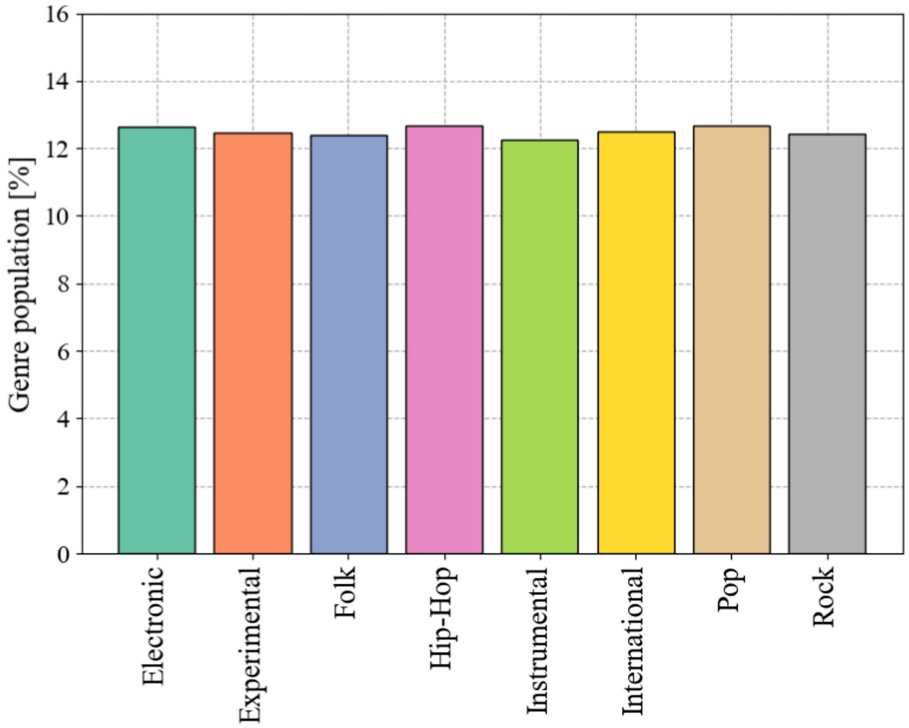


FIGURE 1
The proportion of eight musical analogies in FMA-small data set.

model training and convergence speed, and better adapt to the input requirements of the model, Z-Score the Equation 2.

$$x' = \frac{x - \mu}{\sigma} \tag{2}$$

where x denotes the original data, μ denotes the mean of the original data, σ denotes the standard deviation of the original data, and x' denotes the normalized data.

(4) Slicing the data set

The dataset is sliced into training set, validation set, and test set according to 8:1:1, and the number of songs of different genres in the three sets is guaranteed to be balanced. The librosa library is utilized to load audio clips in each training selection generation.

The audio waveform graphs show the amplitude changes of the audio signal at different time points, and the original audio waveforms are shown in Figure 2.

The horizontal axis of the graph is time, and the vertical axis is the sound amplitude; when the amplitude is larger, the greater the vibration amplitude of the waveform; a smaller amplitude means that the vibration amplitude of the waveform is relatively small.

Although the time domain can be converted to the frequency domain by Fourier transform, so that the frequency distribution expression is intuitive, the time domain information is lost. Therefore, the use of a short-time Fourier, wavelet time-frequency domain analysis method is more effective in avoiding such problems. We mainly use short-time Fourier transform to process the music audio. Through a series of processing methods, the audio is transformed into a Log Mel spectrum, which describes the energy distribution of the music at different times and corresponding frequency ranges, provides the conditions for the model to fully extract the audio features, and has the following advantages:

(1) Distribution of time-frequency information

The Log Mel spectrum contains more information about music features, and the time-frequency distribution is clearer, which describes the change of the music signal frequency over time respectively, and uses the depth of the color in the third dimension to indicate the magnitude of the sound intensity in decibels (dB). The change of frequency peaks can be observed and contains more information about the music features compared to the traditional spectrogram.

(2) Robustness

The Mel filter is used in the transformation process of the Log Mel spectrum, which filters the interference noise in the music audio to a certain extent and is less affected by the noise and environmental changes, which makes the accuracy of the experimental results increase.

(3) Strong modeling capability

As the Log Mel spectrum contains several features such as Mel inverse spectral coefficient, logarithmic amplitude spectrum, etc., it can better characterize the spectral characteristics of the audio signal, and the calculation of the Log Mel spectrum is relatively simple, which

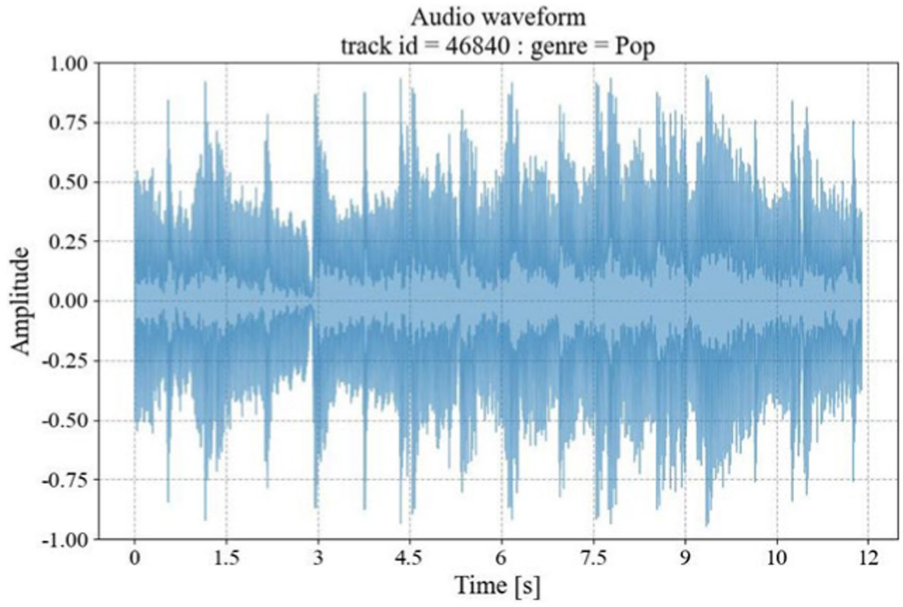


FIGURE 2
Original audio waveform diagram.

only involves the calculation of the Mel filter bank and the operation of taking logarithmic numbers, whereas the MFCC needs to carry out the operation of the inverse spectral transform and the discrete cosine transform, etc. As the representation of the linear features, it is difficult to capture the complex structure and nonlinear features of the audio signal, and it is not easy to capture the complex structure and nonlinear features of the audio signal. As a linear feature representation, MFCC makes it difficult to capture the complex structure and nonlinear features of audio signals, while the Log Mel spectrum is closer to human auditory perception.

(4) Easy to train the model

Log Mel spectrum converts the audio signal into a two-dimensional image matrix, which reduces the input data of the model compared to the original audio form and makes the training efficient. Its form is similar to an image, and the neural network model is suitable for processing image data, and it is easy to be integrated with deep learning frameworks for end-to-end training. The Log Mel spectrum shows the frequency of the audio signal over time, and the flow of the conversion of the original audio to the Log Mel spectrum is shown in Figure 3.

In the music genre classification task firstly the pre-processed music audio is sub-framed and windowed. Music audio is a continuous signal that changes over time, and frame splitting divides the continuous audio signal into short time segments, each of which can be considered as a steady state. Split framing helps to capture the notes and pitch changes of music without affecting the temporal structure of the whole music. At the same time, the music can be considered frequency stable in short time segments, which helps to deal with large frequency variations in music audio, such as pitch changes of musical instruments. Windowing the audio after frame splitting reduces the risk of spectral leakage, and the windowing operation makes the music segments transition smoothly in time and reduces noise due to signal discontinuities.

The short-time Fourier transform is applied to each frame of the audio signal, based on which the energy of each Mel band is obtained by applying the Mel scale filter, and finally, the Log Mel spectrum is obtained after logarithmic compression, which expresses the music information in terms of time, frequency and energy intensity. The short-time Fourier transform the Equation 3:

$$X(t,f)=\int_{-\infty}^{\infty}x(t)w(t-k)e^{-j2\pi fk}\mathrm{d}k \tag{3}$$

where $w(t\text{-}k)$ is the time window function, multiplied by the signal $x(0)$ for Fourier transform, and the spectral operation the Equation 4.

$$SP_x(t,f)=|X(t,f)|^2=\left|\int_{-\infty}^{\infty}x(t)w(t-k)e^{-j2\pi fk}\mathrm{d}k\right|^2 \tag{4}$$

The short-time Fourier transform is a key step in the process of transforming the Log Mel spectrum from audio, and the choice of window length in the short-time Fourier transform directly affects the experimental results. We experimentally created these audio clips using a window containing 4,096 samples, with a jump window size of 1,024, and used the Hanning window function to reduce the amplitude of the discontinuities at the boundaries, which is close to 1 s for 4,096 samples in terms of duration. The window size is chosen to be a power of 2 to ensure the high efficiency of the FFT algorithm, and if this is not the case, a zero-completion operation can be used. In addition, using 4,096 samples as the window size ensures that the FFT is performed with uniform frequency resolution and reduces the occurrence of spectral leakage. Spectral leakage refers to the deviation

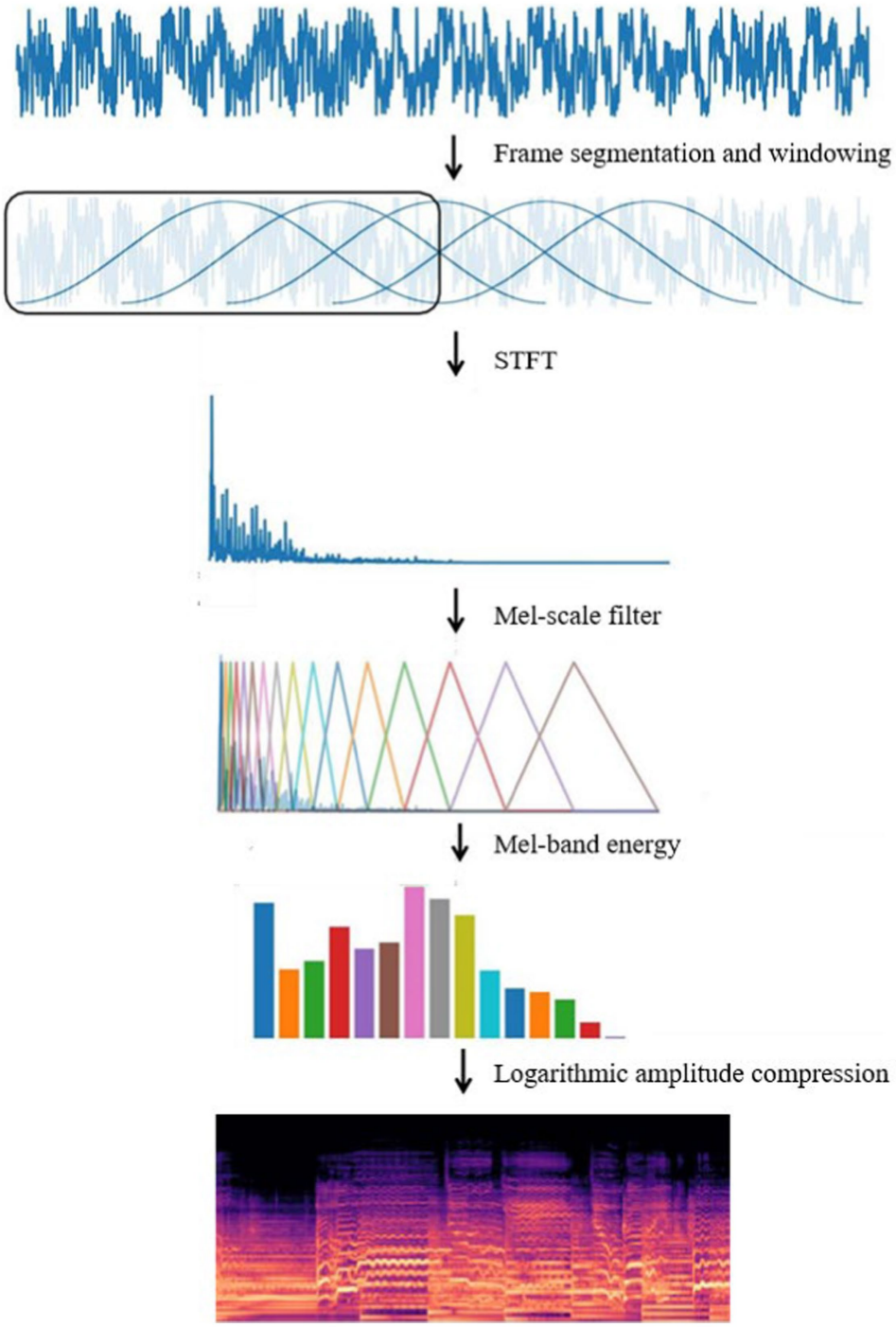


FIGURE 3
Log Mel spectrum conversion process.

of the signal frequency waveform due to the truncation of the window function, which leads to a reduction in the accuracy of spectral estimation.

The human ear perceives low-frequency tones and high-frequency tones nonlinearly, with low-frequency tones being low and thick and high-frequency tones being sharp. The human ear perceives mid-frequency tones, which are the dominant frequency range of most human voices and musical instruments, more strongly. The Mel filter bank simulates the nonlinear perception of sound frequencies by the human ear through the analog Mel scale. The Mel filter bank divides the frequency spectrum into frequency bands by the auditory characteristics of the human ear, and each band corresponds to the energy response of a Mel filter. Therefore, converting the audio signal from a linear frequency to the Mel frequency scale better reflects the way the human ear perceives sound. Mel spectrograms are created by applying a set of 128 overlapping Mel scale filters to calculate the spectral energy of each frequency band, and each Mel spectrogram has the

shape of [128, 512], which fully preserves the characteristic information of the music and reduces the dimensionality of the model input data.

The Mel scale conversion the Equation 5:

$$\mathrm{Mel}(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{5}$$

The amplitude of the Meier spectrogram is converted to a decibel (dB) scale, which compresses the dynamic range of the signal and makes it easier to identify the relative strengths of the different frequency components of the spectrogram. The logarithmic Mel Spectrogram obtained from the conversion of the audio clip waveform graph shown in Figure 2 is shown in Figure 4.

3.2 TS-Resformer model

3.2.1 General structure

We propose an improved residual network and transformer encoder parallel music type classification model-TS-Resformer. The model structure is shown in Figure 5. First, we parallelize the residual network with the Transformer Encoder module, the left branch of the feature extraction is the residual network to extract the spectral feature information, and the right branch of the feature extraction is the four-layer Transformer Encoder to extract the time series information. Then, different scales of filters are designed for the four types of music genres with poor left classification effect, focusing on extracting time domain and frequency domain features respectively, fully extracting the time-frequency features and designing a new time-frequency attention mechanism to be applied on the branch. Finally, to enhance the model's attention to important features between channels and capture the spatial relationship between features, the residuals are combined with the SE Block to form a new SE-Res Block.

3.2.2 TF-attention module

Most attention-based MGC systems add an attention mechanism at the end of deep learning network architecture feature extraction to learn high-level musical representations (Zhang et al., 2025; Zhang et al., 2024). Existing time-frequency attention mechanisms mainly extract high-level features at the bottom of the model, but low-level musical features such as energy, pitch, and tone are not sufficiently extracted, limiting the performance of the MGC task. Therefore, the time-frequency attention is placed after the multi-scale time-frequency feature extraction layer Conv1, and the extracted time-frequency features are learned, and the TF-Attention module, which integrates the time-frequency attention mechanism with time-frequency feature extraction, is shown in Figure 6.

We designed the time-frequency feature extraction layer Conv1 applicable to Log Mel spectrums to extract more features from Log Mel spectrums. Two parallel convolutional filters with different shapes are designed to learn the feature representation across time and frequency as shown in Figure 7.

The data in the Conv1 layer represents input channel 1 and output channel 32 respectively, the filter shape in the frequency domain feature extraction part is $M \times 2$, and the filter shape in the time domain feature extraction part is $2 \times N$. Music is quite different from normal audio, so the shape of the two parallel convolutional filters needs to be re-determined, i.e., the values of M and N need to be determined experimentally. Finally the extracted features from the two branches are fed into the lower network.

Conv1 inputs the extracted music time-frequency features into the Time-Frequency Attention Mechanism module to mine effective low-level music features and assigns corresponding weights to the input audio in the time and frequency domain

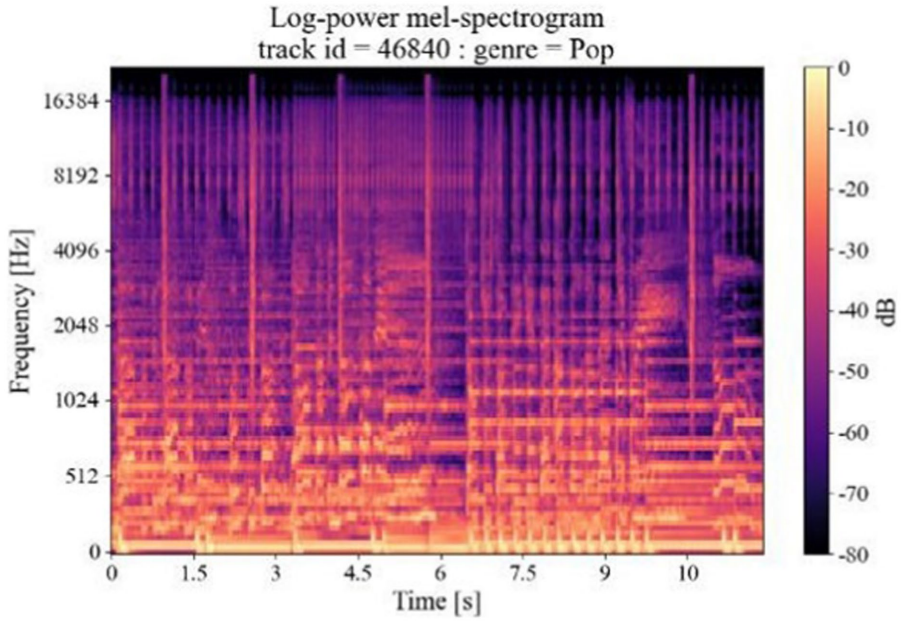


FIGURE 4
Log Mel-spectrogram.

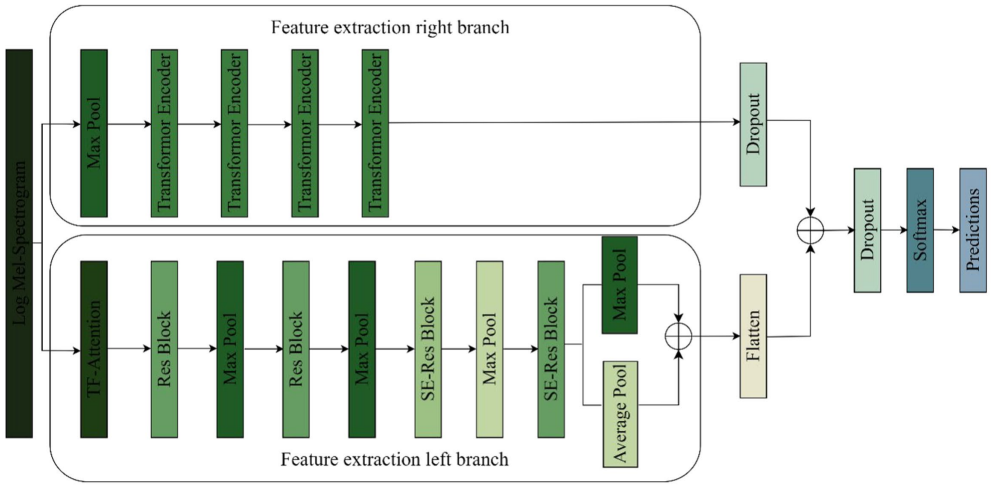


FIGURE 5
TS-Resformer model network architecture.

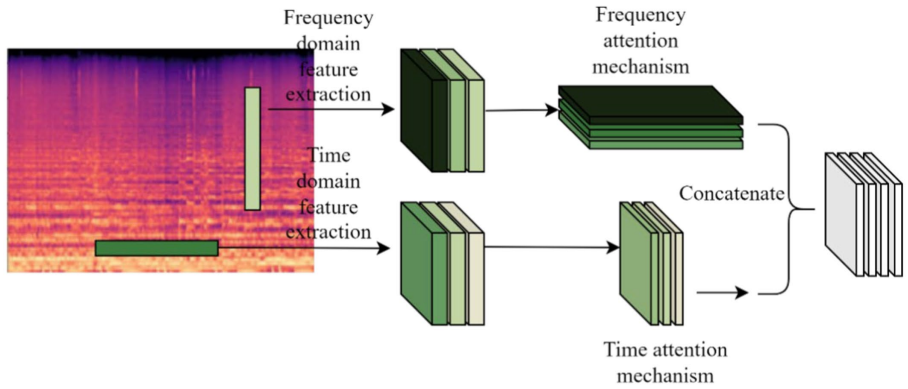


FIGURE 6
TF-attention module.

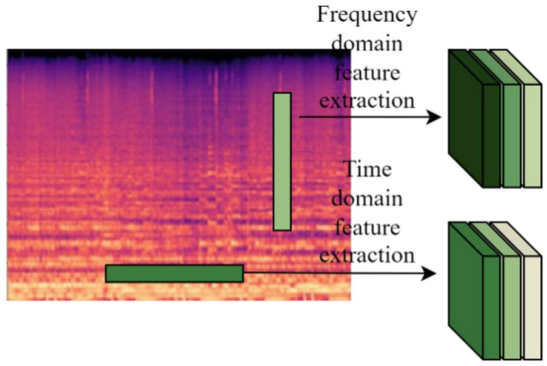


FIGURE 7
Time-frequency feature extraction.

directions to form a feature map with time-frequency weighting information, learns the important information, and discards the information that is irrelevant to music classification. The internal structure of the time domain attention mechanism is shown in Figure 8.

The frequency attention mechanism is similar. The input is 1x128x512, and the feature model is generated after average pooling of X along the time direction, $Y_t \in R^{1*1*T}$, operating as the Equation 6.

$$Y_t = \frac{1}{T}\sum_{n=1}^{T} X(n) \tag{6}$$

The scaling of the first convolutional layer is set to 16, the ReLU activation function is used to process it, and then a layer of convolution is used to construct the inter-channel correlation, and finally the Sigmoid is used to fix the value between 0 and 1 to get the weight in the time direction, which is multiplied by the original input to get the new feature map with the time weights. Similarly, the feature map with frequency weights can be obtained, and the two new feature maps are spliced together and fed into the lower layer network for processing.

Using Sigmoid as a scaling function for audio signals avoids focusing attention on a few time frames. By selectively aggregating the

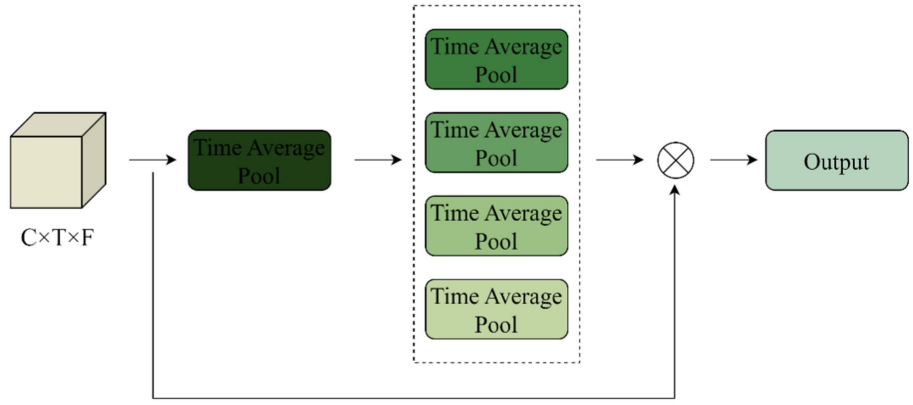


FIGURE 8
Time attention mechanism structure diagram.

front and back frames according to the time-frequency attention, similar musical features achieve a common facilitation, thus improving the compactness within the class. Channel concatenation has advantages: first, the lower layers are able to receive large regions in the time and frequency domains, providing more information for advanced representation learning.

3.2.3 SE-res module

Log Mel spectrums generated after the feature extraction layer of the time-frequency attention mechanism have a better representation of the music, but also do not classify well if the model does not focus well on important feature information. The SE module weights each channel of the feature map and effectively ignores features that are not relevant to the music classification in order to selectively emphasize the features that are more relevant to the music classification. The feature maps are extracted from the Log Mel spectrums, and the weights of each of these channels are dynamically adjusted by the channel attention mechanism to make the network model more representational. The channel attention mechanism consists of two steps: compression and excitation.

The structure of the SE module is shown in Figure 9.

The compression step compresses the feature map $U \in R^{H \times W \times C}$ to 1x1xC by means of a global average pooling operation to achieve the conversion of spatial features to global features, and the resulting global information z is used as the feature weights reflecting the global importance of each channel, i.e., the degree of contribution of each channel to a particular task. The compression step is specified by the Equation 7.

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{l=1}^{H} \sum_{j=1}^{W} u_c(i,j) \tag{7}$$

where $u_c \in R^{H \times W}$ represents the cth channel of U.

In the excitation step, the SE attention mechanism learns the weights of each channel through a fully connected layer. The network receives global features from the compression step as input and outputs a vector of channel attention weights. The specific the Equation 8 for the excitation operation:

$$s = F_{ex}(z,W) = \sigma(g(z,W)) = \sigma(W_2 \delta(W_1 z)) \tag{8}$$

where vector s represents the importance level of each feature map, δ represents the ReLU function, σ represents the Sigmoid activation function, $W_1 \in R^{\frac{C}{r}}$, $W_2 \in R^{\frac{C}{r}}$, and r is the scaling ratio, and the vector s is multiplied by the original feature maps to weight each channel, as specified in the Equation 9:

$$\tilde{X}_c = F_{\text{scale}}(u_c, s_c) = s_c u_c \tag{9}$$

He and Jiang (2021) built an SE-ResNet model for bone age assessment of hand X-ray imaging, so that the model would not lose important information in raw images. Kong et al. (2023) built the SE-ResNet model to form a complete glomerular classification framework to classify glomerular lesions. Liu et al. (2024) built an SE-ResNet model to classify heartbeat between patients, so that the model can effectively learn the long-term characteristics of heartbeat between patients. It can be seen from the above literature that in different fields, the authors combine SE module with ResNet model, so that the classification ability of the model gets a good effect. Therefore, we apply this combination to music signal classification, in order to make the model pay better attention to the important feature information in the Log Mel spectrum and pay better attention to the long distance feature information in the Log Mel spectrum.

The SE layer is placed after the two convolutional layers of the Res Net backbone network, and the jump connections are connected to the output of the SE. The improvement is to replace the fully connected layer in the SE with a 1 × 1 convolution. Firstly, the convolutional layer learns and adjusts the channel attention by sharing parameters. Secondly, the 1 × 1 convolutional layer retains the spatial information when feature extraction is performed on the feature map, and the attention weight of each channel can be adjusted with the spatial information, which is conducive to capturing the spatial relationship between features better. The fully connected layer, on the other hand, spreads the input into vectors without considering the spatial structure of the input Log Mel spectrum and loses the spatial information, so it is not suitable for the processing of the Log Mel spectrum. Finally the convolutional layer has the properties of local connectivity and

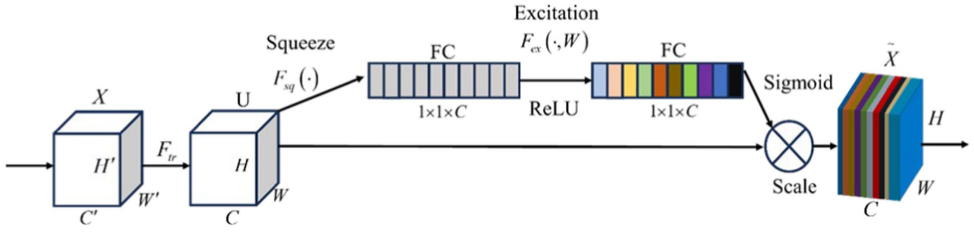


FIGURE 9
Squeeze-and-excitation module structure.

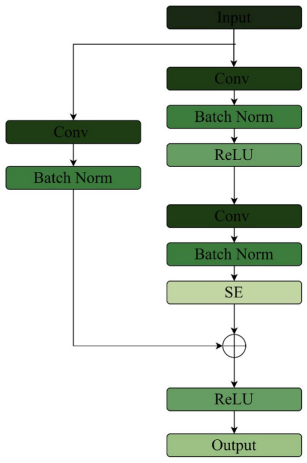


FIGURE 10
SE-res module structure.

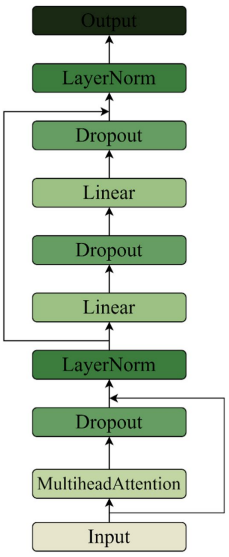


FIGURE 11
Transformer encoder module structure.

parameter sharing, which can simultaneously process input data at multiple locations and improve the model generalization ability. The structure of SE-Res module is shown in Figure 10.

3.2.4 Transformer encoder module

Transformer's multi-headed self-attention layer is able to look at multiple previous time steps when predicting the next time step, and the model is able to capture dependencies over longer distances. The music genre classification task needs to take into account longer contextual information, such as the connection between the intro and chorus parts. Transformer obtains the positional information of the input music sequence through positional encoding, which allows the model to distinguish between different positional elements, with different timesteps representing different pieces of music. Compared to RNN networks, Transformer's property of establishing connections over the entire range of sequences is more suitable for the music genre classification task.

CNNs have excellent performance in extracting spectral features of music, but the network design of a CNN concatenated with an RNN loses time-series information during the extraction of Log Mel spectrum features. Therefore, a parallel Transformer module is used to complement the extraction of time-series information from Log Mel spectrums, while avoiding the problem of time-series information loss in CNN and RNN models. Each encoder block has 4 self-attention layers in each multi-head self-attention layer, and each encoder block has 2 linear layers in the feed-forward network. The Feed Forward is set to 512, and the Drouput is set to 0.4. The structure of the Transformer Encoder is shown in Figure 11.

TABLE 1 Experimental environment.

	Environment configuration
Operating system	Windows 10
CPU	AMD Ryzen 95900HX with Radeon Graphics 3.30 GHz
GPU	NVIDIA GeForce GTX 3080 CUDA 12.1

4 Experiments

4.1 Experimental environment

The TS-Resformer model we designed is built on the PyTorch framework in Python3.8 environment. Comparison experiment and ablation experiment are also implemented in PyTorch framework based on Python3.8 environment. The detailed experimental environment is shown in Table 1.

In the design process of TS-Resformer model, we selected the Adam optimizer, set the learning rate to 10^{-3}, set the Batch size to 32, took the multi-class cross-entropy loss function as the loss function, and ReLU as the activation function. As for data preprocessing, we adopted the music data preprocessing method in Section 3.1, applied short-time Fourier transform to each processed music audio segment, the window function was Hanning window, the window size was 4,096, the jump length was set to 1,024, and the slice length was 12 s, with 50% repeat segments. The audio is processed according to the method described in Section 3.1, the Log Mel spectrum is generated, and the Log Mel spectrum is input into the TS-Resformer model for subsequent classification tasks.

4.2 Evaluation metrics

We choose the accuracy rate, confusion matrix and F1- Score as the evaluation indexes of the experiment, and use them as the basis for comparing the experimental effects.

Accuracy rate refers to the proportion of correctly predicted samples to the total number of samples by the classifier, and the formula for calculating the accuracy rate is shown below and the Equation 10:

$$Accuracy = \frac{1}{k}\sum_{i=1}^{k}\frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{10}$$

where TP_i, TN_i, FP_i, FN_i, are True Positive, False Positive, False Negative, False Negative, respectively, and the variable k denotes the number of classes. Since the dataset in this paper is balanced, macro-averaging is used, a method that calculates the metric for each class and then averages it, assigning the same weight to each class.

Precision is the ratio of the number of samples correctly predicted as positive classes to the number of all samples predicted as positive classes. It measures the proportion of samples that are truly positive categories out of all samples predicted as positive categories by the model, i.e., the accuracy of the model's predictions. The formula for calculating the accuracy rate is shown below the Equation 11 for:

$$Precision = \frac{1}{k}\sum_{i=1}^{k}\frac{TP_i}{TP_i + FP_i} \tag{11}$$

Recall is the ratio of the number of samples correctly predicted to be in the positive category to the number of samples in all true positive categories, and measures the degree of coverage of the model for samples in the positive category, i.e., the model's ability to recognize samples in the positive category. The Equation 12 for recall is shown below:

$$Recall = \frac{1}{k}\sum_{i=1}^{k}\frac{TP_i}{TP_i + FN_i} \tag{12}$$

F1-Score is the reconciled average of precision and recall, which is a comprehensive evaluation of the performance of the classification model, and is a weighted average of precision and recall, which can consider the accuracy and coverage of the model at the same time. The formula for calculating the F1-Score is shown below The Equation 13 for:

TABLE 2 Confusion matrix.

	Real positive results	Real negative results
Predicting positive outcomes	TP	FP
Predicting negative outcomes	FN	TN

$$F1 - Score = 2\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{13}$$

The confusion matrix is shown in Table 2.

4.3 Analysis of experimental results

4.3.1 Training process

The classification accuracy of the TS-Resformer model in each school of thought on the training set is shown in Figure 12.

It can be seen from Figure 12 that during the training process of our TS-Resformer model, around the 46th round, the accuracy rate of the model for various music categories has tended to be stable. This indicates that the model we proposed can converge at a relatively fast speed and achieve satisfactory results.

4.3.2 The effect of different slicing lengths of music on model classification accuracy and training time

We design the audio duration slicing method with overlap rate, slicing the Equation 14.

$$N = \frac{L}{\lambda I} - 1 \tag{14}$$

Where N is the number of segments obtained by slicing an audio segment, L is the length of the original audio, the datasets used in this paper are all 30s segments, so L is 30s. λ is the overlap rate, which is set to 50% in this paper. i is the length of the segment obtained by slicing, and the complementary zero operation is performed for the case of insufficient sampling points. According to this formula, 30s audio can be sliced into [3 s,5 s,6 s,10s,12 s,15 s] according to the 50% overlap rate, and the corresponding number of segments is (Choi et al., 2017; Orjesek et al., 2022; Ali and Siddiqui, 2017; Sarkar and Saha, 2015; Baniya et al., 2014; Logan, 2000).

The short duration of the segment leads to insufficient feature extraction, and the whole audio input significantly increases the computational cost and affects the model design. In order to verify the effect of slicing audio segments of different durations on the experiments, experiments are conducted on the FMA-small dataset with the above audio slicing method. The experimental results are shown in Figure 13.

As the slice length increases, the total number of slices decreases and the training time decreases. a slice length of 3 s has higher accuracy, but makes the dataset extremely large, the training time the longest, and requires more equipment. As the slice length increases, the training time decreases. In the case of 12 s slice length, the accuracy is 0.36% different from the 3 s case.

Therefore, considering the above, we use 12 s to slice the 30s audio file into 4 segments with 50% overlap.

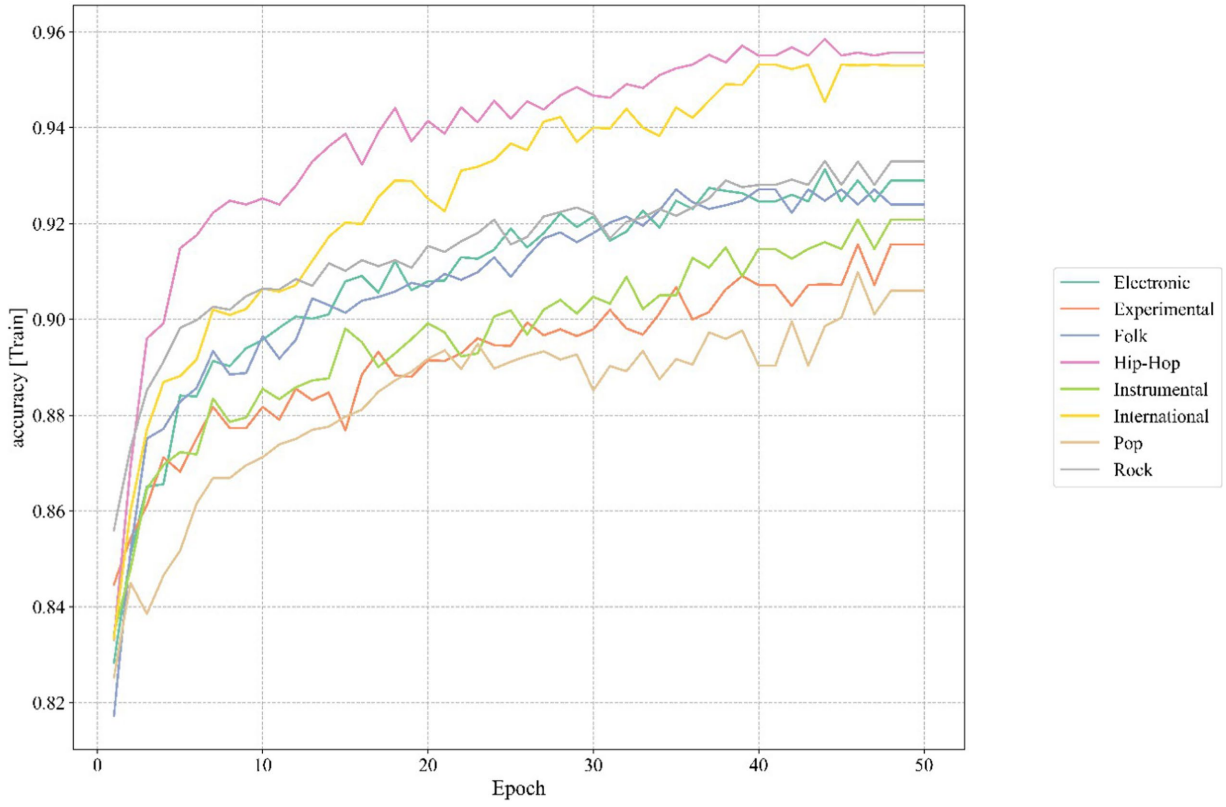


FIGURE 12
Training process.

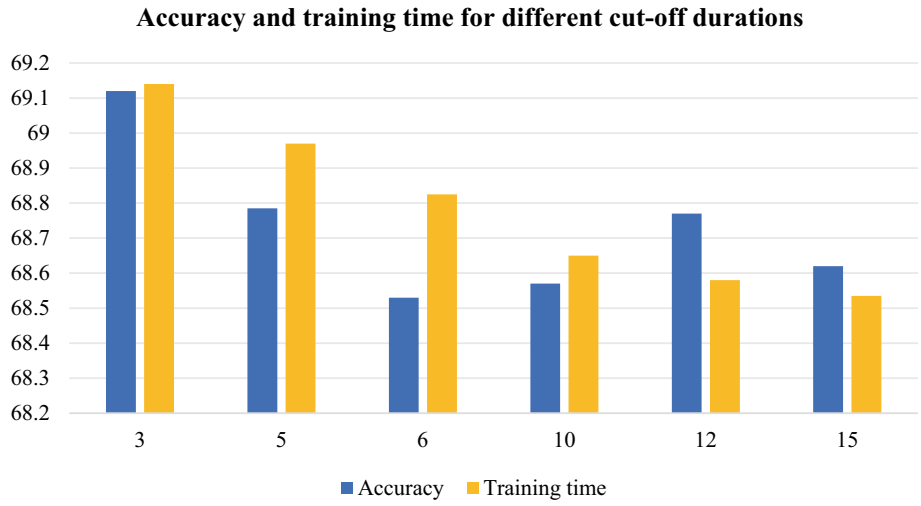


FIGURE 13
Accuracy and training time of different segmentation duration.

4.3.3 Multi-scale convolutional filter shape determination

On the FMA-small dataset, we selected four music genres with poor classification effects to carry out experiments to explore the effect of different shapes of filters on the classification effect under the same experimental environment, and the evaluation index of the experiments is the accuracy rate. Firstly, the filter width is fixed at 2, and the height of the filter is adjusted between 2 and 15, and the experimental results are shown in Figure 14.

Then the height of the filter is fixed to 2, and the width of the filter is adjusted between 2 and 15, and the experimental results are shown in Figure 15.

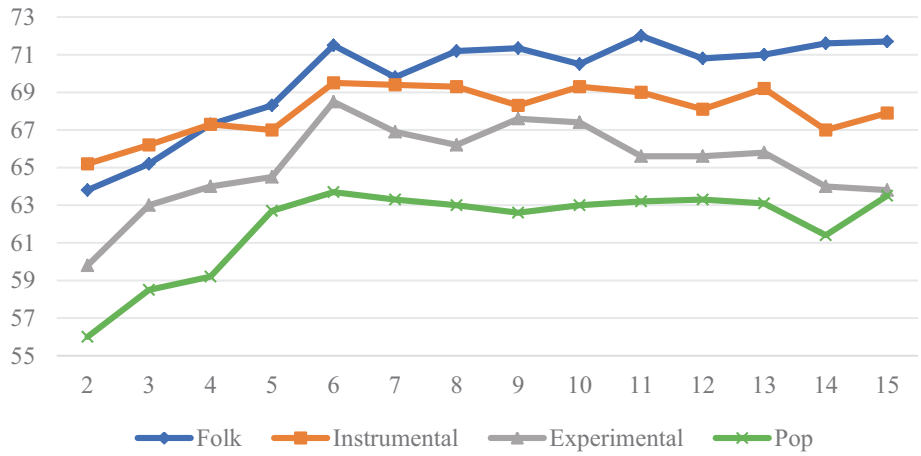


FIGURE 14
Classification accuracy of different filter heights.

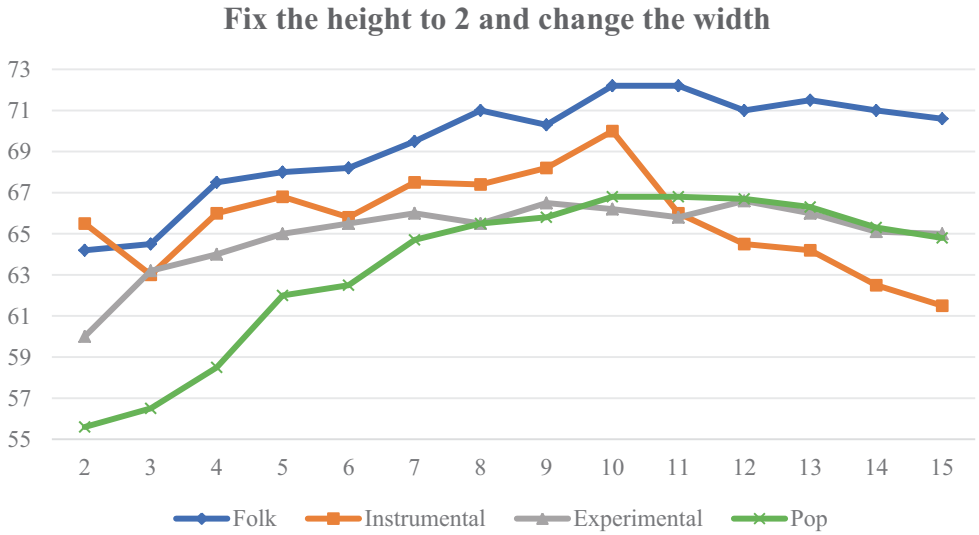


FIGURE 15
Classification accuracy of different filter widths.

Figures 13, 15 are analyzed as follows:

(1) As the height (frequency) increases, the accuracy of the four categories also increases. The overall trend flattens out at a height of about 6, beyond which further increases in frequency range do not result in significant improvements.
(2) When the width (time) is increased, the sensory field time span becomes larger, and Instrumental first increases to a peak and then decreases rapidly, suggesting that Instrumental is expressed by short-term characterization, and that the overall accuracy is highest at a width of 10.

In summary, we chose 6×2 and 2×10 filters to apply to the Conv1 layer of the TF-Attention module.

4.3.4 Ablation experiments

The classification accuracy of our proposed time-frequency attention mechanism, SE-Res module, and different network architectures on the FMA-small dataset is verified by ablation experiments, the results of which are shown in Table 3.

As can be seen from Table 3, by comparing 1 and 2, in terms of the two evaluation metrics of accuracy and F1-Score, the parallel structure we use is higher than the serial structure by 1.01% and 0.032, respectively, which indicates that the feature extraction of the parallel structure is more adequate than that of the serial structure, so that the model acquires sufficient features and improves the accuracy. By comparing 2 and 3, in terms of accuracy and F1-Score, the addition of our proposed TF-Attention module improves 1.51% and 0.019 respectively, indicating that the TF-Attention module

TABLE 3 Ablation experiments.

	Serial structure	Parallel structure	TFA	SE-res module	Accuracy	*F1 score*
1	√				87.64%	0.565
2		√			88.65%	0.593
3		√	√		90.16%	0.612
4		√		√	89.92%	0.603
5		√	√	√	90.23%	0.647

further enables the model to focus on the main feature information in both the time and frequency domains. By comparing 2 and 4, the proposed SE-Res module improves the accuracy and F1-Score by 1.3% and 0.01 respectively, which indicates that the SE-Res module further focuses the model on the global feature information and improves the model's expressive ability. In order to verify the rationality of selecting four Transformer encoder layers for our model, we conducted ablation experiments. We selected a different number of Transformer encoder layers for the model. A is a model containing one Transformer encoder layer, B is a model containing two Transformer encoder layers, C is a model containing three Transformer encoder layers, D is the model proposed in this paper, and E is a model containing five Transformer encoder layers. The experimental results are shown in Table 4. The experimental results show that the accuracy rate increases as the number of selected layers increases. When the number of layers increased to 4 layers, the accuracy reached the peak, and continued to increase the number of layers, the accuracy declined. According to the experiment, when the model chooses 4 Transformer encoder layers, the model has the best music signal classification ability.

4.3.5 Comparison experiment

In this experiment, we evaluate the performance of several classical network structures and their combinations on the FMA-small dataset, including LSTM, GRU, Bi-GRU, ResNet, Transformer, and various combinations of them. Among them, we combine ResNet with LSTM, GRU and Bi-GRU in parallel, and the experimental results are shown in Table 5.

The experimental results show that among all the tested models, the one in which we parallelized Transformer with the SE-Res module exhibited the highest accuracy and F1 score, and this design not only improves the expressive power of the model, but also enables the model to learn the nuances in the data more efficiently, thus achieving the best classification performance. This parallel combination not only utilizes the powerful feature extraction capabilities of the SE-Res module and the TF-Attention module, effectively solves the gradient vanishing problem in the deep network through its internal residual learning mechanism, and enhances the learning of feature representations in both the time and frequency domains, but also takes advantage of the Transformer encoder's property of establishing connections throughout the entire range of sequences, allowing the model to distinguish between different positional elements and different time steps representing different music segments.

In conclusion, the experimental results show that our proposed multimodal fusion structure can significantly enhance the model and provide valuable directions for future research.

TABLE 4 Ablation experiments.

	Number of layers	Accuracy
A	1	87.92%
B	2	88.25%
C	3	89.69%
D	4	90.23%
E	5	89.93%

TABLE 5 Comparison experiments.

Model	Accuracy	*F1 score*
LSTM	69.21%	0.462
GRU	72.32%	0.495
Bi-GRU	73.68%	0.523
Res Net	79.21%	0.562
Transformer	74.52%	0.538
Res Net + LSTM	80.22%	0.586
Res Net + GRU	82.15%	0.592
Res Net + Bi-GRU	86.21%	0.587
SE-Res Net + Transformer Encoder	90.23%	0.647

5 Conclusion and limitation

5.1 Conclusion

Relying on Internet platforms and intelligent terminals, the digital music market has achieved rapid growth in the number of music releases and the number of digital music users. At the same time, competition among music streaming platforms has become increasingly fierce, with new features and services constantly launched among the various platforms to attract more users and improve user retention. The classification of a large number of music genres is conducive to the realization of personalized recommendation on music platforms, improve people's satisfaction with songs pushed by music platforms, and promote the development of music market. In order to improve the accuracy of music type classification, we propose the TS-Resformer model, which mainly completes the following tasks:

(1) In the data processing part, we propose an audio slicing method with different time durations. This method solves the problem that too long audio duration in the data set is not conducive to model training, and at the same time, by comparing the effects

of different slicing durations on the accuracy and training time of music genre classification, we determine the optimal slicing duration and realize data enhancement.

(2) In order to obtain comprehensive and effective music audio feature information and better realize music genre classification, we propose a parallel music genre classification model Res-Transformer based on residual network and Transformer Encoder, which improves the residual network instead of convolution operation, improves the ability of the network to extract features of Log Mel spectrum, and alleviates the problem of gradient disappearance. The model improves the feature extraction capability of the network for Log Mel spectrum by improving the residual network instead of the convolution operation, which alleviates the problem of gradient disappearance.

(3) Aiming at the problem of insufficient feature extraction in the Res-Transformer model, which leads to the low classification accuracy of individual genre, the TS-Resformer model is established by integrating the time-frequency and channel attention mechanisms. The feature extraction layer of the model is designed to capture features from both time and frequency dimensions using filters of different scales, and the filter scales are designed for music genres with low accuracy rates of individual genres, so as to extract low-level music features from specific time and specific frequency. The output of the time-frequency convolution is used as the input of the time-frequency attention mechanism, and the time-frequency attention mechanism applicable to music classification is designed to assign weights to the extracted music features in the time-frequency dimension, which mitigates the problem of insufficient music feature extraction and the problem of low accuracy rate of classification of individual category genres. By fusing the residual block with the improved channel attention mechanism, the weighting of each channel of the time-frequency feature map is realized, which weakens the features that are not related to music classification and reduces the complexity of the global features after feature fusion.

5.2 Limitation

With the gradual expansion of the global music market, the task of music classification in the future should not only require higher classification accuracy, but also achieve non-exclusive music genre classification according to music styles. Therefore, there are still many aspects that could be improved in this paper:

(1) The method of classifying music genres by deep learning is mainly to convert music into Log Mel spectrum like ordinary audio, and then use the feature extraction capability of convolution for images. It is necessary to further study effective music data processing methods based on music characteristics.

(2) This paper is mainly based on single label classification, but in fact, some concerts have the characteristics of multiple genres, which cannot be simply classified into one class. In the future, it is necessary to establish or select multi-label data sets and further study music genre classification.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found at: https://github.com/mdeff/fma.

Author contributions

YZ: Data curation, Funding acquisition, Investigation, Methodology, Project administration, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. This work was supported by 2023 Dalian University of Foreign Languages Research Fund project results 2024XJXM32.

Conflict of interest

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Gen AI was used in the creation of this manuscript.

Publisher's note

References

Ali, M. A., and Siddiqui, Z. A. (2017). Automatic music genres classification using machine learning. *Int. J. Adv. Comput. Sci. Appl.* 8, 188–197. doi: 10.14569/IJACSA.2017.080844

Arabi, A. F., and Lu, G. (2009). "Enhanced polyphonic music genre classification using high level features" in 2009 IEEE international conference on signal and image processing applications (Kuala Lumpur, Malaysia: IEEE), 101–106. doi: 10.1109/ICSIPA.2009.5478635

Baniya, B K, Ghimire, D, and Lee, J. (2014). A novel approach of automatic music genre classification based on timbrai texture and rhythmic content features. 16th international conference on advanced communication technology. Pyeong Chang, Korea: IEEE 96–102. doi: 10.1109/ICACT.2014.6778929

Chen, J., Su, P., Li, D., Han, J., Zhou, G., and Tang, D. (2024). Optimizing fractional-order convolutional neural networks for groove classification in music using differential evolution. *Fractal Fract.* 8:616. doi: 10.3390/fractalfract8110616

Choi, K, Fazekas, G, Sandler, M, et al., (2017). Convolutional recurrent neural networks for music classification. 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). Washington: IEEE. 2392–2396. doi: 10.1109/ICASSP.2017.7952585

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). Pattern classification. USA: John Wiley & Sons, 5–6.

He, J., and Jiang, D. (2021). Fully automatic model based on se-resnet for bone age assessment. *IEEE Acc.* 9, 62460–62466. doi: 10.1109/ACCESS.2021.3074713

Khasgiwala, Y, and Tailor, J. (2021). Vision transformer for music genre classification using mel-frequency cepstrum coefficient. 2021 IEEE 4th international conference on computing, power and communication technologies (GUCON). Kuala Lumpur: IEEE: 1–5.

Kong, X. Y., Zhao, X. S., Sun, X. H., Wang, P., Wu, Y., Peng, R. Y., et al. (2023). Classification of glomerular pathology images in children using convolutional neural networks with improved SE-ResNet module. *Interdiscip. Sci.: Comput. Life Sci.* 15, 602–615. doi: 10.1007/s12539-023-00579-7

Kostrzewa, D., Kaminski, P., and Brzeski, R. (2021). "Music genre classification: looking for the perfect network" in International conference on computational science (Sanya: IEEE), 55–67.

Li, Y., Zhang, Z. H., Ding, H., et al. (2023). Music genre classification based on fusing audio and lyric information. *Multimed. Tools Appl.* 82, 20157–20176. doi: 10.1007/s11042-022-14252-6

Lin, Q. (2022). Music score recognition method based on deep learning. *Comput. Intell. Neurosci.* 2022, 1–12. doi: 10.1155/2022/3022767

Liu, J., Liu, Y., Li, Z., Qin, C., Chen, X., Zhao, L., et al. (2024). A novel diagnosis method combined dual-channel SE-res net with expert features for inter-patient heartbeat classification. *Med. Eng. Phys.* 130:104209. doi: 10.1016/j.medengphy.2024.104209

Logan, B. (2000). Mel frequency cepstral coefficients for music modeling. *Ismir. Plymouth* 270, 11–12.

Orjesek, R., Jarina, R., and Chmulik, M. (2022). End-to-end music emotion variation detection using iteratively reconstructed deep features. *Multimed. Tools Appl.* 81, 5017–5031. doi: 10.1007/s11042-021-11584-7

Ostermann, F., Vatolkin, I., and Ebeling, M. (2023). AAM: a dataset of artificial audio multitracks for diverse music information retrieval tasks. *Eurasip J. Audio Speech Music Process.* 2023:13. doi: 10.1186/s13636-023-00278-7

Pelchat, N., and Gelowitz, C. M. (2020). Neural network music genre classification. *Can. J. Electr. Comput. Eng.* 43, 170–173. doi: 10.1109/CJECE.2020.2970144

Prabhakar, S. K., and Lee, S. W. (2023). Holistic approaches to music genre classification using efficient transfer and deep learning techniques. *Expert Syst. Appl.* 211:118636. doi: 10.1016/j.eswa.2022.118636

Sarkar, R., and Saha, S. K. (2015). Music genre classification using EMD and pitch based feature. 2015 eighth international conference on advances in pattern recognition (ICAPR). New York, USA: IEEE: 1–6.

Solanki, A., and Pandey, S. (2022). Music instrument recognition using deep convolutional neural networks. *Int. J. Inf. Technol.* 14, 1659–1668. doi: 10.1007/s41870-019-00285-y

Sturm, B. L. (2012). An analysis of the gtzan music genre dataset. Proceedings of the second international ACM workshop on music information retrieval with user-centered and multimodal strategies. *Nara*, 7–12. doi: 10.1145/2390848.2390851

Tzanetakis, G., and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Trans. Speech Audio Proc.* 10, 293–302. doi: 10.1109/TSA.2002.800560

Vaibhavi, M., and Krishna, P. R. (2021). Music genre classification using neural networks with data augmentation. *J. Innov. Sci. Sustain. Technol* 1, 21–37.

Verma, V., Benjwal, A., Chhabra, A., Singh, S. K., Kumar, S., Gupta, B. B., et al. (2023). A novel hybrid model integrating MFCC and acoustic parameters for voice disorder detection. *Sci. Rep.* 13:22719. doi: 10.1038/s41598-023-49869-6

Wen, Z., Chen, A., Zhou, G. X., et al. (2024). Parallel attention of representation global time-frequency correlation for music genre classification. *Multimed. Tools Appl.* 83, 10211–10231. doi: 10.1007/s11042-023-16024-2

Xi, S, Xu, C S, and Kankanhalli, M S. (2004). Unsupervised classification of music genre using hidden Markov model. 2004 IEEE international conference on multimedia and expo (ICME), Taibei, China: IEEE: 2023–2026.

Xie, C. J., Song, H. Z., Zhu, H., et al. (2024). Music genre classification based on res-gated CNN and attention mechanism. *Multimed. Tools Appl.* 83, 13527–13542. doi: 10.1007/s11042-023-15277-1

Yang, R., Feng, L., Wang, H. B., et al. (2020). Parallel recurrent convolutional neural networks-based music genre classification method for mobile devices. *IEEE Access* 8, 19629–19637. doi: 10.1109/ACCESS.2020.2968170

Zhai, X., and Luo, Y. (2025). Can urban internet development attract labor force? Evidence from Chinese Cities. *Sustainability* 17:260. doi: 10.3390/su17010260

Zhang, Z., He, Y., Mai, W., Luo, Y., Li, X., Cheng, Y., et al. (2024). Convolutional dynamically convergent differential neural network for brain signal classification. *IEEE Trans. Neural Netw. Learn. Syst.*, 1–12. doi: 10.1109/TNNLS.2024.3437676

Zhang, W. B., Lei, W. K., Xu, X. M., et al. (2016). "Improved music genre classification with convolutional neural networks" in Interspeech (San Francisco: IEEE), 3304–3308.

Zhang, Z., Zhang, J., and Mai, W. (2025). VPT: video portraits transformer for realistic talking face generation. *Neural Netw.* 184:107122. doi: 10.1016/j.neunet.2025.107122

Zhuang, Y. Y., Chen, Y. Z., and Zheng, J. (2020). Music genre classification with transformer classifier. Proceedings of the 2020 4th international conference on digital signal processing. *Xiamen IEEE*, 155–159.

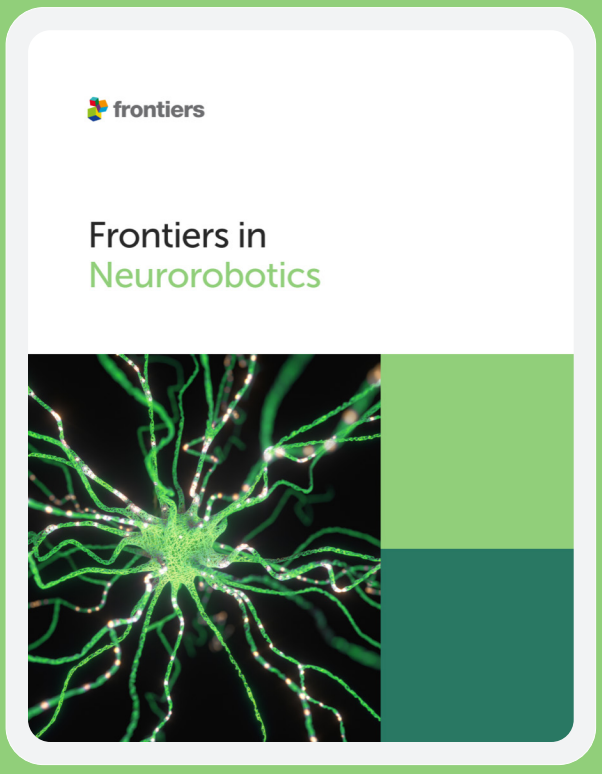
Frontiers in Neurorobotics

Investigates embodied autonomous neural systems and their impact on our lives

Part of the most cited neuroscience series, this journal advances understanding of neurorobotics - from prosthetic devices to brain machine interfaces, and wearable systems to home appliances.

Discover the latest Research Topics

See more →



Frontiers

Avenue du Tribunal-Fédéral 34
1005 Lausanne, Switzerland
frontiersin.org

Contact us

+41 (0)21 510 17 00
frontiersin.org/about/contact