TYPE Methods
PUBLISHED 09 January 2023
DOI 10.3389/fnbot.2022.978225

OPEN ACCESS

EDITED BY
Francesco Rundo,
STMicroelectronics, Italy

REVIEWED BY
Dechao Chen,
Hangzhou Dianzi University, China
Ye Yuan,
Southwest University, China

*CORRESPONDENCE
Weimin Lei
✉ leiweimin@mail.neu.edu.cn

RECEIVED 25 June 2022
ACCEPTED 08 December 2022
PUBLISHED 09 January 2023

CITATION
Yang L, Lei W, Zhang W and Ye T (2023) Dual-flow network with attention for autonomous driving. *Front. Neurorobot.* 16:978225. doi: 10.3389/fnbot.2022.978225

Dual-flow network with attention for autonomous driving

Lei Yang[1,2], Weimin Lei[1,3]*, Wei Zhang[1] and Tianbing Ye[2]

[1]School of Computer Science and Engineering, Northeastern University, Shenyang, China, [2]DAMO Academy, Alibaba Group, Hangzhou, China, [3]Engineering Research Center of Security Technology of Complex Network System Ministry of Education, Shenyang, China

We present a dual-flow network for autonomous driving using an attention mechanism. The model works as follows: (i) The perception network extracts red, blue, and green (RGB) images from the video at low speed as input and performs feature extraction of the images; (ii) The motion network obtains grayscale images from the video at high speed as the input and completes the extraction of object motion features; (iii) The perception and motion networks are fused using an attention mechanism at each feature layer to perform the waypoint prediction. The model was trained and tested using the CARLA simulator and enabled autonomous driving in complex urban environments, achieving a success rate of 74%, especially in the case of multiple dynamic objects.

KEYWORDS

autonomous driving, network architecture, artificial intelligence, CARLA simulator, attention, visual navigation, deep neural network

1. Introduction

Autonomous driving, especially in a complex environment, remains a challenging subject. However, with the evolution of deep networks, breakthroughs have been made in autonomous driving techniques using images as input (Chen et al., 2015; Dosovitskiy et al., 2017; Liang et al., 2018; Li et al., 2018; Sauer et al., 2018). These studies have been broadly based on using images as input, extracting features through deep neural networks, and then converting feature mapping to the vehicle's low-level commands. However, the image is only a spatial representation of the environment $I(x, y)$, so we lose the temporal dimension of the video, which is also crucial for the understanding of motion. Therefore, we attempt to add the consideration of temporal dimension and introduce the method in the domain of video recognition to autonomous driving for completing the extraction of space–time features of the surrounding environment. This raises the first question: *Are the spatial and temporal characteristics of the video equally important?* Weiss et al. (2002) showed that the human retina had a different sensitivity to understanding scene and moving objects, where scene understanding needs to be precise but changes slowly, while the perception of motion of corresponding objects is the opposite. Motivated by Weiss et al. (2002) and Feichtenhofer et al. (2019), we ensure that the perception network maintaining the high-resolution spatial features of the image, while the motion network focuses on real time using a more lightweight network. The ablation experiments (shown in Table 4) demonstrated that a ratio of 8:1 between the number of feature channels of the perception network and the motion network worked best.

The perceptual and motion features of the video are a whole in themselves, which leads to the second question to be solved: *How can we integrate the perceptual features with the motion features?* The attention mechanism has received more focus in recent years owing to its success in natural language processing (NLP), and there have been increased developments in the image domain as well (Dosovitskiy et al., 2020; Liu et al., 2021). Inspired by these studies, we ensure that the dual-flow network performs attention learning in each feature layer (except the first layer) and that the high-resolution perception network aligns motion features to more accurately perceive the motion of objects in a complex environment, such as pedestrian intrusion.

The overall model structure is shown in Figure 1. The vehicle's front camera is used as input, and the features are extracted through a dual-flow network to perform the waypoint prediction, which is finally mapped to the vehicle's low-level commands (steering, throttle, and brake) by proportional–integral–derivative (PID) controllers. As far as we know, the attention-based dual-flow network is being used for the first time in autonomous driving. Our main contributions are as follows: (1) Proposing the dual-flow network to extract space–time features from video; (2) Using an attention mechanism to fuse space–time features in the dual-flow network; and (3) Conducting experiments to demonstrate the network's effectiveness. Using the CARLA (Dosovitskiy et al., 2017) simulator, we can adapt the model to the complex urban environment for autonomous driving, with a success rate of 74%.

2. Related work

2.1. Autonomous driving

Prior approaches for autonomous driving with computer vision are classified into three groups: modular pipelines (MP), imitation learning (IL) (Hussein et al., 2017), and direct perception (DP) (Gibson, 2014).

Modular pipelines are the relatively more traditional and longer studied approach. The task is generally divided into three sub-modules: a perception module, a planning module, and a control module. Among them, the perception module is the core of the whole system and is commonly used for the identification of lanes, fences, dynamic objects, and other hazards, providing inputs for subsequent modules. Geiger et al. (2013) investigated the various sub-tasks. Felzenszwalb et al. (2010) and Lenz et al. (2011) detected vehicles and lanes, respectively. Lin et al. (2017) attempted to segment the image. The outputs of the perception modules of these methods cannot be used directly for autonomous driving tasks and need to be subsequently mapped. As MP relies on many intermediate feature representations, multiple intermediate feature accuracies limit the accuracy of the entire system, which is more challenging in complex environments.

Imitation learning is a bionic approach also called behavioral cloning, which maps the raw sensor inputs directly to the vehicle's low-level control commands, and is an end-to-end learning approach. Our model also belongs to this category. First, Pomerleau (1988) and Muller et al. (2005) used the camera as the main input and used a neural network to predict the related actions. Codevilla et al. (2018) used high-level navigation commands to eliminate ambiguity at intersections for autonomous driving, opening up a new stage of navigation. Li et al. (2018) enabled better generalization of the model by adding branches that assist in predicting image depth and image segmentation. Codevilla et al. (2019) added speed prediction while exploring some limitations of autonomous driving. Chen et al. (2020) proposed a dual-network structure of "teacher" and "student" using the knowledge distillation method and optimized the "student" network through the "teacher" network, achieving good results. A multimodal fusion study by Xiao et al. (2020) was attempted and demonstrated that early feature fusion helped in feature learning. Research related to our work (Prakash et al., 2021) incorporated both image and light-detection-and-ranging (LiDAR) modalities with an attention mechanism, while our model focuses on the mining of video space–time features. Both the modality and the network architecture are different.

Direct perception is an intermediate form of MP and IL, which decompose the autonomous driving task into several key metrics, enhancing interpretation without relying on an intermediate representation of the environment. Chen et al. (2015) attempted to map the camera input to 13 metrics related to the final decision. Sauer et al. (2018) mapped images directly to six sub-tasks emergency stop, red light, speed marker, vehicle distance, steering angle, and lane departure, and those metrics were converted to throttle and brake commands by rules. The DP approach requires a manual setting of prediction sub-tasks, which is difficult to accomplish in complex environments.

2.2. Space–time features

In the image domain, convolutional neural networks (CNN) have dominated, including AlextNet (Krizhevsky et al., 2012), VGG (Simonyan and Zisserman, 2014b), ResNet (Szegedy et al., 2017), and ConvNeXt (Liu et al., 2022), and have now become the base networks for extracting image features.

Recently, there have also been some major developments in the video domain. The network of two-dimensional architectures includes the following: DeepVideo (Karpathy et al., 2014), TwoStreamNet (Simonyan and Zisserman, 2014a) and TSN (Wang et al., 2016), and a series of studies. The network of three-dimensional architectures includes the following: C3D (Tran et al., 2015), I3D (Carreira and Zisserman, 2017), and

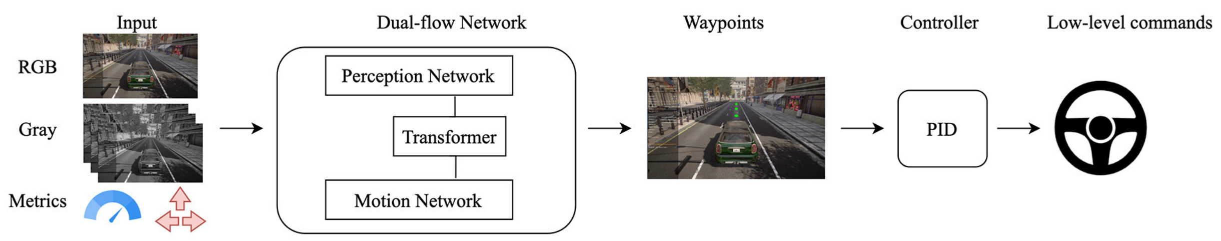

FIGURE 1
Overview of the structure, including the video from the camera, vehicle speed, and high-level navigation commands as input, using a dual-flow network to extract features and output the prediction of waypoints, which are converted to the vehicle's low-level commands through the proportional-integral-derivative (PID) controller.

a series of studies. Transformer network architecture contains TimesTransformer (Bertasius et al., 2021) and ViT (Girdhar et al., 2019).

In the temporal dimension of video, the optical flow defines the movement of objects in an image, specifically the amount of movement of pixels representing the same object in one frame of a video image to the next frame. The most classical traditional optical flow algorithm is that proposed by Lucas and Kanade (1981), which has been widely used given its luminance invariance assumption and neighborhood optical flow similarity assumption, and has been integrated into the OpenCV library. Considering the computational inefficiency of traditional optical flow algorithms, in recent years, people have tried to estimate the optical flow using deep neural networks (DNNs), represented by algorithms such as FlowNet2 (Ilg et al., 2017). In the field of autonomous driving, optical flow also has a wide range of applications. Gern et al. (2002) used optical flow to do lane recognition under adverse weather conditions. In Lieb et al. (2005), the optical flow was used within a road-following algorithm that allows for identifying the road. In Okafuji et al. (2015), steering learning was performed in in-car driving. Camus (1995) performed obstacle avoidance using optical flow. In Capito et al. (2020), the visual potential field was calculated using optical flow vectors for obstacle avoidance. Optical flow distillation paradigms have also recently emerged for application in the field of autonomous driving (Rashed et al., 2019). However, instead of using optical flow as input, our motion network tries to have the network automatically learn motion-related features and then fuse them with the individual feature layers of the perceptual network features. The network features we learn are more related to autopilot-related motion features and are not different from the optical flow feature as a whole (e.g., large changes in building illumination are not relevant for autopilot features).

Our dual-flow network is motivated by a two-stream net and slow-fast net, but has a different input and network architecture from either of them. Our inputs contain RGB images and continuous video frames (grayscale maps), while the feature network uses a modified ResNet network to maintain high resolution and guarantee feature extraction of environmental details and motion.

2.3. Attention mechanisms

An attention (Vaswani et al., 2017) mechanism can learn the attention of features dynamically based on different predicted targets. The earlier applications of attention mechanisms were mainly in the fields of NLP, but recently they have evolved significantly in both image and video domains, from the early squeezeNet (Iandola et al., 2016) and non-local (Wang et al., 2018) to transformer. An attention mechanism is distinctly different in the three domains of NLP, computer vision, and image recognition. The first two are closely related to the temporal dimension, while the input of the third one is a static picture without a clear sequence. Our model uses the attention mechanism in the field of computer vision to accomplish the fusion of dual-flow network features.

3. Method

The overall system is an end-to-end learning framework, as shown in Figure 2. The RGB images sampled at low sample rates are used as input to the perception network, and the grayscale images at high sample rates are used as input to the motion network. The dual-flow network extracts features and fuses them through an attention mechanism to output features, concatenates the speed features, and then completes the selection of the corresponding branches according to the high-level navigation commands to output the waypoints. Finally, the waypoints are mapped to the low-level vehicle commands by the PID controllers. The details are described in later sections.

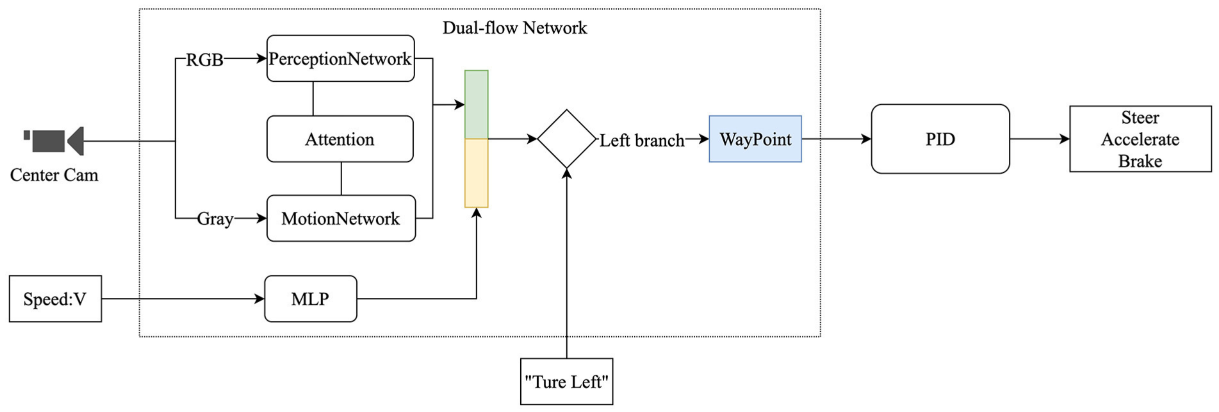

FIGURE 2
Flow chart of the overall system. The process has three steps: (1) Using the vehicle's front camera and speed as inputs, the low-sampling RGB image is used as input to the perception network, and the high-sampling grayscale images are used as input to the motion network; (2) The dual-flow network overlapping the high-level navigation commands performs the prediction of waypoints; and (3) The final output of the vehicle's low-level control commands is produced through the PID controllers.

3.1. Problem setting

Autonomous driving can be seen as a bionic learning or behavior cloning process. The process is seen as a supervised learning problem, where the behavior of an expert is imitated through a deep network. First, the data of the expert-driven vehicle are collected in the environment to form the training data set $D = \{(O_i, W_i)\}_{i=1}^{N}$, where O_i denotes the ith observation of the vehicle to the environment, and W_i denotes the ith ground-truth waypoint. The goal is to fit a function F with θ parameters through a neural network such that the output of F is as similar as possible to that of W. The DNN represents the mapping function F_θ. The learning procedure aims to continuously optimize θ so that the loss value is minimized.

$$Minimize_\theta \sum_i \mathcal{L}(F(O_i;\theta), W_i) \tag{1}$$

where W_i is the ith ground-truth waypoint, F is the waypoint prediction network with the θ parameter, and O_i denotes the ith observation of the vehicle to the environment. L is the loss function for the prediction of waypoints and the ground-truth waypoints.

Each episode has an end position, marked as D_g, while D_g determines the high-level navigation commands at the intersection $D_g = \{C_i, \ldots, C_m\}$ where C_i denotes the ith high-level navigation command in one episode at the intersection, which eliminates the ambiguity of the intersection and has been demonstrated in Codevilla et al. (2018) and subsequent studies to be important. Thus, we also use C_i as a part of the training sample. The final input is represented as $O_i = \{(I, C, V)\}_i$, where I denotes the image input, $C \in \{Left, Right, Straight\}$ denotes the high-level navigation commands, and V denotes the vehicle speed. The final optimization function is derived from Formula 1.

$$\theta^* = argmin_\theta \sum_i \mathcal{L}(F((I, C, V)_i;\theta), W_i) \tag{2}$$

where F is the waypoint prediction network with θ parameter, and I, C, V are the inputs. I represents the image from the camera, C represents the high-level navigation commands, and V is the vehicle speed. W_i is the ith ground-truth waypoint. θ^* represents the final neural network parameters to be learned.

The network outputs the predicted waypoints which are then translated by the PID controller into the low-level control commands for steering, throttling, and braking, expressed as follows:

$$U_i = PID(F(O_i;\theta^*)) \tag{3}$$

where F presents the network with θ parameters and predicts the waypoints using O as inputs. O_i is the ith observation of the environment, specifically containing the camera, vehicle speed, and high-level navigation commands. PID is the PID controller function, which does the mapping of waypoints to low-level commands. Our model uses two PID controllers: lateral PID and longitudinal PID. U_i is the ith specific control command to manipulate the vehicle, $U_i \in \{Steer, Throttle, Brake\}$.

3.2. Perception and plan module

The perception and plan module used by the model specifically refer to the dual-flow network, as shown in Figure 3. T denotes the sampling rate of images per second, and α represents the ratio of the image sampling rate of the perception

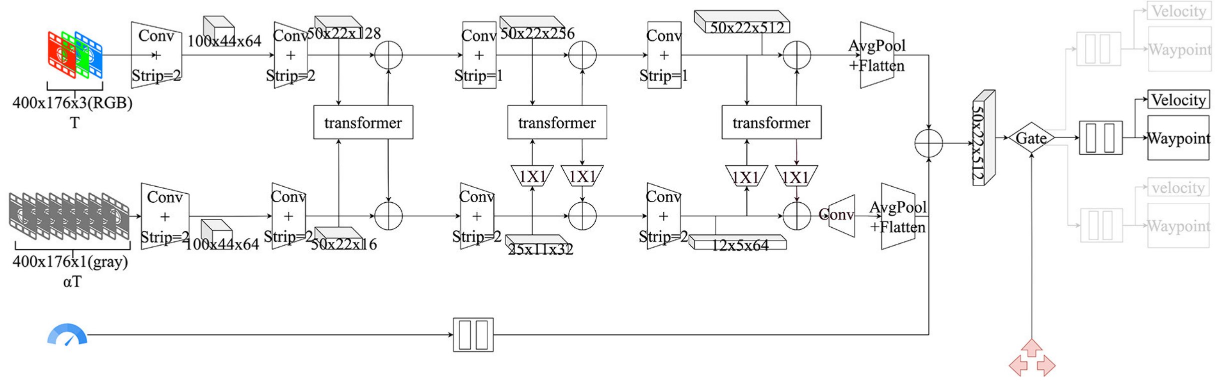

FIGURE 3
Architecture of dual-flow network. The first row is the perception network using an RGB image as input, the second row is the motion network using a sequence gray image as input, and the third row presents the input of the velocity. T denotes the sampling rate of images per second, and α denotes the ratio of the image sampling rate of the perception network and motion network.

network and motion network. The dual-flow network is the core and innovative point that does the mapping from the original image and dashboard metrics to the waypoints. The design of the dual-flow network is inspired by the study of retinal neural cells (Weiss et al., 2002). Specifically, both the perception network and motion network use the video captured by the camera as input; however, the sampling rate is different. Finally, the features are fused using the attention mechanism.

3.2.1. Input and output

The input is divided into images, speed, and high-level navigation commands. The dual-flow network is fed with the video captured by the vehicle's front camera, but at different sampling rates. The perception network takes as the input an RGB image with the T sampling rate, labeled as $I_{rgb} := T$, and the motion network takes as the input a gray image with the αT sampling rate, denoted as I_{gray}. The ratio of input is represented as

$$I_{rgb} = \alpha I_{gray} \tag{4}$$

where α represents the ratio of the image sampling rate of the perception network and motion network; in our model, $T = 3, \alpha = 8$, that is, there are three images per second for the perception network input, corresponding to 8×3 images for the motion network. The speed input is based on the vehicle dashboard speed and is normalized by $V = \frac{V_{current}}{V_{max}}$, where $V_{current}$ represents the current speed of the vehicle, and V_{max} is the maximum speed limited by the model. When the value of $V = 0$, it means stationary, and when the value is 1, it means the maximum speed. The high-level navigation command is denoted by the symbol $C_i \in \{Left, Right, Straight\}$.

The output represents the predicted waypoints. These waypoints are used as inputs to the PID controller and are converted to the vehicle's low-level control commands.

3.2.2. Network architecture

The dual-flow network is not limited to a specific backbone network architecture to be used. Considering the ease of deployment and the broadness, we adopt the RestNet-34 backbone network and make important improvements to it, as shown in the following Table 1.

The perception network's strip from stage 3 in the residual module is set to 1, so the resolution is not reduced, and the final output is a $P_o = (50 \times 22 \times 512)$ feature vector. Intuitively, high-resolution features help recognize image details.

The motion network considers the motion of the object as a whole and does not require pixel-level differentiation. Compared with the perception network, the motion network has a number of channels that is reduced by a factor of 8, and the resolution is reduced by a factor of 32, resulting in a final output $M_o = (12 \times 5 \times 32)$. The motion network helps to provide sensitivity to motion and improves the detection speed, in agreement with Weiss et al. (2002)'s conclusion.

Attention (Vaswani et al., 2017) is the key to fusing the dual-flow network features. Attention completes the learning of different feature weights, which simply means that higher weights are given to the important features. Suppose we have the two features $F1 \in R^{n \times d}$ and $F2 \in R^{m \times d}$, where $\mathbf{n}, \mathbf{m}$, and $\mathbf{d}$ represents a different number of dimensions. $F1$ and $F2$ perform attention such that both vectors have the same dimension $\mathbf{d}$. For instance, Q is the vector corresponding to $F1$, and K and V are the vectors corresponding to $F2$. We calculate the attention

TABLE 1 Details of the structure of the dual-flow network framework.

Stage	Perception net	Motion net	Output size $H \times W \times C$
Input	-	-	$PNet: 400 \times 176 \times 3, MNet: 400 \times 176 \times 8$
Conv1	$7 \times 7, 64, s = 2$	$7 \times 7, 64, s = 2$	$PNet: 200 \times 88 \times 64, MNet: 200 \times 88 \times 64$
Pool1	$3 \times 3, maxpool, s = 2$	$3 \times 3, maxpool, s = 2$	$PNet: 100 \times 44 \times 64, MNet: 100 \times 44 \times 64$
Stage2	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 3 \times 3, \mathbf{8} \\ 3 \times 3, \mathbf{8} \end{bmatrix} \times 3$	$PNet: 100 \times 44 \times 64, MNet: 100 \times 44 \times 8$
Stage3	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4, \mathbf{s=1}$	$\begin{bmatrix} 3 \times 3, \mathbf{16} \\ 3 \times 3, \mathbf{16} \end{bmatrix} \times 4$	$PNet: 50 \times 22 \times 128, MNet: 50 \times 22 \times 16$
Stage4	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6, \mathbf{s=1}$	$\begin{bmatrix} 3 \times 3, \mathbf{32} \\ 3 \times 3, \mathbf{32} \end{bmatrix} \times 6$	$PNet: 50 \times 22 \times 256, MNet: 25 \times 11 \times 32$
Stage5	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3, \mathbf{s=1}$	$\begin{bmatrix} 3 \times 3, \mathbf{64} \\ 3 \times 3, \mathbf{64} \end{bmatrix} \times 3$	$PNet: 50 \times 22 \times 512, MNet: 12 \times 5 \times 64$
Deconv	-	$3 \times 3, s = 4, p = 1, outpadding = 5, c = 512$	$PNet: 50 \times 22 \times 512, MNet: 50 \times 22 \times 512$
Avg + sum	1×1	1×1	$1 \times 1 \times 512$

The bold digit indicates network improvement points, where red and green values correspond to the number of channel and the stripe size, respectively. The perception network maintains the resolution from stage 3. The motion network reduces residual channels to one-eighth of their original size.

formula for F1 attention to F2 as follows:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{5}$$

where $Q \rightarrow F1 \in R^{n\times d}$, $K, V \rightarrow F2 \in R^{m\times d}$. Finally, the learned attention values are transformed by multilayer perceptron (MLP) and added to F1 to obtain the final feature output:

$$F1' = MLP(Attention) + F1 \tag{6}$$

where $F1^{'}$ is the new feature after the attention calculation.

To better learn more features, our model also employs a multi-head attention mechanism. Multi-head attention is expressed as

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{7}$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V). \tag{8}$$

We perform feature fusion in stages 3–5 using the multi-head mechanism, where $Layer = 4, H = 4$, and $D_{token} = 512$ correspond to the number of attention layers, the number of attention heads, and the dimensionality of features, respectively. Note that the outputs of the perception network and motion network have different feature dimensions. To unify the feature dimensions, we adjust the features of the motion network by 1×1 convolution after doing the attention calculation, and then we adjust the original number of channels. Let us take stage 3 of feature fusion as an example. The progress is expressed as the formula $MF^{'} = Conv(Attention(Conv(MF, 512), PF), 16)$, where $PF, MF, andMF'$ represent the feature of the perception network, the feature of the motion network, and the new feature after doing attention calculation, respectively. The attention calculation of the motion network also adds the position of coding information, which is crucial for motion detection because different sequences of pictures represent different movements.

Finally, the features M_o of the motion network perform the deconvolution operation $M_o^{'} = DeConv(M_o)$, which is then summed with the output P_o of the perception network, outputting a 512-dimensional vector of the dual-flow network $Net_o = Flatten(AvgPool(M_o^{'}) + AvgPool(P_o))$. Also referring to Codevilla et al. (2019), we map the velocity of the vehicle by the MLP into a 512-dimensional vector and add it to the output Net_o of the network. Then, the output vector dimension remains the same, at 512 dimensions.

Waypoint regression uses the feature vector Net_o of the dual-flow network as input to perform the prediction of waypoints and velocities by an MLP containing two hidden cells $< 512, 64 >$, labeled as $Head$. As the navigation command C forms three mutually exclusive values, it also forms three mutually exclusive branches $Head \in \{Left, Right, Straight\}$.

3.2.3. Loss function

The model uses the $L1$ loss function MAE to regress speed and waypoints. W_i and W_i^* denote the ground-truth waypoint and the predicted waypoint, respectively. V_i and V_i^* denote the ground-truth speed and the predicted speed, respectively. The loss function is denoted as

$$\mathcal{L} = \sum_{i=1} \lambda |W_i^* - \mathcal{W}_i| + (1 - \lambda)|V_i^* - V_i| \tag{9}$$

where λ denotes the balance coefficient of velocity and waypoints, and we choose $\lambda = 0.8$, which increases the focus on forecasting waypoints.

3.3. PID controller

The proportional-integral-derivative controller fulfills the mapping of waypoints to the low-level controller of the vehicle. We use a longitudinal controller and a lateral controller, where the longitudinal controller outputs the throttle command, and the lateral controller completes the steering command.

The longitudinal controller fits the gap between the average speed of the waypoint and the current speed of the vehicle. The waypoints predicted by the model are denoted as $W^* = \{W_1^*, W_2^*, \ldots, W_k^*\}$, and the average speed of the waypoint is

$$V_t^* = \frac{1}{K} \sum_{k=1}^{K} \frac{\|W_i^* - W_{i-1}^*\|}{\delta t} \tag{10}$$

where δt represents the time interval between waypoints, and $W_0 = 0$. The goal of the controller is to minimize the difference between the vehicle's current speed with V_t^*. When the value of the subtraction of the two is negative or the speed is less than the threshold value ϵ, it means braking. In our model, $K = 4$ and $\epsilon = 20km/h$.

The lateral controller fits the angle of the waypoint to the angle of the vehicle, as shown in Figure 4. We first fit an arc to four waypoints and steer toward a red point P^* on the arc. The goal of the lateral controller is to minimize the angle of the head of the vehicle to P^*. The final result is

$$A^* = tan_{-1} \frac{P_y}{P_x} \tag{11}$$

where A^* is the target steering angle.

4. Experiments

This chapter details the establishment of the experiments, the comparison of the results, and the related ablation experiments.

4.1. Experiment settings

4.1.1. Data collection

Data collection was done in eight cities through the CARLA simulator, version 0.9.12, *via* autopilot mode. City $[1-4, 6-8]$ was used for training, and city 5 was used for testing. For each city, 10 h of data collection were obtained, with 100 random pedestrians and 70 vehicles. Each episode consisted of start and end coordinates (GPS), and a vehicle at the destination of the limited time without collision means tasks success; otherwise, it means failure. However, if the vehicle infringes on the traffic light rules and does not cause a collision, it also means success. During the data collection, we also added steering noise to enhance the generalization of the data. The collected data contained the RGB video data of 20 HZ from the camera on the front of the vehicle and sensor data including speed, high-level navigation commands, waypoints, vehicle position, head angle, throttle, brake, and acceleration. We removed the effect of weather conditions and used only clear noon weather, considering that the goal of the experiment was to test the performance of the vehicle in complex environments and dynamic objects. The evaluation of the model was performed in city 5 with a set of 10 sub-tasks. Each task contained a starting point, a destination point, high-level navigation commands for the task, a travel length of 1,000–2,000 m, and the yielding of 100 pedestrians and 70 vehicles at predefined locations. The evaluation metrics included route completion (RC), count of collisions (Collision), and time-outs.

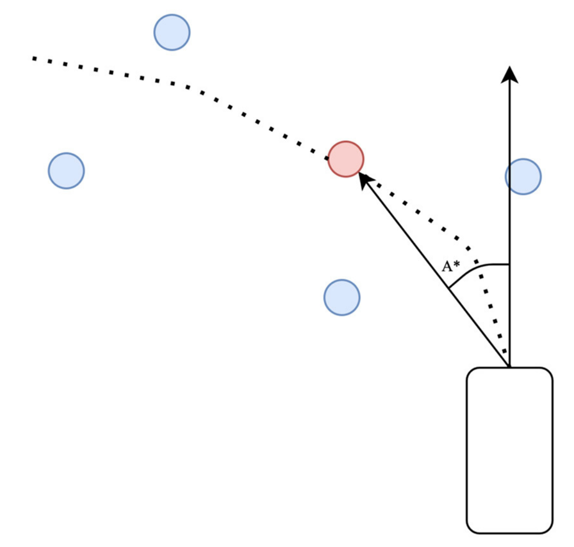

FIGURE 4
Lateral PID controller. The red waypoint is obtained by fitting a curve with four blue predicted waypoints. A^* is the target steering angle.

4.1.2. Data augmentation

Previous studies Laskey et al. (2017) and Codevilla et al. (2019) have shown that data augmentation is crucial for IL. Therefore, data augmentation was applied to our model at the time of data collection and model training, respectively. During data collection, we injected noise (steer, throttle, and brake) and then returned to normal through the vehicle *via* autopilot. For model training, we cropped the input images with the vehicle as the bottom center, then performed a uniform rotation angle of $[-10, 10]$, and finally performed a random uniform pixel shift $[-5, 5]$ pixels. We also used conventional image enhancement techniques, including Gaussian blur, Gaussian noise, salt-and-pepper noise, and region dropout.

4.1.3. Model comparison

Only models that use images as input were selected for comparison. The CILRS (Codevilla et al., 2019) model uses images from the face-on camera as input, and the ResNet34 backbone network is used for feature extraction, which predicts the vehicle's low-level commands. For comparison with our model, the prediction head of the CILRS model was replaced, labeled as CILRS-W, which can also be seen as using only our perception network for prediction. LBC (Chen et al., 2020) takes images as input and uses a knowledge distillation approach to optimize the "student" network from the "teacher" network. In the latest version, an image heat map is applied to improve its performance. GRIAD (Chekroun et al., 2021) is a video-based autonomous driving algorithm that uses deep reinforcement learning. TCP (Wu et al., 2022) explores the combination of trajectory planning and direct control and ranks second in the CARLA Autonomous Driving Leaderboard. We reproduced it according to the latest code.

4.1.4. Training details

The model uses four multi-head attention and four attention heads for each feature layer, and the input of each attention is the same as that of the perception network feature dimension. Note that the initial value of the input channel of the motion network is equal to the average value of the AlexNet channels. All network models use image enhancement techniques. The ResNet networks were all initialized using AlexNet (Krizhevsky et al., 2012) weights, with an initial learning rate of 0.0001 using the Adam optimizer, and we divided the learning rate by 10 when we found that the network error was not decreasing. Finally, we trained the models with 4 RTX 3090 GPUs for 10 h and a batch size of 12 because of the high resolution of the image of 400×176 pixels.

4.2. Comparison with model results and analysis

The training and testing results of the model are shown in Table 2. First, our model has the highest success rate. The test task contains numerous pedestrians and vehicles, so LBC cannot detect dynamic objects effectively with the lowest success rate. CILRS-W uses the prediction of waypoints and speed, which improves the model's performance, but it performs poorly in long-distance tasks and when facing the sudden intrusion of dense pedestrian and vehicle traffic. Our model can be seen as integrating the advantages of both the CILRS-W model and the motion prediction network. In the test, the success rate improved by 5%, and the collision rate decreased by 5% compared to CILRS-W. GRID achieves a success rate comparable to that of our model, which uses a deep reinforcement learning approach, while our model uses a supervised learning approach, and the two types of approaches are not directly comparable. TCP is a powerful model that works best by combining the use of trajectory planning and direct control. However, our model explores the extraction of space–time features, and the focus of our study is different from that of the TCP model. Second, the model of migration is better. Compared with CILRS-W, the transfer of the success rate is improved by 2%, and the collision rate is reduced by 3% (19 and 17% for CILRS-W and 17 and 14% for our model). Finally, our model is relatively cautious compared to CILRS-W. The LBC network does not predict vehicle speed, so the time-out is high relative to the task, and the model has a greater tendency to predict speed to 0. CILRS-W uses speed prediction as an aid to reduce time-out cases. However, our model can be seen as CILRS-W overlapping with the motion network branch so that the model is relatively cautious, and the time-out situation is 1% higher compared to CILRS-W.

We further analyzed the collisions in the test, as shown in Figure 5. The LBC model is the least effective. Compared to the CILRS-W model, our model decreases the pedestrian collision rate, the vehicle collision rate, and the traffic light collision by 3, 1, and 1%, respectively. This is because the perception network maintains high-resolution features that help in the recognition of pedestrians, vehicles, and traffic lights; moreover, the motion network estimates the motions of pedestrians and vehicles, which complement each other.

4.3. Ablation studies

The core designs are mainly for dual-flow networks and features fusion using an attention mechanism. The driving performance of the dual-flow network is demonstrated in Table 2. To further verify the validity of the model components, we conducted a series of ablation experiments.

4.3.1. Is the attention mechanism valid?

The attention mechanism of the network was removed, and the features of the two networks were summed backward and fused, labeled as ours (w/o A). The results are shown in Table 2. The (w/o A) model decreased the success rate by 3%

TABLE 2 Model performance comparisons.

Method	Train			Test		
-	RC↑	Col ↓	Time-out↓	RC↑	Col ↓	Time-out↓
LBC	56	36	8	37	63	16
CILRS-W	88	9	3	69	26	5
Ours(w/o A)	90	7	3	71	25	4
Ours	91	7	2	74	21	5
GRIAD	-	-	-	75	20	5
TCP	95	3	2	78	18	4

The table reports the percentage of each metric, and the arrow indicates the direction of the metric better. RC denotes the success rate, and IC is the number of collisions. Our model is safer and has a higher completion rate, although the time-out metric is slightly higher.

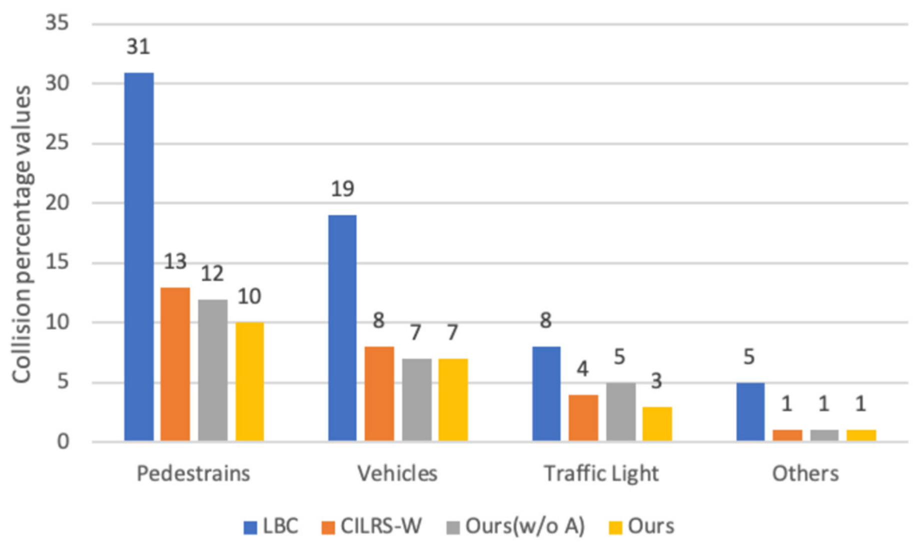

FIGURE 5
Collision chart. The graph shows the percentage of collisions of each model with pedestrians, vehicles, traffic lights, and others. Our models as shown in yellow have the lowest collision rate.

TABLE 3 Attention layer test.

Attention layer	Train RC	Test RC
1	87	72
4*	90	75
6	91	75
8	94	73

Train RC and test RC denote the percentage of completed routes on the training and test sets, respectively. Considering the real-time and completion rate, we used four attention layers, marked by *.

TABLE 4 Impact of sampling rate on the dual-flow network.

α	Train RC	Test RC
2	86	70
8*	90	73
16	83	69

α refers to the ratio of the sampling rates of perception network and motion network inputs. Train RC and Test RC denote the percentage of completed routes on the training and test sets, respectively. α marked by * is equal to 8, which is the best result and the final choice for our model.

and increased the collision rate by 4% compared to our model, proving the effectiveness of the attention mechanism. However, too high a level of feature fusion is not beneficial to the alignment of network features.

4.3.2. Are multiple attention layers needed?

The model used the same number of attention layers for each layer of features, initialized each time using Alex weights, trained for 24 h on the training data set, and then validated on the test set. The results are shown in Table 3. It can be seen that the

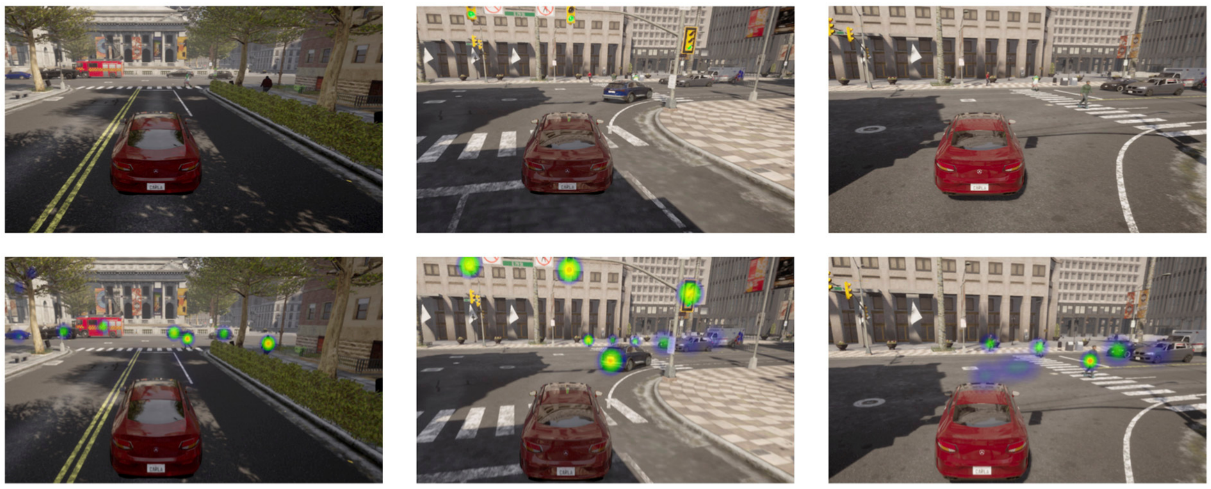

FIGURE 6
Visual attention. The first row is the original image. The second row is the attention image. The dual-flow network focuses on areas near vehicles, pedestrians, and traffic lights.

TABLE 5 Run time.

Method	Late fusion (LF)	Geometric fusion (GF)	TransFuser (TF)	Dual-flow (Ours)
MS/Frame	23.5	43.5	27.6	35.3

We obtained the time taken by each model to process a single frame using the total time taken by the model to finish a route divided by the total number of frames. Our model takes 35.3 ms per frame and is relatively time-consuming.

training error of the model tends to decrease as the number of layers increases, and the reason for this trend is related to the increase in the parameters of the model, resulting in a better fit to the training dataset. In the test dataset, the performance of the other models is comparable, except for the first model, whose performance is considered poor. However, the poor migration effect of the model using eight attention layers may be related to the overfitting of the data. In terms of the overall consideration, using four attention layers is the best choice. Note that using a different number of layers for each layer produces better results.

4.3.3. Is the ratio of the image sampling rate, marked as α of the dual-flow network, reasonable?

For the purpose of verifying the effect of α (in Equation 4), we used the model with four attention layers for scratch training, and the results are shown in Table 4. α takes too small a value, the motion network cannot learn the motion features, and the whole model degenerates into the CILR-W model. Moreover, α takes too large a value, resulting in too long a period, which is not conducive to the alignment of dual-flow network features and reduces the performance of the model.

4.3.4. Attention visualizations

The attention mechanism of the network was visualized, as shown in Figure 6. The network has a high weight for moving pedestrians, cars, and signals, which provides help for dynamic object detection in complex environments.

4.4. Run time

We measured the running time of our model on a single RTX 3090 GPU by averaging over all time steps of the evaluation route, as shown in Table 5. Some of data in Table 5 are quoted from work (Chitta et al., 2022). Our model takes 35.3 ms per frame, an increase of 11.8 ms and 7.7 ms compared to LF(23.5 ms) and TF(27.6 ms), respectively. Mining the temporal features of the video naturally consumes more time, following our estimation.

4.5. Limitations

First, our best model can learn by imitating the driving habits of an expert at the same level as the expert, which is the ceiling of the model. The expert data bring bias in the data distribution, which is a drawback of our model. Second, the real-time performance of our model needs to be further optimized, as analyzed in other work Section 4.4.

5. Conclusion

Motivated by the asymmetry of spatial and temporal sensitivity in the studies of the retinal cell, we proposed a dual-flow network to learn the space–time features of videos for autonomous driving. In the model, the perception network uses a modified ResNet34 backbone to maintain high image resolution and achieve a refined understanding of the environment, while the motion network uses a reduced channel ResNet34 backbone (the channels are reduced to $\frac{1}{8}$), which improves the computational speed and completes the learning of motion information. Finally, the feature layers of two networks are fused using an attention mechanism. Through experiments, we demonstrated that the network structure we designed aided in the detection of dynamic objects in complex environments, achieving a completion rate of 74%.

In the future, we will try to use deep reinforcement learning to relieve the distribution mismatch caused by IL and to increase the adaptability to unknown scenarios. Also, the real-time performance of our model needs to be further optimized. Inspired by recent research (Shang et al., 2022; Yuan et al., 2022a,b), autonomous driving perception algorithms can also be used to extract low-dimensional information relevant to decision-making from high-dimensional information. The application of multi-sensor data, especially video data (high latitude data) and LiDAR point cloud data (sparse data), is a new direction worthy of research. Hopefully, our research will promote the use of space–time features in autonomous driving.

Data availability statement

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

Author contributions

LY, WL, WZ, and TY contributed to conception and design of the study. TY organized the database. LY, WL, and WZ performed the statistical analysis. LY wrote the manuscript. All authors contributed to manuscript revision, read, and approved the submitted version.

Funding

This work was supported by the Fundamental Research Funds for the Central Universities of China (No. N2216010) and the National Key Research and Development Program of China (No. 2018YFB1702000).

Conflict of interest

LY and TY were employed by Alibaba Group.

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

Bertasius, G., Wang, H., and Torresani, L. (2021). Is space-time attention all you need for video understanding. *arXiv preprint* arXiv:2102.05095 2, 4. doi: 10.48550/arXiv.2102.05095

Camus, T. (1995). *Calculating time-to-contact using real-time quantized optical flow*. Technical Report No. 1. Tubingen, Germany.

Capito, L., Ozguner, U., and Redmill, K. (2020). "Optical flow based visual potential field for autonomous driving," in *2020 IEEE Intelligent Vehicles Symposium (IV)* (IEEE), 885–891.

Carreira, J., and Zisserman, A. (2017). "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Honolulu, HI: IEEE), 6299–6308.

Chekroun, R., Toromanoff, M., Hornauer, S., and Moutarde, F. (2021). "Griad: General reinforced imitation for autonomous driving. method ranked# 1 and# 4 in carla challenge 2021," in *NeurIPS 2021 Workshop on Machine Learning for Autonomous Driving*.

Chen, C., Seff, A., Kornhauser, A., and Xiao, J. (2015). "Deepdriving: learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision* (Santiago: IEEE), 2722–2730.

Chen, D., Zhou, B., Koltun, V., and Krähenbühl, P. (2020). "Learning by cheating," in *Conference on Robot Learning* (PMLR), 66–75.

Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., and Geiger, A. (2022). Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *arXiv e-prints* arXiv-2205. doi: 10.1109/TPAMI.2022.3200245

Codevilla, F., Müller, M., López, A., Koltun, V., and Dosovitskiy, A. (2018). "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)* (Brisbane, QLD: IEEE), 4693–4700.

Codevilla, F., Santana, E., López, A. M., and Gaidon, A. (2019). "Exploring the limitations of behavior cloning for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (Seoul: IEEE), 9329–9338.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., et al. (2020). An image is worth 16x16 words: transformers for image recognition at scale. *arXiv preprint* arXiv:2010.11929. doi: 10.48550/arXiv.2010.11929

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). "Carla: an open urban driving simulator," in *Conference on Robot Learning* (PMLR), 1–16.

Feichtenhofer, C., Fan, H., Malik, J., and He, K. (2019). "Slowfast networks for video recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (Seoul: IEEE), 6202–6211.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 1627–1645. doi: 10.1109/TPAMI.2009.167

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: the kitti dataset. *Int. J. Rob. Res.* 32, 1231–1237. doi: 10.1177/0278364913491297

Gern, A., Moebus, R., and Franke, U. (2002). "Vision-based lane recognition under adverse weather conditions using optical flow," in *Intelligent Vehicle Symposium, 2002, Vol. 2* (Versailles: IEEE), 652–657.

Gibson, J. J. (2014). *The Ecological Approach to Visual Perception: Classic Edition*. Psychology Press.

Girdhar, R., Carreira, J., Doersch, C., and Zisserman, A. (2019). "Video action transformer network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (Long Beach, CA: IEEE), 244–253.

Hussein, A., Gaber, M. M., Elyan, E., and Jayne, C. (2017). Imitation learning: a survey of learning methods. *ACM Comput. Surveys* 50, 1–35. doi: 10.1145/3054912

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. *arXiv preprint* arXiv:1602.07360. doi: 10.48550/arXiv.1602.07360

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Honolulu, HI: IEEE), 2462–2470.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Columbus, OH: IEEE), 1725–1732.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*.

Laskey, M., Dragan, A., Lee, J., Goldberg, K., and Fox, R. (2017). "Dart: optimizing noise injection in imitation learning," in *Conference on Robot Learning (CoRL), Vol. 2*, 12.

Lenz, P., Ziegler, J., Geiger, A., and Roser, M. (2011). "Sparse scene flow segmentation for moving object detection in urban environments," in *2011 IEEE Intelligent Vehicles Symposium (IV)* (Baden-Bade: IEEE), 926–932.

Li, Z., Motoyoshi, T., Sasaki, K., Ogata, T., and Sugano, S. (2018). Rethinking self-driving: multi-task knowledge for better generalization and accident explanation ability. *arXiv preprint* arXiv:1809.11100. doi: 10.48550/arXiv.1809.11100

Liang, X., Wang, T., Yang, L., and Xing, E. (2018). "CIRL: controllable imitative reinforcement learning for vision-based self-driving," in *Proceedings of the European Conference on Computer Vision (ECCV)* (Cham: Springer), 584–599.

Lieb, D., Lookingbill, A., and Thrun, S. (2005). Adaptive road following using self-supervised learning and reverse optical flow. *Rob. Sci. Syst.* 1, 273–280. doi: 10.15607/RSS.2005.I.036

Lin, G., Milan, A., Shen, C., and Reid, I. (2017). "Refinenet: multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Honolulu, HI: IEEE), 1925–1934.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., et al. (2021). "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (Montreal, QC: IEEE), 10012–10022.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. (2022). A convnet for the 2020s. *arXiv preprint* arXiv:2201.03545. doi: 10.1109/CVPR52688.2022.01167

Lucas, B. D., and Kanade, T.. (1981). "An iterative image registration technique with an application to stereo vision," in *IJCAI'81: Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vol. 81* (Vancouver, BC).

Muller, U., Ben, J., Cosatto, E., Flepp, B., and Cun, Y. (2005). "Off-road obstacle avoidance through end-to-end learning," in *Advances in Neural Information Processing Systems 18*.

Okafuji, Y., Fukao, T., and Inou, H. (2015). Development of automatic steering system by modeling human behavior based on optical flow. *J. Rob. Mechatron.* 27, 136–145. doi: 10.20965/jrm.2015.p0136

Pomerleau, D. A. (1988). "Alvinn: an autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems 1*.

Prakash, A., Chitta, K., and Geiger, A. (2021). "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (Nashville, TN: IEEE), 7077–7087.

Rashed, H., Yogamani, S., El-Sallab, A., Krizek, P., and El-Helw, M. (2019). Optical flow augmented semantic segmentation networks for automated driving. *arXiv preprint* arXiv:1901.07355. doi: 10.5220/0007248301650172

Sauer, A., Savinov, N., and Geiger, A. (2018). "Conditional affordance learning for driving in urban environments," in *Conference on Robot Learning* (PMLR), 237–252.

Shang, M., Yuan, Y., Luo, X., and Zhou, M. (2022). An α-β-divergence-generalized recommender for highly accurate predictions of missing user preferences. *IEEE Trans. Cybern.* 52, 8006–8018. doi: 10.1109/TCYB.2020.3026425

Simonyan, K., and Zisserman, A. (2014a). "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems 27*.

Simonyan, K., and Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint* arXiv:1409.1556. doi: 10.48550/arXiv.1409.1556

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision* (Santiago: IEEE), 4489–4497.

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*.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., et al. (2016). "Temporal segment networks: Towards good practices for deep action recognition," in *European Conference on Computer Vision* (Springer), 20–36.

Wang, X., Girshick, R., Gupta, A., and He, K. (2018). "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT: IEEE), 7794–7803.

Weiss, Y., Simoncelli, E. P., and Adelson, E. H. (2002). Motion illusions as optimal percepts. *Nat. Neurosci.* 5, 598–604. doi: 10.1038/nn0602-858

Wu, P., Jia, X., Chen, L., Yan, J., Li, H., and Qiao, Y. (2022). Trajectory-guided control prediction for end-to-end autonomous driving: a simple yet strong baseline. *arXiv preprint* arXiv:2206.08129. doi: 10.48550/arXiv.2206.08129

Xiao, Y., Codevilla, F., Gurram, A., Urfalioglu, O., and López, A. M. (2020). "Multimodal end-to-end autonomous driving," in IEEE *Transactions on Intelligent Transportation Systems*.

Yuan, Y., He, Q., Luo, X., and Shang, M. (2022a). A multilayered-and-randomized latent factor model for high-dimensional and sparse matrices. *IEEE Trans. Big Data* 8, 784–794. doi: 10.1109/TBDATA.2020.2988778

Yuan, Y., Luo, X., Shang, M., and Wang, Z. (2022b). A kalman-filter-incorporated latent factor analysis model for temporally dynamic sparse data. *IEEE Trans. Cybern.* 1–14. doi: 10.1109/TCYB.2022.3185117 Available online at: https://ieeexplore.ieee.org/document/9839318