frontiers | Frontiers in Robotics and AI

TYPE Methods
PUBLISHED 15 November 2024
DOI 10.3389/frobt.2024.1468385

OPEN ACCESS

EDITED BY
Giovanni Iacca,
University of Trento, Italy

REVIEWED BY
Bin Zhi Li,
Chinese Academy of Sciences (CAS), China
Giovanni Boschetti,
University of Padua, Italy
Jing-Sin Liu,
Academia Sinica, Taiwan

*CORRESPONDENCE
Jiaxi Lu,
✉ lujx@g.ecc.u-tokyo.ac.jp

RECEIVED 11 September 2024
ACCEPTED 16 October 2024
PUBLISHED 15 November 2024

CITATION
Lu J, Takamido R, Wang Y and Ota J (2024) How to arrange the robotic environment? Leveraging experience in both motion planning and environment optimization. *Front. Robot. AI* 11:1468385. doi: 10.3389/frobt.2024.1468385

How to arrange the robotic environment? Leveraging experience in both motion planning and environment optimization

Jiaxi Lu[1]*, Ryota Takamido[2], Yusheng Wang[2] and Jun Ota[2]

[1]Department of Precision Engineering, School of Engineering, The University of Tokyo, Tokyo, Japan, [2]Research into Artifacts, Center for Engineering (RACE), School of Engineering, The University of Tokyo, Tokyo, Japan

This study presents an experience-based hierarchical-structure optimization algorithm to address the robotic system environment design problem, which combines motion planning and environment arrangement problems together. The motion planning problem, which could be defined as a multiple-degree-of-freedom (m-DOF) problem, together with the environment arrangement problem, which could be defined as a free DOF problem, is a high-dimensional optimization problem. Therefore, the hierarchical structure was established, with the higher layer solving the environment arrangement problem and lower layer solving the problem of motion planning. Previously planned trajectories and past results for this design problem were first constructed as datasets; however, they cannot be seen as optimal. Therefore, this study proposed an experience-reuse manner, which selected the most "useful" experience from the datasets and reused it to query new problems, optimize the results in the datasets, and provide better environment arrangement with shorter path lengths within the same time. Therefore, a hierarchical structural caseGA-ERTC algorithm was proposed. In the higher layer, a novel approach employing the case-injected genetic algorithm (GA) was implemented to tackle optimization challenges in robot environment design, leveraging experiential insights. Performance indices in the arrangement of the robot system's environment were determined by the robotic arm's motion and path length calculated using an experience-driven random tree (ERT) algorithm. Moreover, the effectiveness of the proposed method is illustrated with the 12.59% decrease in path lengths by solving different settings of hard problems and 5.05% decrease in easy problems compared with other state-of-the-art methods in three small robots.

KEYWORDS

industrial robotics, motion planning, environmental arrangement, takt time, optimization, hierarchical algorithm, experience reuse, intelligent manufacturing

1 Introduction

Robotic manipulation tasks such as welding (Zeng et al., 2019), transporting (Fu et al., 2020), assembling (Tereshchuk et al., 2019), and pick-and-place are commonly utilized in intelligent manufacturing industrial robots. In these tasks, takt time (Davies, 2009), the cycle time to perform the specific operation, is seen as an important evaluation criterion for robotic manipulation to assess the efficiency of a robotic system. The longer the takt time for a task, the less efficient the robotic system becomes. In our study, to simplify the problem, we consider the length of the robotic arm's trajectory as the moving time apart from the effect of the dynamics of the robot.

Although the primary focus of industrial robot motion planning lies in enhancing operational efficiency, it is affected by the configuration of various components within the robot environment. These components, including the base, conveyors, sensors, objects, and other robots, all play important roles during task execution, and the accuracy of the robot, which can be enhanced through calibration technology, also contributes to improving industrial automation (Li et al., 2021; Li et al., 2022a; Li et al., 2022b). Furthermore, alterations in the robotic system's environment necessitate corresponding adjustments in the robot's motion, even for repetitive tasks, resulting in notable variations in takt time or energy utilization (Gueta et al., 2007). Consequently, the spatial layout and arrangement of the robot environment significantly influence industrial productivity. To optimize the efficiency of a robotic system, it is imperative to establish an appropriate environment setup alongside devising efficient robot trajectories. Therefore, designing a robotic system that ensures reasonable takt time is imperative for enhancing the efficiency of industrial manufacturing.

However, most conventional studies focused on motion planning or path planning among those two aspects but ignored that with the improper design of the industrial line; the takt time could increase significantly. Hence, there are few studies that have developed the algorithm for identifying both efficient motion and environment arrangements.

Therefore, in this study, we proposed a new optimization method, caseGA-ERTC, for industrial robotic systems, which can facilitate both robot motion and environment arrangements' optimization. Specifically, to address the difficulty of the combined optimization problem of motion planning and environment arrangement, we utilized a hierarchical algorithm and experienced reuse manner for solving our problems. The former is to decompose the complicated problems into simpler problems to reduce the calculation cost, and the latter is to reuse the past solutions in similar optimization problems to improve the results of the past solutions. Therefore, we introduced two experience-based methods for both motion planning and environment arrangement part. Based on the experiments, our proposed method can achieve the placements of conveyors in which robots could work with shorter path lengths compared with non-experience-based state-of-the-art methods.

In our previous study (Lu et al., 2023), only one robot was tested, and limited scenarios and difficulties of problems were proposed. Therefore, in this study, we aim to broaden the definition of problems and utilize information from more complicated situations and more diverse kinds of robots (in Section 4.1) to accomplish the goal of adding to the preliminary results for the comprehensive study.

The following are the contributions made to this field of study:

1. We proposed a hierarchical structure to realize the target of optimizing both environment configuration and motion planning together by decreasing the dimension of the whole environment arrangement problem;
2. We proposed building experience databases for both optimization and motion planning. Motion planning experiences gathered from simulations can also be applied to real-world robotic experiments;
3. Our proposed experience-based method could find results with shorter path lengths compared with non-experience-based methods with the same amount of optimizing time.

The remaining sections of this paper are organized as follows: Section 2 introduces related work in environment arrangement and experience-based motion planning algorithms. Section 3 provides the detailed viewpoints of the proposed methods. Section 4 explains the constructions and methods of execution of experiments. To demonstrate the advantages of our method, we compared our methods with other state-of-the-art methods solving benchmark problems in Section 5. Finally, the conclusion is described in Section 6.

2 Related work

2.1 Environment arrangement optimization problem

Most of the previous research studies focus on the adaptive motion planning algorithms to adapt to diverse and complex environments (Bulut, 2023; Jahanshahi et al., 2019; Levant and Livne, 2011), but failures are inevitable in complex environments. However, if we think about it from another perspective, if the environment itself is good enough for the robot's motion, then it can also reduce the probability of failures and improve the efficiency of the robot's motion planning. Therefore, the problem of environment arrangement optimization, which aims to design a suitable environment for the robot's motions, is introduced.

In recent years, significant advancements have been achieved in the development of algorithms aimed at addressing various optimization challenges in environment arrangement. For instance, Liu et al. (2008) conducted research focusing on enhancing the efficiency and productivity of the grit-blasting process in manufacturing by employing simulation-based modeling within a robotic system.

Similarly, Gueta et al. (2007) tackled intricate optimization dilemmas related to base placement and motion planning in inspection tasks. Their approach involved employing the tabu search algorithm (Fiechter, 1994), which employs a tabu list to navigate away from local minima, to determine optimal base positions. Subsequently, basic inverse kinematics (IK) and velocity profiles were utilized in the motion planning phase to compute joint angles and angular velocities.

However, these previous studies faced challenges in addressing highly complex, high-dimensional robotic environment design problems and did not incorporate the concept of experience reuse.

2.2 Experience-based motion planning

The problem of motion planning in robotics has been studied for decades, and recently, learning from the past, as well as reusing and retrieving from the previous experiences, has emerged as a new trend when dealing with motion planning problems. Experience-based motion planning could reuse past path planning problem results and does not have to plan from scratch (PFS). There are mainly two targets of these experience-based methods compared with non-experience-based methods: 1) improving the quality of the planned path or 2) saving the planning time. In addition, there are mainly two approaches for demonstrating the reuse of experiences while coping with various targets: 1) reconstructing and depicting the environment variables in task-relevant areas and 2) directly reusing and exploiting previously planned motions.

There are many previous research studies to achieve the first target, reconstructing and depicting the environmental variables in task-relevant areas.

Experience graph (E-Graph) (Phillips et al., 2013), Lightning framework (Berenson et al., 2012), and its variant Thunder (Coleman et al., 2015) introduce a novel approach to motion planning by retrieving and repairing previous planned paths.

Even though the above algorithms will become more accurate and effective as the algorithm runs and the volume of database increases because of the characteristic of these algorithms to gather environmental information and reconstruct the environment, these algorithms are more useful in static environments. Therefore, if the target of the tasks or the type of robot changes, the possibility of motion planning failure increases.

To facilitate the reuse of past experiences in addressing new motion planning challenges, the integration of case-based reasoning (Abdelwahed et al., 2018) into sampling-based algorithms has been proposed. This involves storing previous experiences in a case base and selecting relevant instances for new queries based on similarity metrics.

Because of these traits, employing case-based reasoning in complex scenarios like those encountered in 6-DOF robot systems is extremely challenging.

In our research, we employed the experience-driven random tree (ERT) algorithm (Pairet et al., 2021), which directly leverages similar past experiences to enhance result quality and computation speed.

3 Methods

3.1 Problem definition

As shown in Figures 1, 2, the settings of sensors and gathering information from sensors were not considered. The relative positions of the picking object and the end-effector were fixed. Therefore, we defined the problem of environment arrangement as finding the positions of conveyors when the robot picks the object, moving from the picking C-Space $C(\boldsymbol{q}_{pick})$ to a placing C-Space $C(\boldsymbol{q}_{place})$, at which the path length of the planned trajectory is the minimum.

The optimization process in question considers the path length of the motion as the primary evaluation criterion. To be precise, our objective is to determine the positions of conveyors $\boldsymbol{S}$ within the robotic environment and the associated trajectory $\boldsymbol{\mathcal{T}}$ by the final timestep T.

Therefore, we conceptualize the task of arranging the environment as an optimization problem, where the path length $l = f(\boldsymbol{\mathcal{T}})$ generated by the motion planning algorithm serves as the pivotal evaluation criterion.

$$\begin{aligned} \underset{X_S,X_G \in \mathbb{SE}(3)}{\arg\min} \quad & l = f(\boldsymbol{\mathcal{T}}(X_S, X_G)), \\ \text{s.t.,} \quad & flag(\boldsymbol{\mathcal{T}}(X_S, X_G)) = \text{TRUE} \\ & \text{time}(X_S, X_G) \leq T, \end{aligned}$$

where the picking configuration $C(\boldsymbol{q}_{pick})$ calculated from the start Cartesian position $[x_1, y_1, z_1]$ and the orientation of the end-effector corresponding to the defined tasks. Placing configuration $C(\boldsymbol{q}_{place})$ corresponds to the goal Cartesian position $[x_2, y_2, z_2]$, and constraints are added as

$$\text{s.t.,} \quad x_1 = x_2, y_1 = -y_2, z_1 = z_2$$

to simplify the defined problem by decreasing the dimension of the environment parameters. Therefore, less computation time is required when designing the environment arrangements.

Therefore, this environment arrangement problem could be defined as a 3-dimension (x, y, z) plus a 6-dimension motion planning problem, which is a 9-dimensional optimization problem.

In addition, $flag(\boldsymbol{\mathcal{T}}(X_S, X_G))$ is a Boolean function indicating whether the robot can successfully plan the motion for the given configurations of the task. If the task is feasible, it returns true; if not, it returns false.

3.2 Overall structure of our proposed method

The proposed method is structured as follows: as mentioned above, this environment arrangement problem is a high-dimensional optimization problem; therefore, the decomposition of this optimization problem is critical to solving it. This study adopts the case-GA + ERTC to solve the environment optimization problem. As shown in Figure 3, case-GA (Louis and McDonnell, 2004) is introduced first. In each generation of this genetic algorithm, the relative positions of environment components are changed to construct the new environment for the robot pick-and-place tasks.

Then, the motion planning algorithm, ERTConnect algorithm (Pairet et al., 2021), was presented here for planning valid motion in this new environment. The path length of the generated motion is returned to the case-injected GA as the evaluation criterion for the optimization process. After running the algorithm for n generations and by the last timestep T, the final environment arrangements and its corresponding motion when the robot executes the pick-and-place task in this environment would be output as the final results.

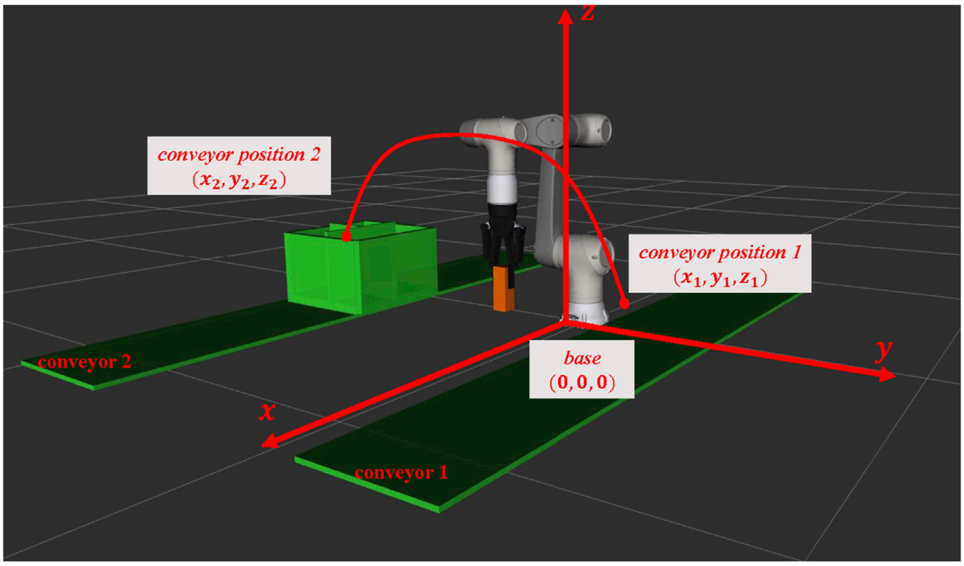


FIGURE 1
Environment settings of the problem to be solved in the 3D world coordinate. The environment is constructed with two conveyors, and one box is placed on one of those conveyors. Our task to be solved is defined as a pick-and-place task; the robot is catching one object from conveyor 1 to place it into the box on conveyor 2. However, the arrangements of the environment and the placement of the conveyors would influence the moving time for the robotic arm when executing the above tasks. To simplify the problem, we consider the length of the robotic arm's motion trajectory as the moving time of executing the pick-and-place task. Therefore, the target of this study is to arrange the environment arrangements in specific positions of conveyors for achieving the goal of optimizing the path length of the trajectory when executing the pick-and-place tasks.

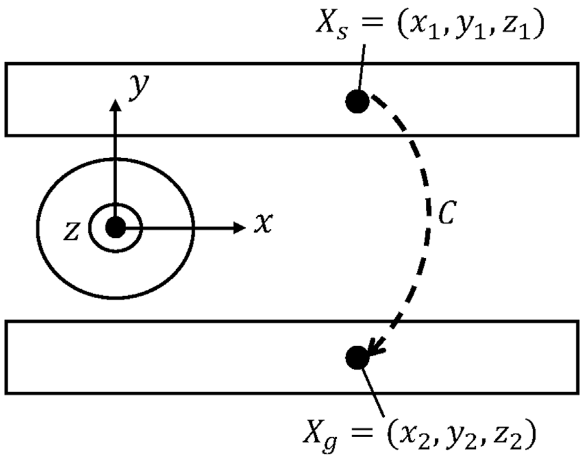


FIGURE 2
Two-dimensional environment setting of the problem to be solved (top view).

3.3 Construction of datasets

As shown in Figure 3, the problem is solved by the hierarchical construction of methods. The higher-layer optimization problem and lower-layer motion planning problem are solved. Two datasets are introduced for methods in each layer.

Casebase $\mathcal{C}$, used in higher layer: case-GA, is composed of solutions. Therefore, solutions from the previously solved environment arrangement problems $[x, y, z]$ were collected into the casebase $\mathcal{C}$.

In the lower layer, the motion planner's dataset $\mathcal{A}$ was constructed with trajectory information. As shown in Figure 5, because of the discontinuity of the sampling-based methods, trajectories are composed of time-series points in C-Space. When the robot's end-effector moves to the designated Cartesian pose X_G, each series point provides joint angle information.

Therefore, each trajectory is demonstrated with joint angles in C-Space $[\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n]$. The starting and ending configurations of each trajectory were saved for the comparison of similarity when selecting experiences.

3.4 Experience-based optimization method

Case-GA was used for the main structure of this optimization problem; therefore, the algorithm should start from the random generation of the first generation of the genetic algorithm.

For each positions of conveyors, which defines the construction of the environment, we decided for planning the motion of the robot for 10 times and chose the trajectory which has the smallest path length to mitigate the problem caused by the randomness of those sampling-based motion planning algorithms. As a result, for each individual, it will take approximately 4 s–40 s (if all failed). However, the total optimization time should be limited in real engineering industrial environment design problems.

Therefore, in this study, we set the total generation of this algorithm as 20. Here, we also defined the size of each population as 20; therefore, 20 $[x_0, y_0, z_0]$s are generated for constructing 20

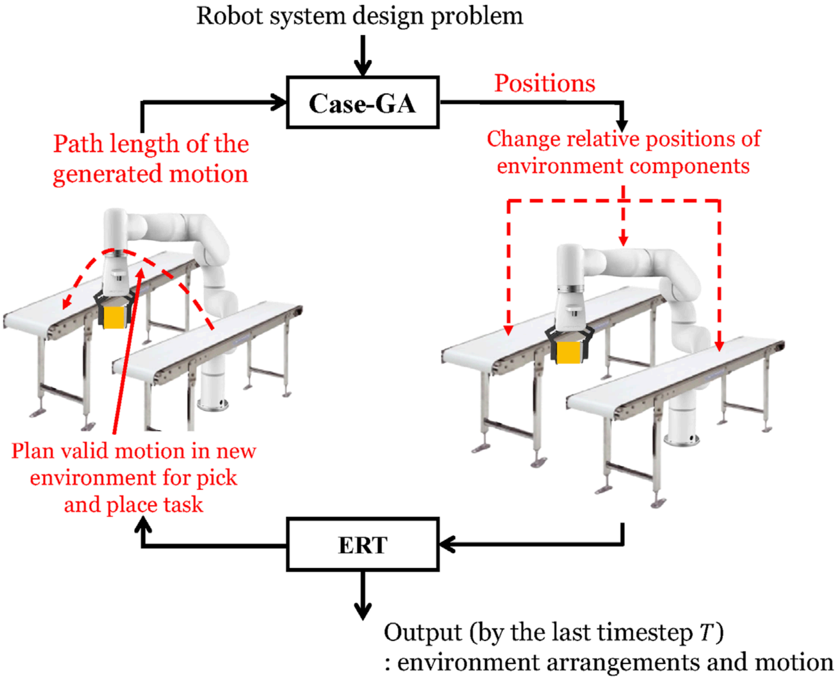

Robot system design problem
Case-GA
Positions
Change relative positions of environment components
Path length of the generated motion
Plan valid motion in new environment for pick and place task
ERT
Output (by the last timestep T)
: environment arrangements and motion

FIGURE 3
Overview of the methods.

different environments. Then, the motion planning algorithm is introduced for planning the pick-and-place task in 20 previously designated environments with 20 different pick-and-place positions.

The evaluation criterion revolves around the path lengths determined by the motion planning algorithm; therefore, two of the above $[x, y, z]$ with the minimum path lengths are defined as parents. The individual with the minimum path length would be saved and directly transferred into the next generation, and two parents mentioned above would be used for generating 19 individuals for the next generation by real-coded simulated binary crossover (SBX) and polynomial mutation methods (Pattanaik et al., 2018a). Then, the case-inject process of the genetic algorithm is considered (marked as purple in Figure 4). Case injection action is conducted over a specific period of generations; here, we defined it as 4. Since the total generation of this GA was designed as 20, case injection was conducted five times in each problem of scenery.

For each case-injection action, cases with the minimum difference with the best individual in the injecting population were selected; here, we define $[x_e, y_e, z_e]$ based on the following equation:

$$\boldsymbol{a} = \underset{\boldsymbol{a}_i \in \mathcal{C}}{\arg\min}\ \mathbf{dist}(\boldsymbol{a}_i - \boldsymbol{d}),$$

where $\boldsymbol{d}$ represents the best individual $[x_b, y_b, z_b]$ in the current injecting generation and $\boldsymbol{a}_i$ represents the $[x_i, y_i, z_i]$ solutions in casebase $\mathcal{C}$. When injecting cases into the population, there is a possibility that the selected case $[x_e, y_e, z_e]$ is already inside the population, especially not during the first injection. Therefore, only when the individual is not already inside the generation, the selected case would be injected. Then, the algorithm would be run for the next generation until meeting the stopping criterion. At the end of the algorithm, the final solution $[x_f, y_f, z_f]$ with the minimum path length and its corresponding motion would be returned.

3.5 Experience-based motion planning algorithm

Path length calculated by the motion planning algorithm in environments designated in the higher layer GA is seen as the evaluation criterion in the GA process. Therefore, experience-based planning was performed by the ERTConnect planner (Pairet et al., 2021) (marked as red in Figure 4).

As shown in Algorithm 1, positions of the conveyors $\boldsymbol{S}$ would be converted into start and goal configurations $\boldsymbol{q}_{start}$ and $\boldsymbol{q}_{goal}$ of the 6-DOF robot in C-Space with the IK solver since the ERTConnect motion planner searches the trajectory in configuration space.

In the selection of experiences from the database, a similarity equation is devised to assess the Euclidean distance between the configurations of the start and end states, as depicted below:

$$\xi_e = \underset{\xi_{e_i} \in \mathcal{A}}{\arg\min}\ \mathbf{dist}\left(\xi_{e_i}(start) - \boldsymbol{q}_{start}\right) + \mathbf{dist}\left(\xi_{e_i}(goal) - \boldsymbol{q}_{goal}\right), \quad (1)$$

where $\boldsymbol{q}_{start}$ and $\boldsymbol{q}_{goal}$ are the start and goal configurations of the current planning problem, respectively; meanwhile, ξ_{ei} is the experience path stored in library $\mathcal{A}$.

Then, the ERTConnect algorithm was executed for planning trajectories connecting starting and ending configurations using the previously selected experience ξ_e.

The fundamental operation of the ERT algorithm is similar to the Expansive Spaces Trees (EST) algorithm (Hsu et al., 1997) because of its foundation in tree-based sampling techniques.

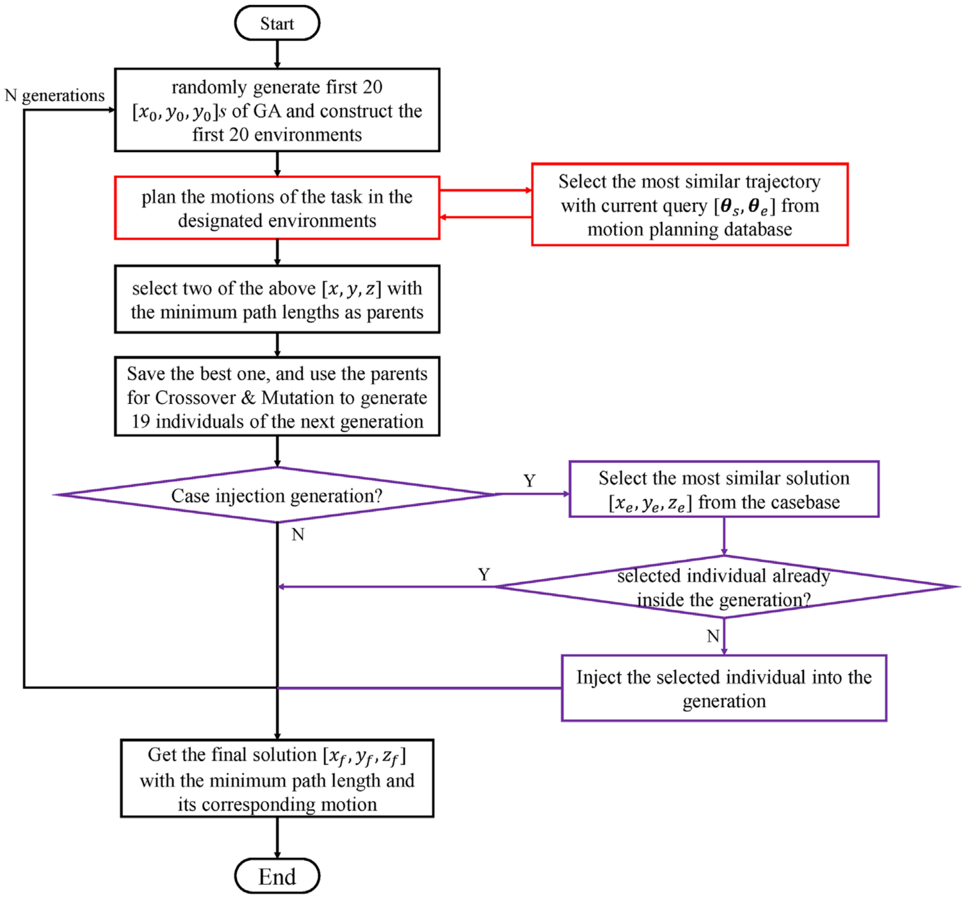

Start
N generations
randomly generate first 20 $[x_0, y_0, y_0]s$ of GA and construct the first 20 environments
plan the motions of the task in the designated environments
Select the most similar trajectory with current query $[\boldsymbol{\theta}_s, \boldsymbol{\theta}_e]$ from motion planning database
select two of the above $[x, y, z]$ with the minimum path lengths as parents
Save the best one, and use the parents for Crossover & Mutation to generate 19 individuals of the next generation
Case injection generation?
Y
Select the most similar solution $[x_e, y_e, z_e]$ from the casebase
N
Y
selected individual already inside the generation?
N
Inject the selected individual into the generation
Get the final solution $[x_f, y_f, z_f]$ with the minimum path length and its corresponding motion
End

FIGURE 4
Flowchart of the proposed experience-based environment arrangement method.

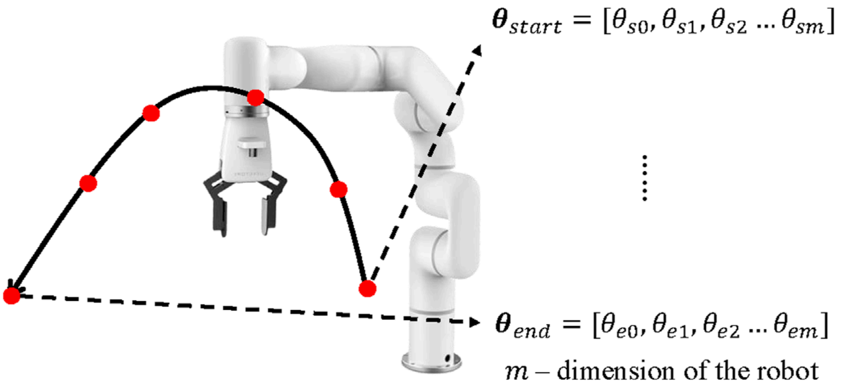

$\boldsymbol{\theta}_{start} = [\theta_{s0}, \theta_{s1}, \theta_{s2} \ldots \theta_{sm}]$
$\boldsymbol{\theta}_{end} = [\theta_{e0}, \theta_{e1}, \theta_{e2} \ldots \theta_{em}]$
m – dimension of the robot

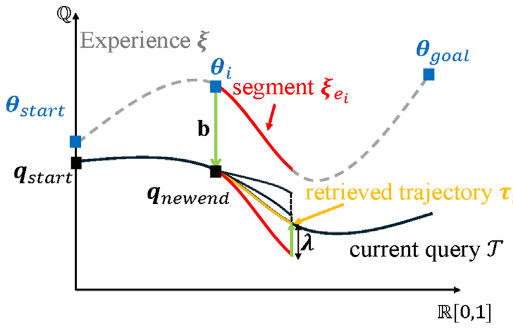
FIGURE 5
Trajectories stored in the ERTConnect algorithm dataset are time-series points in C-space.

However, it diverges with its integration of the experience-reuse component, which facilitates the establishment of spatial connectivity. Additionally, the ERTConnect algorithm represents a bidirectional iteration of the ERT algorithm.

During each query, the ERTConnect algorithm ascertains a seamless trajectory within the C-Space, spanning from the starting state to the target configuration, utilizing the given experience. As shown in Figure 6, the prior experience (composed of points in C-Space $\xi_e = [\boldsymbol{\theta}_0 \cdots \boldsymbol{\theta}_n]^T$) would be first randomly divided into several micro-experiences. These micro-experiences would be mapped to the current query piece by piece. For each micro-experience, the starting configuration $\boldsymbol{\theta}_i$ would be linked to the ending configuration

```
Input: Positions of conveyors $\boldsymbol{S}(x, y, \text{and } z)$ and Motion planner's database $\mathcal{A}$.
Output: Trajectory $\mathcal{T}$ and its corresponding path length $l$
  convert $\boldsymbol{S}$ to start and goal configurations $\boldsymbol{q}_{start}$, $\boldsymbol{q}_{goal}$ in C-space with the IK solver
  select experience $\boldsymbol{\xi}_e$ with the minimum Euclidean distance with the current query $\boldsymbol{q}_{start}$, $\boldsymbol{q}_{goal}$ from the database $\mathcal{A}$
  ERTConnect() with experience $\boldsymbol{\xi}_e$ plan for feasible solution $\mathcal{T}$
  calculate the path length $l$ of the trajectory $\mathcal{T}$
  end
  return $l$
```

Algorithm 1. ERTConnect motion planner ().



$\mathbb{Q}$
Experience ξ
$\boldsymbol{\theta}_{goal}$
$\boldsymbol{\theta}_i$
segment $\boldsymbol{\xi}_{e_i}$
$\boldsymbol{\theta}_{start}$
b
$\boldsymbol{q}_{start}$
$\boldsymbol{q}_{newend}$
retrieved trajectory $\boldsymbol{\tau}_i$
$\boldsymbol{\lambda}$
current query $\mathcal{T}$
$\mathbb{R}[0,1]$

FIGURE 6
Procedure for the ERTConnect algorithm [adapted from Pairet et al. (2021)]. The experience trajectory ξ_e is selected and set as a reference for planning a new trajectory. First, this experience trajectory ξ_e is randomly divided into several micro-experiences (e.g., the red segment ξ_{ei}). This experience segment will then be retrieved to a new query (in yellow) according to Equation 2 (in green). This process will be executed from the starting configuration $\boldsymbol{q}_{start}$ until the final configuration $\boldsymbol{q}_{goal}$ is realized.

$\boldsymbol{q}_{newend}$ of the planned trajectory using affine transformation:

$$\boldsymbol{\tau}_i = \boldsymbol{\lambda}_{m\times n} \cdot \boldsymbol{\xi}_{ei} + \boldsymbol{b}_{m\times 1}, \tag{2}$$

where $\boldsymbol{b}_{m\times 1} = \boldsymbol{q}_{newend} - \boldsymbol{\theta}_i$ could be seen as the shifting vector used to link the starting configuration of the micro-experience and new query. In addition, m represents the dimension of the robot. $\boldsymbol{\lambda}_{m\times n} = [0, \frac{\boldsymbol{\alpha}}{n-1}, \frac{2\boldsymbol{\alpha}}{n-1}, \ldots, \boldsymbol{\alpha}]$, in which $\boldsymbol{\alpha} = \frac{\boldsymbol{q}_{goal} - \boldsymbol{b}_{m\times 1}}{\boldsymbol{\theta}_{goal}}$. Therefore, the ending configuration of each micro-experience could be generated.

The path would be planned until the start and goal configurations are connected in the configuration space. In the end, the path length l of the planned trajectory $\mathcal{T}$ would be returned into the GA process.

4 Experiments

In this section, we evaluate the effectiveness of our proposed method by comparing it with other non-experience-based state-of-the-art optimization and motion planning algorithms. We assess the performance in terms of path length with different stopping criteria in simulation experiments.

4.1 Experimental setups

4.1.1 Construction of the problems

Regarding benchmark tests for the environment arrangement solver, we defined several environment constraints for constructing different pick-and-place tasks. As shown in Figure 9, three picking directions, namely, vertical (A), horizontal (B), and back-forward (C), and three placing directions, namely, vertical (D), horizontal (E), and back-forward (F), are designed for setting the tasks. In addition, whether the box exists in the environment is also considered an environment constraint. Therefore, in total, 36 benchmark problems are created for testing. Among them, tasks are separated into hard and easy tasks, according to the direction and existence of the box. Picking or placing from the vertical sides often involves motions that are primarily in the vertical or near-vertical plane. This can result in fewer rotational degrees of freedom required for the robotic arm. Therefore, benchmark problems when no box is in the environment or the box is placed in the vertical side are defined as easy tasks, and benchmark problems when the box placed on the horizontal side and back-forward side are considered as hard tasks. In total, 18 easy tasks and 18 hard tasks were designed.

4.1.2 Implementation of the methods

For the two layers of this hierarchical structural problem-solving method, two experience-based methods and two state-of-the-art non-experience-based methods were utilized for the test. For the upper level of the hierarchical structure, which aims to solve the optimization problem, GA (Pattanaik et al., 2018b) and case-injected GA (Louis and McDonnell, 2004) are used. For the lower level of the hierarchical structure, which aims to solve the motion planning problem, motion planners RRTConnect algorithm (Kuffner and LaValle, 2000) and ERTConnect algorithm (Pairet et al., 2021) are introduced. Therefore, , in total, 2 (for higher layer) × 2 (for lower layer) = 4 combinations of environment arrangement solvers, namely, GA-RRTC, caseGA-RRTC, GA-ERTC, and caseGA-ERTC, are introduced for solving this environment arrangement problem.

4.1.3 Equipment utilized for the experiments

Four robots with different sizes and configuration structures are utilized for testing the effectiveness of our proposed method:

- xArm6: 6-DOF, maximum reach: 691 mm
- Elfin5: 6-DOF, maximum reach: 800 mm
- Motoman mhj: 6-DOF, maximum reach: 909 mm
- AUBO-i7: 6-DOF, maximum reach: 1,150 mm.

4.1.4 Environment settings

The center positions of the box and the left conveyor were aligned, and the sizes of the boxes and conveyors

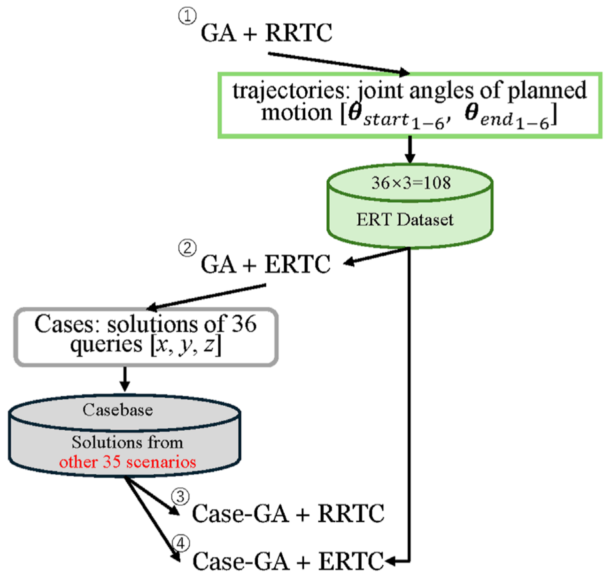

① GA + RRTC
trajectories: joint angles of planned motion [$\boldsymbol{\theta}_{start_{1-6}}$, $\boldsymbol{\theta}_{end_{1-6}}$]
36×3=108
ERT Dataset
② GA + ERTC
Cases: solutions of 36 queries [x, y, z]
Casebase
Solutions from
other 35 scenarios
③ Case-GA + RRTC
④ Case-GA + ERTC

FIGURE 7
Sequence of the execution of experiments.

were set in proportion to the reach of each robotic arm, as stated in Section 4.1.3. The height of the box was set to 0.2R, and the width of the box was made equal to the width of the conveyor at 0.3R. The lengths of the box and the conveyor were set to 0.65R and 1.36R, respectively, where R represents the maximum reach of each robot.

4.2 Dataset setups

Dataset collection is related to the sequential execution of experiments. Therefore, we designed four steps to execute our experiments, as shown in Figure 7. In the beginning, 36 environment arrangement problems were solved via the GA-RRTC algorithm. The trajectories of the final results and other randomly selected two trajectories were stored into the dataset, which would be used for the ERTConnect algorithm in later experiments; therefore, in total, 108 trajectories were stored into the ERT dataset.

Trajectories were represented and stored by the angles of joints of the robotic arm in configuration space corresponding to each discrete point in 3D space. However, only the start configuration $\boldsymbol{\theta}_{start_{1-6}}$ and goal configuration $\boldsymbol{\theta}_{goal_{1-6}}$ were used in the evaluation of the similarity according to Equation 1.

Then, the same environment arrangement problems are solved by the combination of the GA-ERTC algorithm. Here, the final solutions of 36 queries (x, y, and z) are seen as cases to be saved into the casebase. However, to ensure the casebase used by the query does not contain the solution that was solved by the same problem, only 35 solutions would be saved into the casebase for each query. These casebases would be used for case-injected GA for the next two experiment testings: the case-injected GA-RRTC algorithm and case-injected GA-ERTC algorithm.

5 Results and discussion

We have done simulation experiments in four robots (AUBO-i7, Elfin5, mhj, and xArm6, as shown in Figure 8) with easy tasks and hard tasks using four combinations of methods mentioned above. Here, we access our results by calculating the average value of the path length (rad) of the 18 tasks when stopping the environment arrangement algorithm at different times. For the auboi7 robot, whose size is the largest, the time spent for planning feasible motions was longer than other robots with smaller sizes. Therefore, for the AUBO-i7 robot only, the stopping time criteria are longer than the three other robots.

Figure 10 shows an example of our obtained results, in which the AUBO-i7 robot was used to solve the environment optimization problem when performing the task involves picking from the horizontal side and placing from the vertical side. The final positions of the conveyors are $[x_1, y_1, z_1] = [0.3524, 0.6594, 0.3137]$m and $[x_2, y_2, z_2] = [0.3524, -0.6594, 0.3137]$m, with the path length of the trajectory as 2.98 rad. According to the results obtained from the simulation shown in Table 1, the proposed experience-based environment arrangement solver case-injected GA-ERTC showed the highest performance in most of the experimental tasks, especially in robots with small sizes, such as Elfin5, xArm6, and mhj robots. When stopping the algorithm in the 2000s in easy tasks and 3000s in hard tasks, our proposed experience-based method shows better results. In experiments conducted in the Elfin5 robot, there is a decrease in the final result (path length) of 8.06% in easy tasks and 19.64% in hard tasks. In the mhj robot, there is a decrease in the final result of 0.54% in easy tasks, 8.99% in hard tasks. In the xArm6 robot, there is a decrease in the final result of 6.55% in easy tasks and 9.15% in hard tasks.

However, when stopping the algorithm in 5000s in hard tasks of the large robot AUBO-i7, our proposed experience-based method did not show good performance. There might be mainly two reasons for this phenomenon.

1) The experience-reuse efficiency in the AUBO-i7 robot is lower than other robots. As shown in Table 2, we calculated the average distance (rad) between the selected experience and current query when using the ERTConnect algorithm and compared the results between the AUBO-i7 robot and Elfin5 robot. The results show that the distance in AUBO-i7 cases is approximately twice that of Elfin5 cases. This represents that experiences reused in AUBO-i7 could be more different with a new query; therefore, the experience-reuse efficiency would be lower than small robots.
2) The hard tasks of the AUBO-i7 robot are more difficult than other robots because of the environment settings (the set of the box). We set the size of the boxes in proportion to the reach of the robots, not scaled to the robot's configuration space. Thus, the large size of the robot AUBO-i7 increases the potential for collisions. Therefore, the motion planning in the large robot, AUBO-i7, is much harder than that in small robots, which leads to the increase in final results (path length) in hard tasks of the AUBO-i7 robot.

In reason 1, average distances between current query and selected experiences in the AUBO-i7 robot are higher than those in the Elfin5 robot. The reason for this might be related to the second

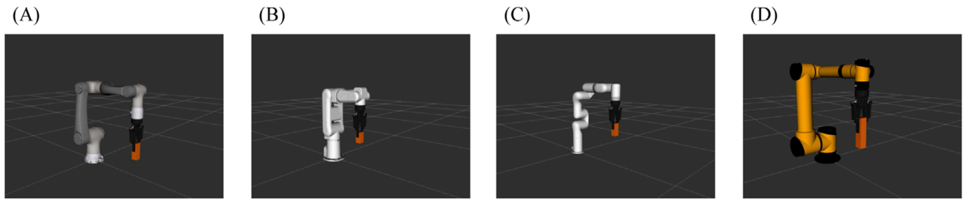


FIGURE 8
3D Model in RViz of the robots used in experiments: **(A)** Elfin5 robot. **(B)** mhj robot. **(C)** xArm6 robot. **(D)** AUBO-i7 robot.

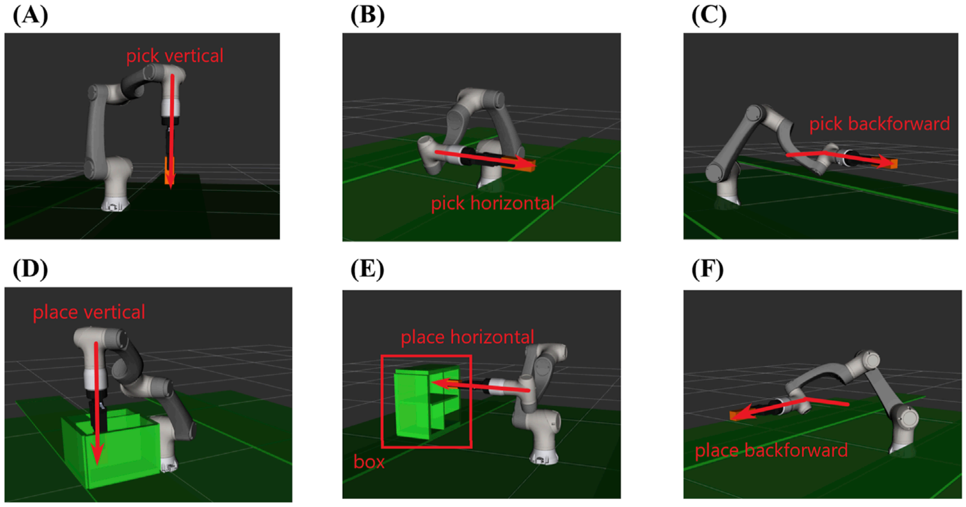


FIGURE 9
Environment constraints for defining the problem to be solved. In total, there are three picking constrains: **(A)** picking from the vertical side; **(B)** picking from the horizontal side; and **(C)** picking from the back-forward side and three placing constraints: **(D)** placing from the vertical side; **(E)** placing from the horizontal side; and **(F)** placing from the back-forward side. Whether the box [as shown in **(E, F)**] exists in the environment is also considered an environment constraint.

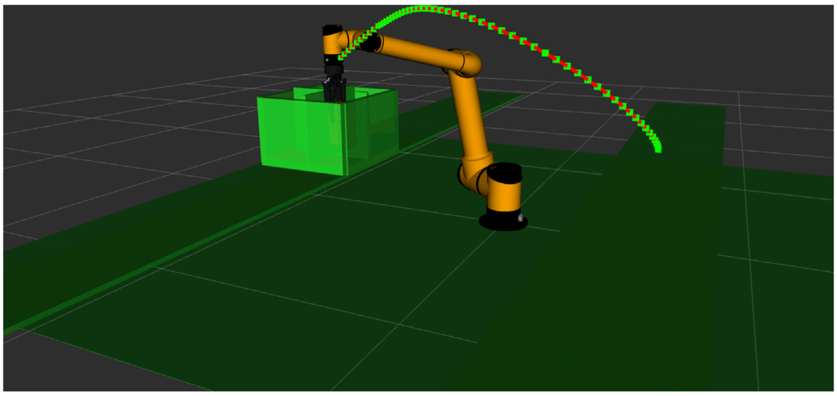


FIGURE 10
Result trajectory of the AUBO-i7 robot when performing the task involves picking from the horizontal side and placing from the vertical side.

reason. When the positions of the boxes change in the environment, because of the high difficulty of planning feasible motions of the AUBO-i7 robot, even the small changes in the positions of the box could lead to a large slip of the starting and ending configurations compared with the dataset planned by the IK module at the first stage of motion planning.

In addition to the abovementioned results, we also introduce a laddered reference for users who are willing to use our proposed

TABLE 1 Path length (rad) returned when stopping the optimization algorithm in different progresses in four robots (AUBO-i7, Elfin5, mhj, and xArm6).

	AUBO-i7 (easy tasks)		
Condition	Path length (1000s)	Path length (2000s)	Path length (3000s)
GA-RRT	2.407574	2.247776	2.210905
caseGA-RRT	2.515666	2.346976	2.295159
GA-ERT	2.478247	2.31395	2.259126
caseGA-ERT	2.486048	2.242487	**2.167826** (−1.9%)
	AUBO-i7 (hard tasks)		
Condition	Path length (1000s)	Path length (3000s)	Path length (5000s)
GA-RRT	2.747944	2.480159	**2.391671**
caseGA-RRT	3.041041	2.680214	2.572279
GA-ERT	2.638694	2.505068	2.469904
caseGA-ERT	2.723162	2.569482	2.459562
	Elfin5 (easy tasks)		
Condition	Path length (800s)	Path length (1400s)	Path length (2000s)
GA-RRT	1.911629	1.892187	1.880808
caseGA-RRT	1.909614	1.895111	1.886558
GA-ERT	1.838229	1.790196	1.780574
caseGA-ERT	1.787515	1.737445	**1.729197** (−8.06%)
	Elfin5 (hard tasks)		
Condition	Path length (1000s)	Path length (2000s)	Path length (3000s)
GA-RRT	3.582743	3.448506	3.378888
caseGA-RRT	3.403129	3.12497	3.073381
GA-ERT	2.985809	2.832983	2.780494
caseGA-ERT	2.956463	2.781493	**2.715406** (−19.64%)
	mhj (easy tasks)		
Condition	Path length (800s)	Path length (1400s)	Path length (2000s)
GA-RRT	2.303998	2.274186	2.236647
caseGA-RRT	2.29979	2.276456	2.259014
GA-ERT	2.254757	2.246878	2.246562
caseGA-ERT	2.234666	2.229086	**2.224507** (−0.54%)
	mhj (hard tasks)		
Condition	Path length (1000s)	Path length (2000s)	Path length (3000s)
GA-RRT	3.437869	3.323296	3.249903
caseGA-RRT	3.371937	3.235782	3.149138
GA-ERT	3.111261	3.048727	2.990963
caseGA-ERT	3.057014	2.979651	**2.960559** (−8.99%)
	xArm6 (easy tasks)		
Condition	Path length (800s)	Path length (1400s)	Path length (2000s)
GA-RRT	1.930181	1.899212	1.848715
caseGA-RRT	1.939437	1.911066	1.866544
GA-ERT	1.794222	1.784214	**1.775055**
caseGA-ERT	1.835677	1.811967	1.793110 (−6.55%)
	xArm6 (hard tasks)		
Condition	Path length (1000s)	Path length (2000s)	Path length (3000s)
GA-RRT	3.383504	3.259521	3.158789
caseGA-RRT	3.78115	3.415445	3.380669
GA-ERT	3.209672	2.973886	2.939468
caseGA-ERT	3.130803	2.976547	**2.869729** (−9.15%)

The best results are highlighted in bold

(Continued on the following page)

TABLE 1 (*Continued*) Path length (rad) returned when stopping the optimization algorithm in different progresses in four robots (AUBO-i7, Elfin5, mhj, and xArm6).

TABLE 2 Average distance (Rad) between the current query and selected experience of AUBO-i7 and Elfin5 robot in the pick-and-place task from the back-forward side when there is box in the environment.

	AUBO-i7	Elfin5
Average distance (rad)	3.8717168	1.2448932

methods. In real manufacturing, tolerable planning time might be shorter than our proposed maximum stopping time. Therefore, our results also show how long the path lengths of the designed environments can be in a shorter planning time (800s, 1400s, and 2000s), which could be reused in practical environment arrangement problems for engineers.

6 Conclusion and future work

In this study, we proposed an experience-based environment arrangement solver to solve the combination problem of environment arrangement and motion planning. To solve this high-dimensional problem, the hierarchical structure is proposed to decompose the high-dimensional problem into two semi-high-dimensional problems. Case-injected GA was used for solving the optimization problem, and the path length planned by the ERTConnect algorithm was used as an evaluation criterion in the process of genetic algorithm. We compared our proposed experience-based methods with non-experience-based state-of-the-art methods.

The results of simulation experiments showed that our proposed experience-based method case-injected GA-ERTC algorithm can sufficiently re-utilize the past experiences and design the environment with the average 5.05% decrease in path length in 18 easy and 12.59% in 18 hard pick-and-place tasks in three robots (Elfin5, mhj, and xArm6).

However, there are still several limitations in this study. First, as shown in our results, the effectiveness of our proposed method could be unclear when being used in the large robot while increasing the difficulty of tasks. Second, our goal is to solve problems and increase the quality of the final results within specified limited time, so we do not seek to ensure the optimality of the result. Third, we have only tested our proposed method on 6-DOF robots, so it would be another interesting study if we could extend our vision to 7-DOF robots.

Therefore, there could be three future directions for our work. First, for different types of tasks, a switch between methods can be proposed. For simpler tasks without obstacles, non-experience-based methods, which do not require referencing previous experiences, could be employed to increase the calculation speed. However, for more complex tasks where the success rate of non-experience-based motion planning methods is low, experience-based approaches would be more suitable. Second, discussion about the optimality of the optimization algorithms could be another direction. Third, since our proposed methods are only tested on 6-DOF robots, the transfer of experiences among 6-DOF robots and 7-DOF robots could also be an interesting research topic.

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: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing–original draft, and writing–review and editing. RT: conceptualization, methodology, software, supervision, and writing–review and editing. YW: conceptualization, methodology, supervision, and writing–review and editing. JO: funding acquisition, methodology, project administration, resources, supervision, and writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research was under support by the New Energy and Industrial Technology Development Organization (NEDO) under project JPNP20016. This paper represents a collaborative outcome from joint research with the ROBOT Industrial Basic Technology Collaborative Innovation Partnership.

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

Abdelwahed, M. F., Saleh, M., and Mohamed, A. E. (2018). Speeding up single-query sampling-based algorithms using case-based reasoning. *Expert Syst. Appl.* 114, 524–531. doi:10.1016/j.eswa.2018.08.035

Berenson, D., Abbeel, P., and Goldberg, K. (2012). "A robot path planning framework that learns from experience," in *2012 IEEE International Conference on Robotics and automation (IEEE)*, 3671–3678.

Bulut, V. (2023). Path planning of mobile robots in dynamic environment based on analytic geometry and cubic bézier curve with three shape parameters. *Expert Syst. Appl.* 233, 120942. doi:10.1016/j.eswa.2023.120942

Coleman, D., Şucan, I. A., Moll, M., Okada, K., and Correll, N. (2015). *Experience-based planning with sparse roadmap spanners.* 2015 IEEE International Conference on Robotics and Automation ICRA IEEE, 900–905.

Davies, S. (2009). A matter of takt, 4, 62, 65. doi:10.1049/et.2009.0915*[manufacturing takt time]* (Engineering and Technology).

Fiechter, C.-N. (1994). A parallel tabu search algorithm for large traveling salesman problems. *Discrete Appl. Math.* 51, 243–267. doi:10.1016/0166-218x(92)00033-i

Fu, Y.-T., Hsu, C.-M., Chen, Z.-Y., and Chou, J.-H. (2020). "Path planning for continuous-curvature avoidance using hierarchical four parameter logistic curves," in *2020 IEEE international conference on systems, man, and cybernetics (SMC)* (IEEE), 3358–3363.

Gueta, L., Chiba, R., Ota, J., Arai, T., and Ueyama, T. (2007). Design and optimization of a manipulator-based automated inspection system. *SICE Trans. Industrial Appl.* 6, 41–51.

Hsu, D., Latombe, J.-C., and Motwani, R. (1997). Path planning in expansive configuration spaces. *Proc. Int. Conf. robotics automation (IEEE)* 3, 2719–2726. doi:10.1109/robot.1997.619371

Jahanshahi, H., Jafarzadeh, M., Sari, N. N., Pham, V.-T., Huynh, V. V., and Nguyen, X. Q. (2019). Robot motion planning in an unknown environment with danger space. *Electronics* 8, 201. doi:10.3390/electronics8020201

Kuffner, J. J., and LaValle, S. M. (2000). "Rrt-connect: an efficient approach to single-query path planning," in Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065) (IEEE), 995–1001. doi:10.1109/robot.2000.844730

Levant, A., and Livne, M. (2011). Exact differentiation of signals with unbounded higher derivatives. *IEEE Trans. Automatic Control* 57, 1076–1080. doi:10.1109/tac.2011.2173424

Li, Z., Li, S., Bamasag, O. O., Alhothali, A., and Luo, X. (2022a). Diversified regularization enhanced training for effective manipulator calibration. *IEEE Trans. Neural Netw. Learn. Syst.* 34, 8778–8790. doi:10.1109/tnnls.2022.3153039

Li, Z., Li, S., Francis, A., and Luo, X. (2022b). A novel calibration system for robot arm via an open dataset and a learning perspective. *IEEE Trans. Circuits Syst. II Express Briefs* 69, 5169–5173. doi:10.1109/tcsii.2022.3199158

Li, Z., Li, S., and Luo, X. (2021). An overview of calibration technology of industrial robots. *IEEE/CAA J. Automatica Sinica* 8, 23–36. doi:10.1109/jas.2020.1003381

Liu, D., Dissayanake, G., Manamperi, P., Brooks, P., Fang, G., Paul, G., et al. (2008). "A robotic system for steel bridge maintenance: research challenges and system design," in *Proceedings of the 2008 australasian conference on robotics and automation, ACRA 2008.*

Louis, S. J., and McDonnell, J. (2004). Learning with case-injected genetic algorithms. *IEEE Trans. Evol. Comput.* 8, 316–328. doi:10.1109/tevc.2004.823466

Lu, J., Takamido, R., and Ota, J. (2023). "Experience-based problem solver for robot system design," in *2023 9th international conference on automation, robotics and applications (ICARA)* (IEEE), 55–59.

Pairet, È., Chamzas, C., Petillot, Y., and Kavraki, L. E. (2021). Path planning for manipulation using experience-driven random trees. *IEEE Robotics Automation Lett.* 6, 3295–3302. doi:10.1109/lra.2021.3063063

Pattanaik, J. K., Basu, M., and Dash, D. P. (2018a). Improved real coded genetic algorithm for dynamic economic dispatch. *J. Electr. Syst. Inf. Technol.* 5, 349–362. doi:10.1016/j.jesit.2018.03.002

Pattanaik, J. K., Basu, M., and Dash, D. P. (2018b). Improved real coded genetic algorithm for dynamic economic dispatch. *J. Electr. Syst. Inf. Technol.* 5, 349–362. doi:10.1016/j.jesit.2018.03.002

Phillips, M., Dornbush, A., Chitta, S., and Likhachev, M. (2013). "Anytime incremental planning with e-graphs," in *2013 IEEE international Conference on Robotics and automation (IEEE)*, 2444–2451.

Tereshchuk, V., Stewart, J., Bykov, N., Pedigo, S., Devasia, S., and Banerjee, A. G. (2019). An efficient scheduling algorithm for multi-robot task allocation in assembling aircraft structures. *IEEE Robotics Automation Lett.* 4, 3844–3851. doi:10.1109/lra.2019.2929983

Zeng, Q., Liu, X., Qiu, C., and Li, A. (2019). "Kinematic characteristics analysis of cooperative welding robot with multiple manipulators," in *2019 4th asia-pacific conference on intelligent robot systems (ACIRS)* (IEEE), 1–5.