ISA Transactions xxx (xxxx) xxx

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

Practice article

Computation-efficient path planning using evolving artificial repulsive force over expanding obstacles for robotic manipulator

Yi Zhou [a,b], Bowen Huang [a], Yim Ho Cheung [a,b], Pengfei Ye [a], Molong Duan [a,b,c,*]

[a] Department of Mechanical and Aerospace Engineering, The Hong Kong University of Science and Technology, Hong Kong Special Administrative Region of China
[b] HKUST-BYD Embodied AI Joint Laboratory, The Hong Kong University of Science and Technology, Hong Kong Special Administrative Region of China
[c] Center for Embodied AI and Computer Vision, Shenzhen Loop Area Institute, Shenzhen, Guangdong, China

ARTICLE INFO

Keywords:
Computation-efficient path planning
Artificial repulsive force
Artificial potential field
Obstacle expansion
Obstacle cluster
Robotic manipulator

ABSTRACT

With growing demand for high productivity in industrial environments, fast path planning for robotic manipulators is critical, as planning time directly impacts production cycle efficiency. Traditional path planning methods, such as grid-based methods, artificial potential field methods, sampling-based methods, and optimization methods, often suffer from high computation costs in high-dimensional space. In particular, artificial potential field methods require the construction of global potential fields or the computation of gradients and attractive forces, increasing the computational burden. To address these issues, a computation-efficient path planning method, using the evolving artificial repulsive force over the expanding obstacles (EARFEO), is proposed. EARFEO incorporates obstacle clustering and virtual obstacle insertion with box-envelope strategy to mitigate the local minima, and iteratively deforms the path using artificial repulsive forces generated by expanding obstacles. A uniform resampling strategy is employed after each expansion step to maintain consistent waypoint distribution, followed by a final refinement phase to ensure complete clearance of the whole path after obstacle expansion is complete. Simulations and experiments across static and dynamic scenarios demonstrate that EARFEO reduces planning time significantly.

1. Introduction

Robotic manipulators have been developed in automation technology for high productivity [1]. Efficient path planning is critical for robot systems [2]. Robot manipulation in dense environments requires fast and obstacle-avoidance planning under dynamic scene variations, such as bin-picking, construction, and manufacturing [3–6]. The overall planning time of industrial robots directly influences cycle times and economic efficiency of automation in production lines. Efficient and fast path planning for robotic manipulators greatly improves the efficiency of production lines [7].

Robot path planning is to find the collision-free path between the given initial state and target state of the robot in the configuration space under the constraints [8]. Path planning methods are divided into grid-based methods, artificial potential field (APF) methods, sampling-based methods, and optimization methods [9]. Grid-based methods discretize the configuration space into a set of states connected by feasible paths, allowing motion planning problems to be formulated as graph search [10]. Dijkstra incrementally expands the shortest known distances from the start node, selecting the closest unvisited node at each step and updating the path costs to neighboring nodes [11], while A* extends Dijkstra's algorithm by introducing a heuristic cost, guiding the search toward the goal and accelerating the process by pruning the search space [12]. Weighted-A* dynamically adjusts the heuristic weight inversely proportional to the depth in the search tree, producing a narrower depth-first search than traditional weightings [13]. The heuristics of A* are inherited by D*, and D* reuses prior search effort to accommodate edge-cost updates in dynamic scenes [14]. A modified A-Star algorithm adopts the bidirectional sector variable step search strategy to reduce the computation time [15]. Anytime-A* quickly finds a suboptimal solution and incrementally improves the suboptimal solution toward optimality as time allows [16]. However, these methods are originally designed for 2D motion planning, and their computational cost increases significantly when extended to high-dimensional path planning [1].

Sampling-based algorithms randomly sample the configuration

* Correspondence to: Room 2570, Academic Building, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China.
E-mail addresses: yzhouev@connect.ust.hk (Y. Zhou), bhuangau@connect.ust.hk (B. Huang), yhcheungam@connect.ust.hk (Y.H. Cheung), pyeae@connect.ust.hk (P. Ye), duan@ust.hk (M. Duan).

https://doi.org/10.1016/j.isatra.2026.07.028
Received 12 March 2026; Received in revised form 31 July 2026; Accepted 31 July 2026
Available online 1 August 2026

space to explore the connectivity, enabling efficient planning in complex environments [17]. Probabilistic Roadmap (PRM) builds a graph of collision-free configurations during a learning phase and searches for a path by connecting the start and goal in a query phase [18]. Rapidly-Exploring Random Trees (RRT) grows a tree by incrementally extending toward randomly sampled points to explore high-dimensional spaces [19]. RRT-Connect accelerates feasibility by bidirectional extensions [20]. Asymptotic optimality is achieved with RRT* by incrementally rewiring the tree to reduce path cost during exploration [21]. Informed-RRT* improves RRT* by focusing sampling on a heuristic-defined subset of the state space after finding an initial solution, enabling faster convergence [22]. AMRRT* explores the configuration space with the multi-tree sampling structure [23]. Hybrid-RRT* utilizes both non-uniform and uniform samplers to achieve a balance between exploitation and exploration [24]. Sampling-based algorithms perform sampling over the entire configuration space, leading to high computational cost and suboptimal paths, requiring time-consuming post-processing for refinement [25].

Optimization-based methods formulate obstacle-avoidance motion planning as an optimization problem that minimizes task-related objectives subject to collision-avoidance, kinematic, and physical constraints [17]. Classical trajectory optimization methods include CHOMP, STOMP, TrajOpt, GPMP2, etc. CHOMP uses covariant gradient techniques to improve the quality of sampled trajectories [26]. STOMP employs a derivative-free stochastic optimization approach to iteratively minimize cost functions that may be non-differentiable or non-smooth [27]. TrajOpt discretizes the path and applies sequential convex optimization, iteratively linearizing collision constraints and solving a series of quadratic programs [28]. GPMP2 models the trajectory as a Gaussian process and formulates planning as sparse nonlinear least-squares inference on a factor graph [29]. Optimization-based methods yield locally optimal solutions and may need to be repeated with different initial conditions to obtain a feasible result, potentially leading to high computational cost [30]. In addition to trajectory-level planners, constraint-optimization-based methods explicitly encode trajectory tracking, joint limits, singularity avoidance, manipulability, obstacle avoidance, and task-priority requirements as equality or inequality constraints [31]. An HQP-based prescribed-performance control framework is proposed for redundant mobile manipulators, where trajectory tracking, physical constraints, obstacle avoidance, and avoidance-direction guidance are organized according to task priorities [32]. Quadratically constrained quadratic programming for end-effector trajectory generation is combined with QP-based whole-body reactive obstacle avoidance for nonholonomic mobile manipulators [33]. Constraint-optimization-based methods enable explicit constraint handling and task prioritization, but generally require solving constrained optimization problems at each planning or control step, and computational cost may increase with the number of constraints and prioritized tasks [31].

The artificial potential field methods guide a robot in configuration space by modeling the goal as an attractive force and obstacles as repulsive forces, directing motion based on the resulting vector field [34]. APF is prone to getting stuck in local minima. Random movements are introduced when the robot gets trapped in a local minimum, but this approach takes a long time to escape in some configurations and may reduce path smoothness [35]. The annealing and tempering strategies are combined with a potential function to avoid being trapped in local minima [36]. To mitigate local minima and accelerate convergence, APF is integrated into RRT* to balance exploration and exploitation, such as P-RRT* [37]. PQ-RRT* adds Q-RRT*'s fast-convergence strategy to P-RRT*, yielding better initial solutions and quicker asymptotic optimality [38]. DBVS-APF-RRT* improves RRT* by directionally biased variable step sizes, artificial potential fields, and pruning strategies to optimize path quality and stability with accelerated initial path generation [39]. However, the randomness of the search range, the repeatability of the collision check, and the generation of large invalid nodes lead to a significant computational cost [40].

In this work, to enable fast collision-free path planning, path planning using evolving artificial repulsive force over expanding obstacles is proposed. Different from conventional APF methods that guide robots using global attractive and repulsive potential fields, EARFEO formulates obstacle avoidance as a path-level deformation process. Instead of computing a goal-directed attractive force or constructing a global potential field, EARFEO first generates an initial path to the target and then progressively reshapes it using repulsive effects from expanding obstacles. This formulation improves planning efficiency. EARFEO also differs from APF-enhanced RRT variants, as it does not rely on stochastic sampling or extensive collision checking over random nodes. Furthermore, obstacle pre-clustering and virtual-obstacle insertion with the box-envelop strategy modify the obstacle topology before deformation, helping mitigate local-minimum trapping. The core contributions are summarized as follows:

- Proposed the evolving artificial repulsive force to push the initial path away from obstacles during the path planning process, eliminating the need for constructing a global potential field or computing gradients and attractive forces.
- Developed an obstacle expansion strategy to gradually push the path away from obstacles, avoiding excessive adjustments, and deformed the path consecutively through iterative repulsion to ensure a collision-free path upon completion of the obstacle expansion.
- Designed obstacles pre-clustering diagram and introduced virtual obstacles within each cluster and box-envelope strategy to mitigate the risk of getting trapped in local minima.
- Reallocated discrete sampling points along the reshaped path after obstacle expansion to ensure uniform distribution, as the artificial repulsive force naturally causes point aggregation in low-repulsion regions and dispersion in high-repulsion regions.
- Demonstrated the computational efficiency, collision avoidance capability, and applicability of the proposed EARFEO method across multiple industrial scenarios.

The detailed paper is structured as follows: Computation-efficient path planning using the evolving artificial repulsive force over expanding obstacles method is detailed in Section 2. Simulations and verifications are detailed in Section 3. Finally, the conclusion is illustrated in Section 4.

2. Computation-efficient path planning using evolving artificial repulsive force over expanding obstacles

2.1. Path planning problem for robotic manipulators

Considering the robotic manipulator with υ joints, the robot joint state is defined as

$$\boldsymbol{q} = [q_1, q_2, \cdots, q_\upsilon]^\top. \tag{1}$$

The position and orientation of the endpoints of each link are computed using forward kinematics. The pose of the end of the i-th link is defined as

$$\begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{\phi}_i \end{bmatrix} = f_i(\boldsymbol{q}), \tag{2}$$

where $\boldsymbol{x}_i(\boldsymbol{q})$ denotes the position, and $\boldsymbol{\phi}_i(\boldsymbol{q})$ represents the Euler angles of the i-th link (following the ZYX convention). The function $f_i(\cdot)$ represents the forward kinematics mapping from joint space to the position and orientation of the i-th link end in the task space. The robot joint state connects the linear and angular velocity of the i-th link end with the Jacobian matrix $\boldsymbol{J}_i \in \mathbb{R}^{6\times\upsilon}$, defined as

$$\begin{bmatrix}\dot{\boldsymbol{x}}_i\\ \boldsymbol{\omega}_i\end{bmatrix}=\underbrace{\begin{bmatrix}\boldsymbol{J}_{v,i}(\boldsymbol{q})\\ \boldsymbol{J}_{\omega,i}(\boldsymbol{q})\end{bmatrix}}_{\triangleq \boldsymbol{J}_i(\boldsymbol{q})}\dot{\boldsymbol{q}}. \tag{3}$$

where $\boldsymbol{J}_{v,i}(\boldsymbol{q})\in\mathbb{R}^{3\times v}$ and $\boldsymbol{J}_{\omega,i}(\boldsymbol{q})\in\mathbb{R}^{3\times v}$ are the translational and rotational parts of the geometric Jacobian, respectively. Here, $\boldsymbol{\omega}_i$ denotes the angular velocity of the end of the i-th link. Define $\boldsymbol{X}$ as the configuration space. $\boldsymbol{X}_o$ is the obstacle space. $\boldsymbol{X}_f=\boldsymbol{X}/\boldsymbol{X}_o$ is the free space. Denote the start configuration $\boldsymbol{q}_0\in\boldsymbol{X}_f$, goal configuration $\boldsymbol{q}_1\in\boldsymbol{X}_f$, lower and upper bounds of joint movement $\underline{\boldsymbol{q}}$ and $\overline{\boldsymbol{q}}$, and path parameter ξ. The objective is to find a collision-free path $\boldsymbol{\psi}$,

$$\begin{aligned}&\boldsymbol{\psi}:[0,1]\rightarrow\mathbf{X}_f,\\ &\text{s.t. }\boldsymbol{\psi}(0)=\boldsymbol{q}_0,\boldsymbol{\psi}(1)=\boldsymbol{q}_1,\\ &\boldsymbol{\psi}(\xi)\in\mathbf{X}_f,\ \ \underline{\boldsymbol{q}}\le\boldsymbol{\psi}(\xi)\le\overline{\boldsymbol{q}},\forall\xi\in[0,1].\end{aligned} \tag{4}$$

To fairly compare deterministic and stochastic planners, the same set of start-goal pairs is employed for all methods. The evaluation reports average performance over these start-goal pairs. Two evaluation settings are considered: single-shot evaluation and best-of-N_b evaluation. Single-shot evaluation measures each planner's performance in a single execution. Best-of-N_b evaluation measures the best performance obtained from N_b independent executions. The evaluation uses four basic performance metrics: success rate, planning time, path length, and path curvature. For each start-goal pair, a planning trial returns a success outcome, a planning time, and, if successful, a path length and a path curvature.

- Success outcome s: A value of $s=1$ indicates that a valid collision-free path is found. A value of $s=0$ indicates that no valid path is found.
- Planning time t_c: The planning time is defined as the time from the start of the planning query to the termination of the planner. The planner terminates when either a feasible path is found or the maximum allowed planning time is reached.
- Path length l_e: The path length is computed for successful trials, defined as the cumulative distance of the end effector along the discretized planned path $\mathscr{C}=\left\{\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_{N_d}\right\}$,

$$l_e=\sum_{i=2}^{N_d}\|\boldsymbol{x}_v(\boldsymbol{q}_i)-\boldsymbol{x}_v(\boldsymbol{q}_{i-1})\|_2, \tag{5}$$

  where $\boldsymbol{x}_v$ denotes the end-effector position of the robotic manipulator. N_d denotes the number of waypoints in the planned path, with $N_d>1$.
- Path curvature c_e: The path curvature is computed only for successful trials. A smaller curvature value indicates a smoother path. The average curvature c_e along the end-effector path is then computed as

$$\begin{aligned}&c_e=\frac{1}{N_d-2}\sum_{i=2}^{N_d-1}\frac{\theta_i}{\frac{1}{2}(\|\Delta\boldsymbol{x}_{i+1}\|_2+\|\Delta\boldsymbol{x}_i\|_2)},\\ &\theta_i=\arccos\left(\frac{\Delta\boldsymbol{x}_{i+1}^{\top}\Delta\boldsymbol{x}_i}{\|\Delta\boldsymbol{x}_{i+1}\|_2\|\Delta\boldsymbol{x}_i\|_2}\right),\\ &\Delta\boldsymbol{x}_i=\boldsymbol{x}_v(\boldsymbol{q}_i)-\boldsymbol{x}_v(\boldsymbol{q}_{i-1}),\ i=2,\cdots,N_d-1.\end{aligned} \tag{6}$$

In the single-shot evaluation, each start-goal pair is evaluated using one independent planning trial. For a deterministic planner, this trial gives a fixed output for the given input. For a stochastic planner, this trial gives one random execution result. Given M start-goal pairs, the single-shot success rate R_s^s is computed as

$$R_s^s=\frac{1}{M}\sum_{j=1}^{M}s_j, \tag{7}$$

where s_j denotes the success outcome of the single trial for the j-th start-goal pair. Let $\mathscr{S}^s=\{j|s_j=1\}$ be the set of successfully solved start-goal pairs. The reported single-shot planning time, path length, and path curvature are defined as the average performance over all start-goal pairs,

$$t_c^s=\frac{1}{|\mathscr{S}^s|}\sum_{j\in\mathscr{S}^s}t_{c,j},\ l_e^s=\frac{1}{|\mathscr{S}^s|}\sum_{j\in\mathscr{S}^s}l_{e,j},\ c_e^s=\frac{1}{|\mathscr{S}^s|}\sum_{j\in\mathscr{S}^s}c_{e,j}, \tag{8}$$

where $t_{c,j}$, $l_{e,j}$, and $c_{e,j}$ denote the planning time, path length, and path curvature of the single trial for the j-th start-goal pair.

In the best-of-N_b evaluation, each start-goal pair is evaluated using the best performance of N_b independent planning trials for each planner, as the outputs of stochastic planners may vary across runs. For deterministic planners, repeated runs under the same input produce identical outputs. For the j-th start-goal pair and the m-th trial, let $s_{j,m}$, $t_{c,j,m}$, $l_{e,j,m}$, and $c_{e,j,m}$ denote the success outcome, planning time, path length, and path curvature. Here, $m=1,2,\ldots,N_b$. The reported best-of-N_b metrics reflect the best attainable performance from N_b independent trials. Given M start-goal pairs, the reported Best-of-N_b success rate is

$$R_s^b=\frac{1}{M}\sum_{j=1}^{M}s_j^b,s_j^b=\max_{1\le m\le N_b}s_{j,m}. \tag{9}$$

Let $\mathscr{S}^b=\left\{j|s_j^b=1\right\}$ be the set of start-goal pairs solved by at least one trial. The reported best-of-N_b planning time, path length, and path curvature are

$$\begin{aligned}&t_c^b=\frac{1}{|\mathscr{S}^b|}\sum_{j\in\mathscr{S}^b}t_{c,j}^b,\ \ t_{c,j}^b=\min_{m:s_{j,m}=1}t_{c,j,m},\\ &l_e^b=\frac{1}{|\mathscr{S}^b|}\sum_{j\in\mathscr{S}^b}l_{e,j}^b,\ \ l_{e,j}^b=\min_{m:s_{j,m}=1}l_{e,j,m},\\ &c_e^b=\frac{1}{|\mathscr{S}^b|}\sum_{j\in\mathscr{S}^b}c_{e,j}^b,\ \ c_{e,j}^b=\min_{m:s_{j,m}=1}c_{e,j,m}.\end{aligned} \tag{10}$$

In addition to single-shot and best-of-N_b evaluation, the time to first successful generation (TTFS) of the path across trials for each planner is reported. Let $\mathscr{S}^t=\{j|\exists m,\ s_{j,m}=1\}$ be the set of start-goal pairs solved by at least one trial. The reported TTFS planning time is

$$t_c^t=\frac{1}{|\mathscr{S}^t|}\sum_{j\in\mathscr{S}^t}t_{c,j}^t,\ \ t_{c,j}^t=\sum_{m=1}^{m_j^*}t_{c,j,m},\ \ m_j^*=\min\ \{m|s_{j,m}=1\}. \tag{11}$$

2.2. Evolving artificial repulsive force over expanding obstacles

To improve planning efficiency in high-dimensional configuration spaces, this section presents EARFEO, a path-planning method based on evolving artificial repulsive forces over expanding obstacles. As shown in Fig. 1(a), EARFEO starts from a linearly interpolated path between the start and goal configurations, followed by obstacle clustering and virtual-obstacle insertion in narrow regions. For clusters intersecting the initial path, a box-envelope-based deformation trial is applied to promote global bypassing of obstacle groups. Otherwise, the standard clustered-obstacle representation is used. The path is then iteratively deformed by goal attraction and repulsive torques from progressively expanded obstacles, as shown in Fig. 1(b). After collision checking, refinement, simplification, and smoothing, EARFEO returns a feasible collision-free path.

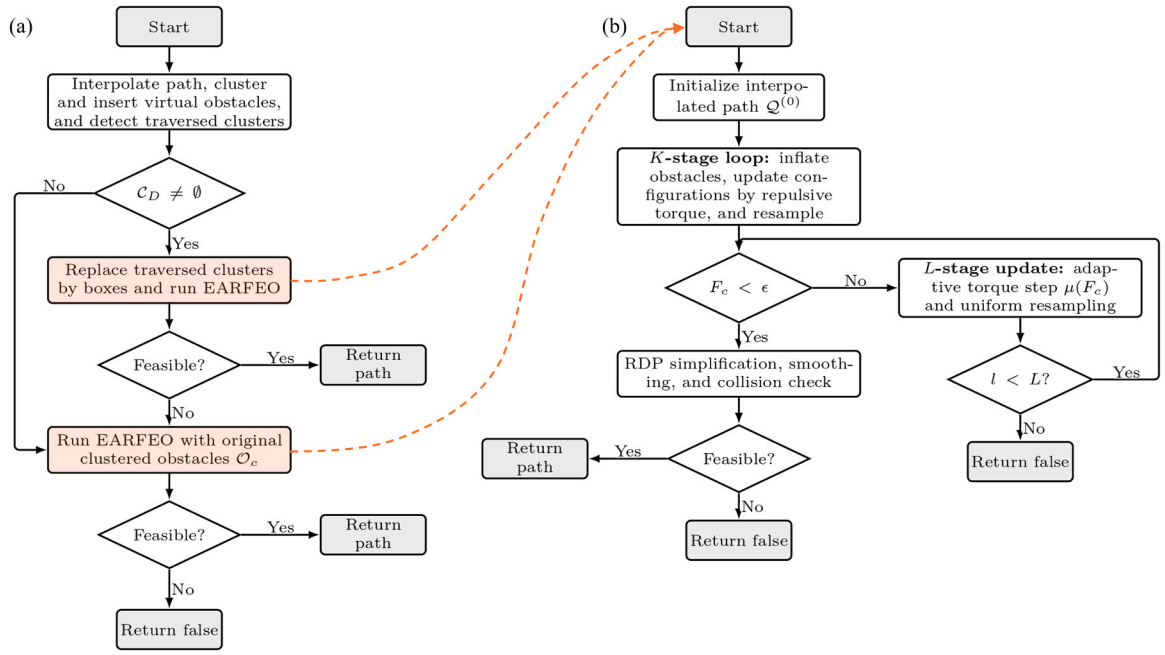

Fig. 1. (a) Overall workflow of the proposed path planning method process, (b) detailed schematics of EARFEO.

2.2.1. Obstacle clustering

Considering the issue of local minima in the traditional APF method, an obstacle preprocessing technique that involves clustering and virtual obstacle generation is proposed. The key idea is to classify the original obstacles $\mathscr{O}_s$ into clusters based on the pairwise minimum distance, and connect two obstacles with a virtual obstacle according to (14), as shown in Fig. 2(a). After clustering and virtual-obstacle insertion, the resulting obstacle set is denoted as $\mathscr{O}_\mathscr{C}$. An undirected graph $G(\mathscr{O}_s, A)$ is constructed to represent the connections. The undirected graph allows multiple obstacles to be grouped into the same cluster based on mutual connectivity if the obstacles are transitively connected. Each node $o_m \in \mathscr{O}_s$ corresponds to an obstacle. The adjacency matrix A encodes whether two obstacles are spatially close, with A_{mn} denoting the minimum distance d_{mn} between nodes o_m and o_n,

$$A_{mn} = \begin{cases} 1, & 0 \le d_{mn} \le \bar{\varepsilon} \\ 0, & \text{otherwise} \end{cases}. \tag{12}$$

where $\bar{\varepsilon}$ is the predefined upper bound. To avoid inserting virtual obstacles between intersecting, overlapping, or nearly coincident obstacles, a separation matrix T is defined as,

$$T_{mn} = \begin{cases} 1, & d_{mn} \ge \underline{\varepsilon}, \\ 0, & \text{otherwise}. \end{cases} \tag{13}$$

where $\underline{\varepsilon}$ is the predefined lower distance bound. The virtual-obstacle insertion matrix M is defined as

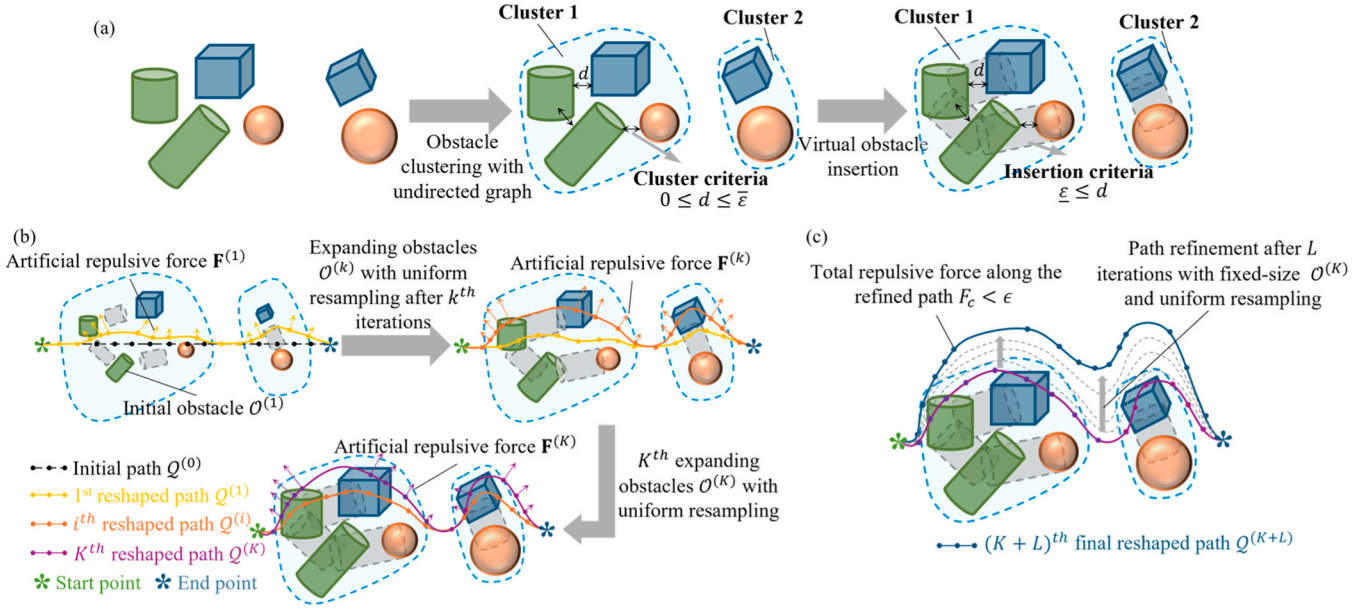

Fig. 2. Overview of the EARFEO pipeline. (a) Obstacle clustering via an undirected graph and virtual obstacle insertion based on the cluster and insertion criteria. (b) Expansion-based path deformation using repulsive forces from progressively expanded obstacles with uniform resampling. (c) Final refinement under fixed fully expanded obstacles until the total repulsive force falls below the threshold.

$$M_{mn} = A_{mn} T_{mn}, \tag{14}$$

where $M_{mn} = 1$ indicates that a virtual obstacle is inserted between obstacles o_m and o_n. Therefore, a virtual obstacle is inserted only when the two obstacles within one cluster satisfy $\underline{\varepsilon} \le d_{mn} \le \overline{\varepsilon}$, meaning that these two obstacles are close enough to form a narrow passage but still have a valid positive separation. The clustering and virtual-obstacle generation step transforms the original obstacle map into a structured representation that captures spatial proximity and suppresses narrow passages that may cause local minima in the APF-based planning process.

The goal-directed behavior is implicitly embedded in the initial path $\boldsymbol{\psi}_0(\xi)$, defined via linear interpolation between the start and goal configuration,

$$\boldsymbol{\psi}_0(\xi) = (1-\xi)\boldsymbol{q}_0 + \xi \boldsymbol{q}_1, \quad \xi \in [0,1]. \tag{15}$$

The initial path serves as the baseline path, independent of obstacle influence. For efficient computation of repulsive forces, the continuous path $\boldsymbol{\psi}_0(\xi)$ is uniformly discretized into $\mathscr{Q}^{(0)}$ with N_d waypoints,

$$\mathscr{Q}^{(0)} = \left\{ \boldsymbol{q}_i^{(0)} = \boldsymbol{\psi}_0\left(\frac{i-1}{N_d - 1}\right) \right\}_{i=1}^{N_d}. \tag{16}$$

Based on $\mathscr{Q}^{(0)}$, the clusters intersected by the initial path are detected and denoted as $\mathscr{C}_{\mathscr{I}}$. If $\mathscr{C}_{\mathscr{I}} \neq \varnothing$, each traversed cluster is temporarily replaced by an oriented bounding box (OBB) that encloses all obstacles within the cluster [41]. Only the traversed clusters are replaced, while the remaining obstacles retain their original geometric representations. The generated OBB is treated as an ordinary box obstacle in the subsequent EARFEO procedure and uses the same procedures and parameters as the original obstacles. This box replacement aims to improve the success rate by alleviating the path from being trapped among multiple obstacles within the cluster. If the box-based attempt fails, the original clustered obstacle set $\mathscr{O}_{\mathscr{C}}$, including the inserted virtual obstacles, is restored for a second EARFEO attempt.

2.2.2. Path deformation under expanding obstacles

In this stage, obstacles are progressively expanded to guide the path deformation process. The expansion is coupled with an evolving repulsive force field that is applied to an initial path, gradually steering the path away from obstacle regions, as shown in Fig. 2(b). As the expansion size is gradually increased, the path is continuously guided to the collision-free area using the expanded-obstacle repulsive force field. The progressive obstacle expansion approach ensures that the path is guided into a safe collision-free region without requiring the path to overcome steep repulsive forces in a single step, mitigating overshooting. To model the progressive expansion of obstacles in a shape-consistent manner, a uniform geometric scaling approach is adopted. Given an original j-th obstacle $\mathscr{O}_j$, the expanded obstacle at the k-th step is defined as

$$\mathscr{O}_j^{(k)} = g^{(k)} \mathscr{O}_j, \quad g^{(k)} = \frac{k}{K}, \quad k = 1, 2, \cdots, K, \tag{17}$$

where K is the expansion iterations, $g^{(k)}$ is the scaling factor, and the operation $g^{(k)}\mathscr{O}_j$ denotes isotropic scaling of the obstacle's geometric parameters, centered at the origin, ensuring that the shape of the obstacle is preserved while the obstacle's size increases proportionally. The expanded obstacle set at the k-th iteration is denoted as $\mathscr{O}^{(k)}$. For each path point $\boldsymbol{q}_i^{(k)}$ in the k-th iteration, the repulsive force $\boldsymbol{F}_{imn}^{(k)} \in \mathbb{R}^6$ is defined as

$$\boldsymbol{F}_{imn}^{(k)} = \begin{cases} \left[\boldsymbol{f}_{imn}^{(k)}, 0, 0, 0\right]^\top & d_{mn}^{(k)} < d_0 \\ [0,0,0,0,0,0]^\top & d_{mn}^{(k)} \ge d_0 \end{cases}, \tag{18}$$

where $\boldsymbol{f}_{imn}^{(k)}$ denotes the translational repulsive force applied at the closest sampled point on the m-th link due to the n-th obstacle, given by

$$\boldsymbol{f}_{imn}^{(k)} = \eta \left(\frac{1}{d_{mn}^{(k)} + \delta} - \frac{1}{d_0} \right) \boldsymbol{d}^\top, \tag{19}$$

where η is a scaling coefficient for the repulsive force, $d_{mn}^{(k)}$ is the minimum distance between the link cylinder m and the obstacle n in the k-th iteration, $\boldsymbol{d}$ is the safety threshold distance for EARFEO's repulsive-field construction, and $\boldsymbol{d}$ is the direction vector, given by

$$\boldsymbol{d} = \frac{{}^w\mathbf{p}_{im} - {}^w\mathbf{c}_n + \delta \mathbf{v}}{\left\| {}^w\mathbf{p}_{im} - {}^w\mathbf{c}_n + \delta \mathbf{v} \right\|_2}, \tag{20}$$

where ${}^w\mathbf{p}_{im} \in \mathbb{R}^3$ is the closest sampled point in the link cylinder m to the obstacle n in the i-th path point, and $\mathbf{v} \in \mathbb{R}^3$ is a randomly sampled unit vector in case $\left\| {}^w\mathbf{p}_{im} - {}^w\mathbf{c}_n \right\|_2$ is zero. This zero-distance case is a degenerate geometric case where the repulsive direction is undefined. The random unit vector is used as a numerical regularization to provide a nonzero escape direction and avoid division by zero. δ is a small threshold (e.g., $\delta = 10^{-16}$), avoiding numerical instability. For each link, the link-obstacle distance is evaluated using multiple sampled points. This provides a closest-point-based discrete approximation of the distributed link-level repulsive effect, where the closest sampled point represents the locally most critical collision-risk region and determines the dominant obstacle-avoidance direction. Accordingly, the repulsive force $\boldsymbol{F}_{imn}^{(k)}$ is generated at the closest sampled point ${}^w\mathbf{p}_{im}$ on the m-th link at the i-th path point. The saturated repulsive force $\boldsymbol{F}_{imn}^{(k)}$ is defined as,

$$\boldsymbol{F}_{imn}^{(k)} = \begin{cases} \boldsymbol{F}_{imn}^{(k)} & \left\| \boldsymbol{F}_{imn}^{(k)} \right\|_2 \le F_{\max} \\ F_{\max} \dfrac{\boldsymbol{F}_{imn}^{(k)}}{\left\| \boldsymbol{F}_{imn}^{(k)} \right\|_2} & \left\| \boldsymbol{F}_{imn}^{(k)} \right\|_2 > F_{\max} \end{cases}, \tag{21}$$

where $F_{\max}$ denotes the maximum allowable magnitude of the repulsive force. For torque computation, this point force $\boldsymbol{F}_{imn}^{(k)}$ is converted into an equivalent endpoint wrench $\widetilde{\boldsymbol{F}}_{imn}^{(k)}$. This wrench preserves the resultant force and includes the moment induced by the offset from the closest point to the endpoint. The corresponding joint-space torque $\tau_{imn}^{(k)}$ is computed via the Jacobian matrix,

$$\tau_{imn}^{(k)} = \boldsymbol{J}_m^\top \left(\boldsymbol{q}_i^{(k-1)} \right) \widetilde{\boldsymbol{F}}_{imn}^{(k)}, \tag{22}$$

where $\boldsymbol{J}_m\left(\boldsymbol{q}_i^{(k-1)}\right)$ is the Jacobian of the m-th link endpoint at configuration $\boldsymbol{q}_i^{(k-1)}$. The path is updated iteratively,

$$\boldsymbol{q}_i^{(k)} = \boldsymbol{q}_i^{(k-1)} + \lambda \tau_i^{(k)}, \quad \tau_i^{(k)} = \sum_m \sum_n \tau_{imn}^{(k)}. \tag{23}$$

where λ is a step size and $\tau_i^{(k)}$ is the total torque at the configuration $\boldsymbol{q}_i^{(k-1)}$. Intermediate path points are often perturbed by repulsive forces to avoid obstacles. The updated path $\mathscr{Q}^{(k)}$ may become non-uniformly distributed, especially near obstacles, where points are pushed away. The non-uniform spacing leads to inaccuracies in collision checking, particularly when large gaps exist between consecutive configurations. A uniform resampling strategy is employed to address the non-uniform spacing problem. The resampling is mainly used to maintain the joint-space resolution for collision checking after repulsive-force-based path deformation, instead of approximating the Cartesian arc-length. Given a maximum joint difference threshold $\theta_{\max}$, segment i from $\boldsymbol{q}_i^{(k)}$ to $\boldsymbol{q}_{i+1}^{(k)}$ is subdivided to ensure that the largest joint-wise difference remains below the threshold. Specifically, the maximum joint difference for segment i is defined as

$$\Delta_i = \left\| \boldsymbol{q}_{i+1}^{(k)} - \boldsymbol{q}_i^{(k)} \right\|_\infty. \tag{24}$$

The infinity norm identifies the maximum displacement among all joints and thus provides a direct component-wise measure of joint-space variation. The number of interpolation points n_i for each segment i is determined by

$$n_i = \max\left(1, \frac{\Delta_i}{\theta_{\max}}\right). \tag{25}$$

Linear interpolation for each segment produces a densified path $\widetilde{\mathscr{C}}^{(k)}$, where the total number of waypoints after interpolation is generally larger than N_d. This subdivision ensures that any two consecutive configurations in the densified path satisfy

$$\left\| \widetilde{\boldsymbol{q}}_{r+1}^{(k)} - \widetilde{\boldsymbol{q}}_r^{(k)} \right\|_\infty \le \theta_{\max}. \tag{26}$$

This enforces a per-joint resolution bound of $\theta_{\max}$ between adjacent collision-checking configurations, explicitly limiting the maximum joint-wise motion that may affect the workspace swept region of the robotic manipulator. To enforce uniform spacing, the densified path $\widetilde{\mathscr{C}}^{(k)}$ is resampled using a joint-space arc-length parameterization. The joint-space arc-length of each segment is computed based on the maximum joint-wise difference between consecutive configurations. Based on the cumulative arc-length, N_d new configurations are interpolated at equally spaced positions, producing a uniformly spaced path $\overline{\mathscr{C}}^{(k)}$, and then $\mathscr{C}^{(k)} = \overline{\mathscr{C}}^{(k)}$. The resulting uniformly resampled path serves as the input for the next iteration of repulsive perturbation, as shown in Fig. 2(b). After K iterations of path deformation with repulsive force and uniform resampling, a preliminary path $\overline{\mathscr{C}}^{(K)} \in \mathbb{R}^{N_d \times \nu}$ is obtained.

2.2.3. Final refinement and post-processing

Although the progressive obstacle expansion effectively guides the path away from regions with high repulsive forces, the obstacle expansion strategy does not strictly guarantee a collision-free path. To further enhance the safety margin, a final refinement stage is introduced. In this stage, repulsive forces continue to be applied to the path without further expanding the obstacles, allowing the path to be smoothly adjusted until full clearance is achieved, as shown in Fig. 2(c). Denote the number of refinement iterations L. At each refinement step $l \in \{1, 2, \cdots, L\}$, the path from the previous step after the uniform resampling is denoted as $\overline{\mathscr{C}}^{(K+l-1)} = \left\{ \overline{\boldsymbol{q}}_1^{(K+l-1)}, \cdots, \overline{\boldsymbol{q}}_{N_d}^{(K+l-1)} \right\}$, and then $\mathscr{C}^{(K+l-1)} = \overline{\mathscr{C}}^{(K+l-1)}$. The updated path is

$$\boldsymbol{q}_i^{(K+l)} = \boldsymbol{q}_i^{(K+l-1)} + \mu(F_c)\tau_i^{(K+l)}, \tag{27}$$

where $\tau_i^{(K+l)}$ is calculated by using (18)-(22) and $\mu(F_c)$ is an adaptive step size, confining the joint movement under large virtual forces to avoid over-adjustment and promoting the path adjustment under small forces, and the total repulsive force magnitude F_c along the entire path is computed as

$$\mu(F_c) = \underline{\lambda} + (\overline{\lambda} - \underline{\lambda})(1 - \exp(-\gamma/F_c)),\ F_c = \sum_i \sum_m \sum_n \left\| \boldsymbol{F}_{imn}^{(K+l)} \right\|_2, \tag{28}$$

where $\overline{\lambda}$ and $\underline{\lambda}$ are the upper and lower bound thresholds of the step size, γ is the scaling coefficient, $\boldsymbol{F}_{imn}^{(K+l)}$ denotes the repulsive force exerted on the m-th link at the i-th path point due to the n-th obstacle during the $(K + l)$-th iteration. The adaptive rule in (28) is designed based on the obstacle-proximity implication of the repulsive force. A large F_c indicates strong obstacle influence; therefore, $\mu(F_c)$ approaches the lower bound $\underline{\lambda}$ to avoid excessive path deformation and potential oscillations. Conversely, when F_c is small, the path is generally away from obstacles and the collision risk is lower, so $\mu(F_c)$ increases toward the upper bound $\overline{\lambda}$ to accelerate refinement under weak repulsive forces. The updated path $\mathscr{C}^{(K+l)}$ is then resampled using the same uniform resampling strategy. The refinement is terminated when the total repulsive force magnitude F_c along the entire path $\mathscr{C}_f$ falls below a predefined threshold ϵ. This condition guarantees that no significant repulsive force remains along the path, implying effective collision avoidance. Since $\mu(F_c) \in [\underline{\lambda}, \overline{\lambda}]$ and the refinement terminates once $F_c < \epsilon$, the proposed rule balances conservative updates near obstacles with faster adjustment in low-risk regions.

While the refinement stage ensures that the path maintains sufficient clearance from obstacles, the resulting path may still contain unnecessary detours, local loops, or abrupt directional changes introduced during repulsive perturbations. To eliminate these artifacts in the path while preserving the essential shape, a post-processing step with the Remer-Douglas-Peucker (RDP) algorithm [42] is introduced. The path $\mathscr{C}_f$ is mapped into workspace coordinates $\mathscr{P}_f$ through forward kinematics. The RDP algorithm is applied to the position sequence $\mathscr{P}_f$, using a user-defined threshold ϵ_p to limit the maximum allowable deviation. The algorithm operates recursively by evaluating the perpendicular distance of intermediate points to the line connecting the endpoints. Points exceeding the deviation threshold are retained as key waypoints, while the rest are discarded. This simplification process effectively removes minor loops and redundant fluctuations in the original path, without altering the global topology, ensuring a cleaner and more efficient execution path $\overline{\mathscr{P}_f}$ in the workspace (with the joint-space counterpart $\overline{\mathscr{C}_f}$), as shown in Fig. 3. To achieve smooth transitions between waypoints, cubic polynomials are applied over $\overline{\mathscr{C}_f}$. For each pair of consecutive points, a cubic polynomial segment is constructed such that the overall path is smooth, with continuous first and second derivatives across segment boundaries. The path is considered collision-free if the total repulsive force F_c along the entire path $\widetilde{\mathscr{C}}_f$ falls below a predefined threshold ϵ. If the first planning attempt fails, the planner discards the box-shaped approximations and restores the original cluster-based obstacles $\mathscr{O}_\mathscr{C}$ with the inserted virtual obstacles, and repeats EARFEO from the original start and goal configurations. Thus, the box-shaped obstacle representation is used as a success-rate-oriented first attempt to reduce local-minimum trapping within traversed clusters.

2.2.4. Local convergence analysis

APF-based methods are generally difficult to guarantee global completeness or global convergence due to possible local minima [43,44]. As EARFEO belongs to the class of artificial-potential-field-based methods, global convergence is not claimed here. This subsection analyzes the local convergence property of the proposed method. For the i-th path point, the m-th robot link, and the n-th obstacle, let $\mathbf{x}_{im} = {}^w\mathbf{p}_{im}$ be the closest sampled point on the link, and let $d_{mn} = d_{mn}(\mathbf{x}_{im})$ denote its minimum distance to the obstacle. For each link-obstacle shortest distance satisfying $d_{mn} < d_0$, define the local potential ϕ_{imn} as

$$\phi_{imn}(d_{mn}) = \eta\left[\ln\left(\frac{d_0 + \delta}{d_{mn} + \delta}\right) - \frac{d_0 - d_{mn}}{d_0}\right], \tag{29}$$

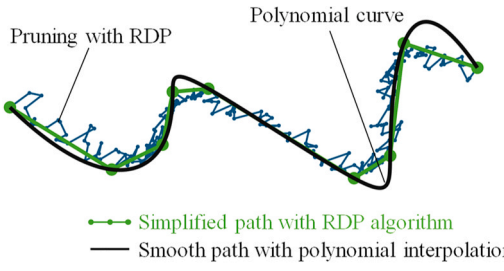

Fig. 3. Path simplification with the RDP algorithm [42] and path smoothness with a polynomial curve.

and set $\phi_{mn} = 0$ for $d_{mn} \geq d_0$. $\delta > 0$ is introduced to regularize the potential near zero distance and avoid numerical singularity. The potential ϕ_{imn} is nonnegative in the active region $d_{mn} < d_0$. In the active region $d_{mn} < d_0$, differentiating ϕ_{imn} with respect to d_{mn} gives

$$\frac{\partial \phi_{imn}}{\partial d_{mn}} = -\eta\left(\frac{1}{d_{mn}+\delta} - \frac{1}{d_0}\right). \tag{30}$$

Assuming that the distance function $d_{mn}(\mathbf{x}_{im})$ is locally differentiable,

$$\nabla_{\mathbf{x}_{im}} d_{mn} = \boldsymbol{d}, \tag{31}$$

where $\boldsymbol{d}$ is the direction vector defined in (20). By the chain rule,

$$\nabla_{\mathbf{x}_{im}} \phi_{imn} = \frac{\partial \phi_{imn}}{\partial d_{mn}} \nabla_{\mathbf{x}_{im}} d_{mn} = -\eta\left(\frac{1}{d_{mn}+\delta} - \frac{1}{d_0}\right)\boldsymbol{d}. \tag{32}$$

This translational repulsive force $\boldsymbol{f}_{imn}$ corresponds to the first three translational components of the repulsive vector defined in (19),

$$\boldsymbol{f}_{imn} = \eta\left(\frac{1}{d_{mn}+\delta} - \frac{1}{d_0}\right)\boldsymbol{d} = -\nabla_{\mathbf{x}_{im}} \phi_{imn}. \tag{33}$$

Thus, the proposed repulsive force $\boldsymbol{f}_{imn}$ is the negative gradient of the local potential ϕ_{imn} in task space. Since $\mathbf{x}_{im}$ depends on the robot configuration $\boldsymbol{q}_i$, the joint-space gradient of ϕ_{imn} is given by

$$\nabla_{\boldsymbol{q}_i} \phi_{imn} = \boldsymbol{J}_{p,im}^{\top}(\boldsymbol{q}_i) \nabla_{\mathbf{x}_{im}} \phi_{imn} = -\boldsymbol{J}_{p,im}^{\top}(\boldsymbol{q}_i) \boldsymbol{f}_{imn} = -\tau_{imn}, \tag{34}$$

where $\boldsymbol{J}_{p,im}(\boldsymbol{q}_i)$ is the translational Jacobian of the point $\boldsymbol{x}_{im}$. Define the total local potential $U_i(\boldsymbol{q}_i)$ for the i-th path point as

$$U_i(\boldsymbol{q}_i) = \sum_m \sum_n \phi_{imn}. \tag{35}$$

Then, with (34) and (35), the total local potential $U_i(\boldsymbol{q}_i)$ is differentiated as

$$-\nabla_{\boldsymbol{q}_i} U_i(\boldsymbol{q}_i) = \sum_m \sum_n \tau_{imn} = \tau_i. \tag{36}$$

Accordingly, (23) is updated with the gradient-descent calculation (36) of the local potential U_i

$$\boldsymbol{q}_i^{(k)} = \boldsymbol{q}_i^{(k-1)} - \lambda \nabla_{\boldsymbol{q}_i} U_i\left(\boldsymbol{q}_i^{(k-1)}\right). \tag{37}$$

To prove the local descent property, assume that U_i has a $\mathscr{D}$-Lipschitz continuous gradient in a compact local neighborhood, and that the iterations of (37) remain within this local neighborhood. The Lipschitz-gradient assumption is standard in local convergence analysis of gradient-based nonlinear optimization [45]. Under this assumption, the descent lemma is given by

$$U_i(\boldsymbol{q}_i^{(k)}) \leq U_i\left(\boldsymbol{q}_i^{(k-1)}\right) + \nabla_{\boldsymbol{q}_i} U_i\left(\boldsymbol{q}_i^{(k-1)}\right)^{\top}\left(\boldsymbol{q}_i^{(k)} - \boldsymbol{q}_i^{(k-1)}\right) + \frac{\mathscr{D}}{2}\left\|\boldsymbol{q}_i^{(k)} - \boldsymbol{q}_i^{(k-1)}\right\|_2^2, \tag{38}$$

Substituting the gradient-descent update (37) into (38) gives

$$U_i(\boldsymbol{q}_i^{(k)}) \leq U_i\left(\boldsymbol{q}_i^{(k-1)}\right) - \lambda(1 - \mathscr{D}\lambda/2)\left\|\nabla_{\boldsymbol{q}_i} U_i\left(\boldsymbol{q}_i^{(k-1)}\right)\right\|_2^2, \tag{39}$$

If the step size satisfies $0 < \lambda \leq 2/\mathscr{D}$, then $\lambda(1 - \mathscr{D}\lambda/2) \geq 0$. Thus, $U_i(\boldsymbol{q}_i^{(k)})$ is monotonically non-increasing. Since each ϕ_{imn} is nonnegative in the active region $d_{mn} < d_0$ and is zero outside the active region, then $U_i(\boldsymbol{q}_i) \geq 0$. $U_i(\boldsymbol{q}_i) \geq 0$ locally converges to a stationary point. Therefore, the analysis provides a local convergence of the repulsive update.

2.3. Minimum distance evaluation between robot links and the obstacles

To support efficient and accurate computation of repulsive forces, the geometric modeling of obstacles and the analytic methods used for distance computation are proposed. Both real and virtual obstacles are represented using a common set of canonical primitives. To simplify the collision checking, obstacles are approximated using three primitive objects: spheres, cylinders, and boxes, and the links of the robotic manipulator are approximated as cylinders, allowing for efficient analytical computation.

Table 1 summarizes the parameters used to define objects. Each object is defined by the geometric parameters, position, and orientation. The orientation is represented by Euler angles in the ZYX convention, and the corresponding rotation matrix ${}^{w}_{\kappa}R(\boldsymbol{\vartheta}_\kappa)$ is computed to transform coordinates from the world frame w to the object's local frame κ. In addition to real obstacles, virtual obstacles $\mathscr{O}_v$ are incorporated as auxiliary guidance elements to facilitate collision-free path generation. These virtual obstacles are designed to steer the path away from undesirable regions, such as areas prone to local minima. Virtual obstacles are placed in the free space between adjacent real obstacles within each cluster, mitigating the risk of local minima. For computational simplicity and geometric consistency, the virtual obstacle $\mathscr{O}_v$ is modeled as a virtual cylinder, defined as

$$\mathbf{c}_v^w = \frac{\mathbf{c}_m^w + \mathbf{c}_n^w}{2}, \quad h_v = \left\|\mathbf{c}_m^w - \mathbf{c}_n^w\right\|_2, \quad r_v = \min(r_m, r_n), \tag{40}$$

where ${}^{w}\mathbf{c}_v$, h_v, and r_v represent the geometric parameters of the virtual obstacle $\mathscr{O}_v$, adaptively determined based on the spatial configuration of the original obstacles. ${}^{w}\mathbf{c}_\alpha$ ($\alpha = m, n$) means the center position in the world frame and r_α is the radius of the object α. The orientation of the virtual cylinder is aligned with the normalized vector,

$$\mathbf{w} = \left[w^{(1)}, w^{(2)}, w^{(3)}\right]^{\top} = \frac{{}^{w}\mathbf{c}_m - {}^{w}\mathbf{c}_n}{\max\left(\left\|{}^{w}\mathbf{c}_m - {}^{w}\mathbf{c}_n\right\|_2, \delta\right)}. \tag{41}$$

The vector $\mathbf{w}$ denotes the regularized direction from ${}^{w}\mathbf{c}_m$ to ${}^{w}\mathbf{c}_n$. It reduces to the corresponding unit vector when $\left\|{}^{w}\mathbf{c}_m - {}^{w}\mathbf{c}_n\right\|_2 > \delta$, while the lower bound δ prevents singular normalization for nearly coincident points. The cylinder orientation is then expressed in Euler angles as

$$\boldsymbol{\vartheta}_v = \left[0, \sin^{-1}\left(-w^{(3)}\right), \tan^{-1}\left(w^{(2)}/w^{(1)}\right)\right]^{\top}, \tag{42}$$

where the roll angle is fixed arbitrarily at zero for simplicity. In cases where $w^{(1)} = 0$, the yaw is set to $\pm\pi/2$ according to the sign of $w^{(2)}$.

With the geometric representations of both real and virtual obstacles established, the minimum distance between these objects is introduced. For three primitive shape objects, 6 types of minimum distances are defined as follows. For sphere-sphere, the minimum distance between spheres m and n is

Table 1
Definition of the three primitive geometric shape objects' parameters.

Type	Sphere	Cylinder	Box
Center position	${}^{w}\mathbf{c} \in \mathbb{R}^3$	${}^{w}\mathbf{c} \in \mathbb{R}^3$	${}^{w}\mathbf{c} \in \mathbb{R}^3$
Orientation	N/A	$\boldsymbol{\vartheta} \in \mathbb{R}^3$	$\boldsymbol{\vartheta} \in \mathbb{R}^3$
Size parameter	Radius r	Radius r; Height h	Edge lengths $\mathbf{s} = \left[s^{(1)}, s^{(2)}, s^{(3)}\right]^{\top}$; Virtual radius $r = \left(s^{(1)} + s^{(2)} + s^{(3)}\right)/3$
Local frame	Origin at ${}^{w}\mathbf{c}$; Axes arbitrary	Origin at ${}^{w}\mathbf{c}$; Local Z-axis aligned with the cylinder axis; Cross-section in the XY plane	Origin at ${}^{w}\mathbf{c}$; Axes aligned with the three edge directions

$$d = \max\left(\left\|{}^{w}\mathbf{c}_m - {}^{w}\mathbf{c}_n\right\|_2 - (r_m + r_n), 0\right). \tag{43}$$

For sphere-box, the minimum distance between any point ${}^{w}\mathbf{p}$ and box n is

$$\begin{aligned} d_b &= \left\|\mathbf{c}_{mn}^{n} - \mathbf{p}_{mn}^{n}\right\|_2, \\ \mathbf{c}_{mn}^{n} &= R_n^w(\vartheta_n)\left(\mathbf{p}_w - \mathbf{c}_n^w\right), \\ \mathbf{p}_{mn}^{n} &= \min\left(\max\left(\mathbf{c}_{mn}^{n}, -\mathbf{s}_n/2\right), \mathbf{s}_n/2\right), \end{aligned} \tag{44}$$

where ${}^{n}\mathbf{c}_{mn}$ is the point ${}^{w}\mathbf{p}$ described in the local box n frame, and ${}^{n}\mathbf{p}_{mn}$ is the closest point on the box surface computed by clamping each coordinate of the point within the box bounds if ${}^{n}\mathbf{p}_{mn}$ is outside the box. Otherwise, ${}^{n}\mathbf{p}_{mn}$ equals to ${}^{n}\mathbf{c}_{mn}$ if ${}^{n}\mathbf{p}_{mn}$ is inside the box. For sphere-box, the point ${}^{w}\mathbf{p}$ equals to the sphere center ${}^{w}\mathbf{c}_m$.The minimum distance between sphere m and box n is

$$d = \max(d_b - r_m, 0). \tag{45}$$

For sphere-cylinder, the minimum distance between any point ${}^{w}\mathbf{p}$ and cylinder n is

$$\begin{aligned} d_c &= \sqrt{d_r^2 + d_z^2}, \\ d_r &= \max\left(\left\|\left(c_{mn}^{(1)}, c_{mn}^{(2)}\right)\right\|_2 - r_n, 0\right), \\ d_z &= \max\left(\left\|c_{mn}^{(3)}\right\|_1 - h_n/2, 0\right), \end{aligned} \tag{46}$$

where d_r and d_z are the radial and vertical distances beyond the cylinder boundary, and ${}^{n}\mathbf{c}_{mn} = \left[{}^{n}c_{mn}^{(1)}, {}^{n}c_{mn}^{(2)}, {}^{n}c_{mn}^{(3)}\right]^{\top}$ is the sphere center described in the local cylinder n frame. The minimum distance between sphere m and cylinder n is

$$d = \max(d_c - r_m, 0). \tag{47}$$

For cylinder-box, the point ${}^{w}\mathbf{p}$ is along the central axis of the cylinder m. The computation of the minimum distance between the cylinder m and box n is to find the nearest point on the cylinder's central axis closest to the box n based on (44). The minimum distance between the cylinder and the box is approximated to find the minimum distance between the sampling point and box n,

$$d = \max\left(\min_{{}^{w}\mathbf{p}} d_b\left({}^{w}\mathbf{p}\right) - r_m, 0\right). \tag{48}$$

For cylinder-cylinder, similarly, the computation of the minimum distance between these points and the cylinder n is similar to (46). The minimum distance between the two cylinders is

$$d = \max\left(\min_{{}^{w}\mathbf{p}} d_c\left({}^{w}\mathbf{p}\right) - r_m, 0\right). \tag{49}$$

For box-box, the minimum distance between two oriented bounding boxes (OBB) is computed with the separation axis theorem [41]. The local axes of the OBBs m and n are ${}^{w}\mathbf{u}_m = \left\{{}^{w}\mathbf{u}_{m1}, {}^{w}\mathbf{u}_{m2}, {}^{w}\mathbf{u}_{m3}\right\}$ and ${}^{w}\mathbf{u}_n = \left\{{}^{w}\mathbf{u}_{n1}, {}^{w}\mathbf{u}_{n2}, {}^{w}\mathbf{u}_{n3}\right\}$ in the world frame, respectively. The intersection state of the OBB is determined with 15 separate axes $\mathscr{U}$,

$$\mathscr{U} = \left\{{}^{w}\mathbf{u}_{mi}, {}^{w}\mathbf{u}_{ni}, {}^{w}\mathbf{u}_{mi} \times {}^{w}\mathbf{u}_{ni}\right\}, (i,j = 1,2,3). \tag{50}$$

The projection radius on the axis $\mathbf{u} \in \mathscr{U}$ $\left(\mathbf{u} \in \mathbb{R}^3\right)$ for two OBBs is

$$\begin{aligned} \rho_m(\mathbf{u}) &= \sum_i s_m^{(i)} \left\|\mathbf{u}_{mi}^{\top}\mathbf{u}\right\|_1/2, \\ \rho_n(\mathbf{u}) &= \sum_i s_n^{(i)} \left\|\mathbf{u}_{ni}^{\top}\mathbf{u}\right\|_1/2, i = 1,2,3. \end{aligned} \tag{51}$$

The minimum distance between two boxes is

$$d = \max\left(\min_{\mathbf{u}\in\mathscr{U}}\left(\left\|\left({}^{w}\mathbf{c}_m - {}^{w}\mathbf{c}_n\right)^{\top}\mathbf{u}\right\|_1 - \rho_m(\mathbf{u}) - \rho_n(\mathbf{u})\right), 0\right). \tag{52}$$

The simple minimum distance reduces the computation cost to analyze spatial relationships.

3. Simulation and experiment verifications

3.1. Algorithm setup

To validate the effectiveness of the proposed algorithm, simulations are conducted using the Huashu CO−603 robotic manipulator. All simulations are implemented on a laptop equipped with an Intel i7–11800H CPU (2.30 GHz) and 16 GB of RAM. The collision model of the robotic manipulator across different motion planners is simplified as cylinders. The baseline algorithms include both sampling-based (RRT-Connect, P-RRT*, PRM) and optimization-based (CHOMP, STOMP) approaches, providing a comprehensive comparison across different planning paradigms in both static environments and repetitive dynamic industrial scenarios with structural variations. Recent trends have shown that the popularity of traditional grid-based algorithms, such as A*, has been gradually declining [17]. Furthermore, due to the significant computational burden associated with high-dimensional configuration spaces, particularly in 6-DOF robotic manipulators [17], A* algorithm is excluded from the simulation.

As illustrated in Fig. 4, the cylindrical model encapsulates the actual link geometry, with the orange dashed lines indicating the central axes of the links. The pose of each cylindrical link is obtained from the robot kinematics. Oblique link orientations are transformed with the corresponding Euler angle before distance evaluation. To evaluate the minimum distance between a link and surrounding obstacles, as defined in (46)-(49), a set of a_m discrete points is uniformly sampled along the central axis of the m-th cylindrical link with radius r_m and height h_m. The point-based approximation reduces computational complexity by replacing continuous, dense collision checking with a sparse set of point-to-obstacle distance queries. Insufficient sampling density along the link axis may lead to underestimation of the true minimum distance between the link surface and nearby obstacles, potentially resulting in undetected near-collision configurations. To address the limitation while preserving computational efficiency, a radius expansion strategy is employed. The radius of each cylindrical link is artificially increased during distance evaluation, as shown in Fig. 4. Each sampled point is then associated with a spherical region with an expanded radius $\tilde{r}_m$,

$$\tilde{r}_m = \sqrt{r_m^2 + \left(\frac{h_m}{a_m - 1}\right)^2}. \tag{53}$$

The expanded radius compensates for the unsampled axial intervals and

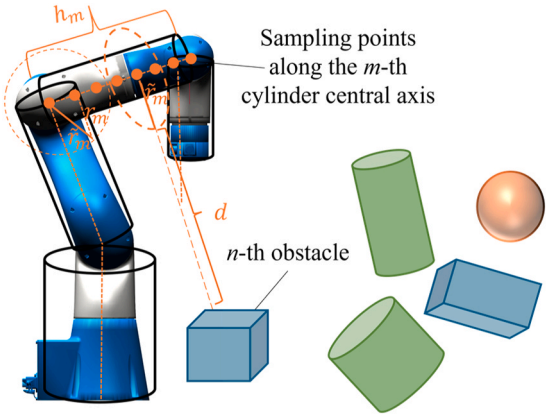

Fig. 4. Simplified representation of a robotic manipulator for collision check. Each link is approximated as a cylinder with radius r_m and height h_m, and discrete sampling points (orange dots) are uniformly distributed along the central axis (orange dashed line), where each sampled point is associated with an expanded spherical region of radius $\tilde{r}_m$.

ensures that the sampled spheres cover the original cylinder geometry. The minimum distance between the m-th link and the n-th obstacle is computed over all sampled points. The adjustment yields a conservative estimate of proximity rather than the exact continuous minimum distance, but the radius expansion reduces false-negative collision detections caused by sparse sampling and improves computational efficiency. The cylinders are aligned with the link axes, and the radius of each cylinder is set according to the geometry of the corresponding link. To ensure fair comparisons across all planners, the same simplified cylindrical collision model is also integrated into MoveIt for each baseline, so that all methods operate under the same robot geometry, collision-checking constraints, and minimum safety-distance requirement. In the EARFEO, each cylindrical segment is discretized into 7 equally spaced points along its central axis. The detailed parameters for EARFEO are listed in Table 2. The same expansion procedure and parameters are applied to all obstacle primitives, including the box envelopes generated for traversed clusters. The box-envelope attempt does not require any additional parameters.

The simulation experiments are conducted in the static and dynamic environments. To evaluate the performance of the proposed algorithm, six representative static simulation scenarios are constructed, covering a range of environments from planar and spatial setups to industrial application cases, as introduced in [40] and [46]. The details of each scenario are as follows:

- Scenario 1: Randomly distributed obstacles on a 2D plane, representing a planar tabletop manipulation task (Fig. 5(a)).
- Scenarios 2–4: Spatial environments with obstacles at varying heights and orientations, emulating cluttered 3D workspaces (Fig. 5 (b-d)).
- Scenario 5: An industrial shelf manipulation task, where the robot operates in the narrow, confined structure of a multi-layer shelf (Fig. 5(e)).
- Scenario 6: A bin-picking task, representative of common industrial automation setups (Fig. 5(f)).

Two dynamic industrial tasks are further conducted to demonstrate the applicability of EARFEO. The first task is corner-wall bricklaying, representing repetitive pick-and-place operations with structural variations and partial occlusions, as shown in Fig. 9(a). The simulation and experiment utilize a Huashu CO603 six-axis robotic manipulator equipped with a ChangingTek CTAG2F120 two-finger gripper. The task involves sequential pick-and-place operations for 18 bricks, resulting in a total of 36 paths. The environment includes moderate variations in brick placement and partial occlusions, reflecting the complexity of typical industrial scenarios. The second dynamic task is a conveyor-belt-based selective pick-and-place task, as illustrated in Fig. 9(b). Considering the growing need for agile robotic systems in unstructured production environments, a conveyor-belt-based robotic pick-and-place system is designed to enhance productivity. A moving conveyor transports multiple objects, while an Intel RealSense D435i depth camera and ArUco markers are used to localize target objects and surrounding obstacles. Target objects are grasped and placed into a predefined bin, whereas non-target objects remain on the conveyor.

Table 2
Parameter setting for EARFEO path planning.

Parameters	Value	Parameters	Value
Upper joint limit $\overline{\boldsymbol{q}}$	[180;35;230;180;130;180] °	Safe distance threshold for repulsive force d_0	25 mm
Lower joint limit $\underline{\boldsymbol{q}}$	-[180;215;50;180;130;180] °	Repulsive force coefficient η	3
Radii for the six links	[140,120,120,120,100,80] mm	Number of path sampling points N_d	25
Obstacle clustering threshold $\bar{\varepsilon}$	200 mm	Step size λ	0.001
Virtual obstacle insertion threshold $\underline{\varepsilon}$	5 mm	Upper step size bound $\overline{\lambda}$	0.016
Force saturation bound $F_{\max}$	2 N	Lower step size bound $\underline{\lambda}$	0.008
Obstacle expansion iterations K	6	Scaling coefficient γ	1
Maximum optimization Iterations L	35	Small positive constant ϵ	0.01 N
Threshold ϵ_p for pruning	30 mm	/	/

3.2. Static environment simulation

The simulation results of different path planners under 6 static scenes are shown in Fig. 5. These scenes include typical challenges such as narrow passages and cluttered workspaces. All planners obtain feasible paths in the six scenes, except CHOMP in Scenario 5. Scenario 2 is selected as a representative case for path-characteristic comparison, as shown in Fig. 6. The upper row of Fig. 6 shows the six joint movements along the normalized path parameter, whereas the lower row shows the corresponding end-effector pose in Cartesian space. Compared with sampling-based methods such as RRT-Connect, PRM, and P-RRT*, EARFEO and optimization-based methods generally produce smoother paths in both joint and Cartesian spaces, while RRT-Connect and PRM show path discontinuities. These results demonstrate the effectiveness of the expanded-obstacle guidance mechanism in promoting smooth path generation. To quantitatively evaluate the performance of each planner, four key performance metrics are considered, including planning success rate, planning time, path length, and path curvature. For each scenario, 50 randomly sampled collision-free start-goal configuration pairs are selected as test cases. The proposed EARFEO algorithm is deterministic and therefore executed once for each configuration pair. In contrast, the baseline planners are stochastic due to random sampling or heuristic-based optimization and may produce different results for the same input.

To provide a more comprehensive and fair comparison, the evaluation is conducted under two settings: single-shot and best-of-N_b. In the single-shot setting, each planner is executed once for each start-goal pair, and the resulting success rate, planning time, path length, and curvature are recorded. This setting reflects the planner's performance when only one planning attempt is allowed. In the best-of-N_b setting, each stochastic baseline planner is executed up to 100 times for each start-goal pair. A planning instance is considered successful if at least one valid, collision-free path is found within these 100 attempts. For successful cases, the best path among the successful attempts is selected according to the evaluation criterion, and its corresponding planning time, path length, and curvature are recorded. If no valid solution is obtained after 100 attempts, the planning instance is marked as a failure. Since EARFEO is deterministic, its single-shot and best-of-N_b results are identical. In addition, the paired probability of superiority is employed to report the proportion of matched successful start-goal cases where one method outperforms another on the same planning task.

Fig. 7 summarizes the scene-wise comparison of planning time, path length, curvature, and success rate. The annotated heatmap entries are the exact metric values. For planning time, only the color scale is computed using logarithmic values to improve readability, while the displayed numbers remain the original time values in seconds. Red boxes mark the best-performing method in each scene, where lower values are better for planning time, path length, and curvature, and higher values are better for success rate. As shown in the first column of Fig. 7, EARFEO achieves the shortest planning time in all six scenes under both settings. Compared with the fastest baseline, RRT-Connect, EARFEO reduces the planning time by 93% and 87% on average in the single-shot and best-of-N_b settings, respectively. The paired

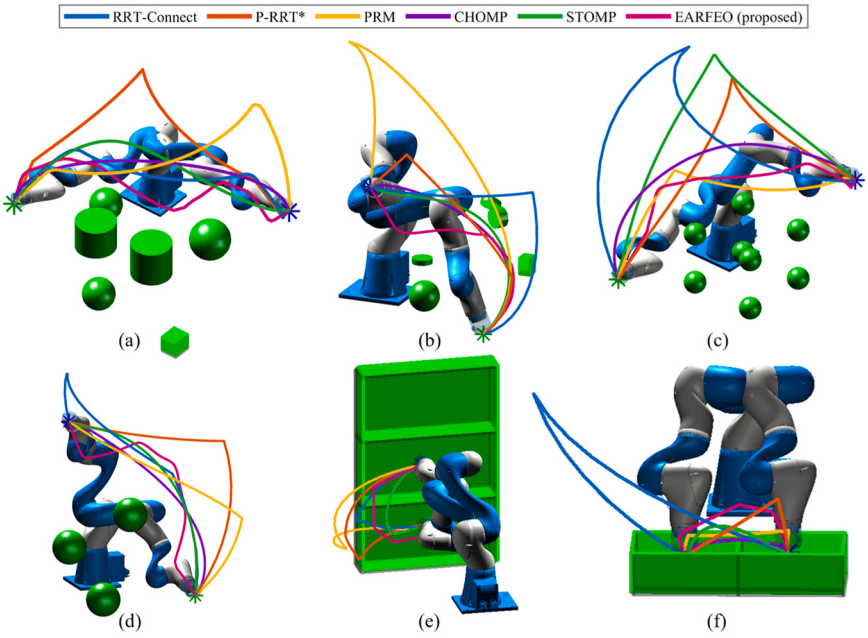

Fig. 5. Simulation results of different path planners with obstacle expansion in the six representative scenarios: (a) a typical planar tabletop manipulation task setting, (b-d) various obstacles in 3D space, (e) a typical industrial shelf manipulation task setting, (f) a common bin picking application setting.

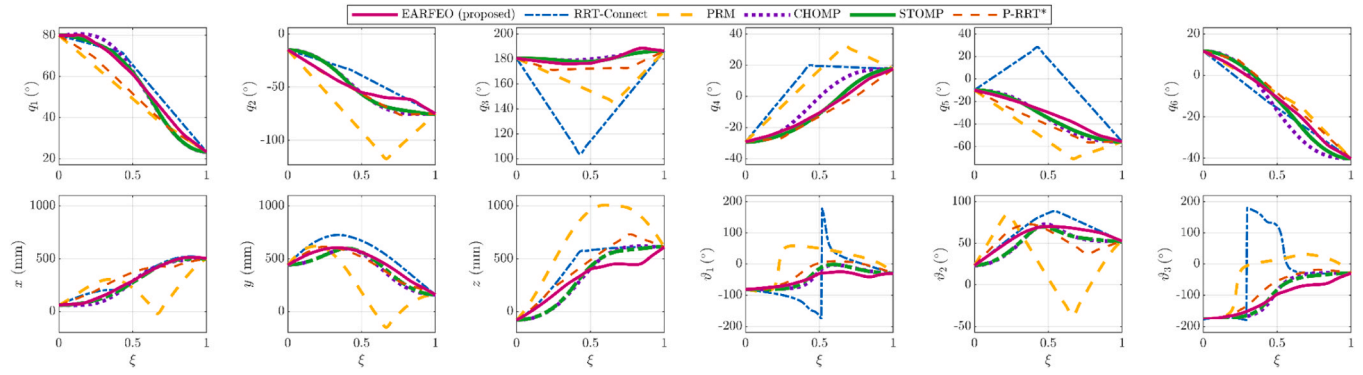

Fig. 6. Joint-space and Cartesian-space paths for Scene−2 generated by six planners. The top row illustrates the six joint angles over the path parameter ξ for different planners, while the bottom row presents the end-effector's position and orientation in Cartesian space.

probability of superiority also shows that EARFEO achieves shorter single-shot planning time than the baselines in 100% of matched successful cases. The TTFS results in Table 3 further confirm this advantage. Since EARFEO is deterministic, its TTFS is equal to its planning time. In contrast, stochastic baseline planners may require multiple attempts before obtaining the first valid path. This is particularly evident for STOMP in Scene 5 and P-RRT* in Scenes 5 and 6.

For path length, EARFEO provides competitive single-shot results. EARFEO generates shorter paths than most baseline planners across the six scenes, as shown in the second column of Fig. 7. This advantage is particularly evident for RRT-Connect and PRM, as the paths of sampling-based methods are usually longer due to random exploration. Pooled over all six scenes, the paired probability of superiority shows that EARFEO achieves shorter single-shot path lengths than the baselines in 84% of matched successful cases. In the best-of-N_b setting, stochastic planners benefit from repeated sampling and best-case selection. In particular, P-RRT* obtains the shortest average path length in Scenes 1–4 and Scene 6. These results suggest that EARFEO provides compact paths in a single deterministic execution.

For path curvature, EARFEO generates relatively low and stable single-shot performance. As shown in the third column of Fig. 7, EARFEO achieves lower curvature than RRT-Connect, PRM, and P-RRT* in all scenes, lower curvature than CHOMP in all successful scenes, and lower curvature than STOMP in Scenes 2, 4, 5, and 6. Although STOMP performs better in Scenes 1 and 3, sampling-based planners generally exhibit larger curvature variations in the single-shot setting. The paired probability of superiority indicates EARFEO achieves lower single-shot curvature than the baselines in 87% of the matched cases. In the best-of-N_b setting, RRT-Connect, PRM, STOMP, and P-RRT* outperform EARFEO in several scenes, especially P-RRT* in Scenes 1–4. Overall, EARFEO provides stable and competitive curvature in a single-shot run, whereas stochastic planners can achieve smoother paths through multiple executions.

For the single-shot setting, EARFEO attains success rates of 72%–92% across the six scenes, generally outperforming CHOMP and STOMP and remaining competitive with sampling-based planners in Scenes 1–4, as shown in the fourth column of Fig. 7. In Scenes 5 and 6, however, RRT-Connect and PRM achieve higher success rates, reaching 92%–100%. Under best-of-N_b, stochastic planners improve substantially. RRT-Connect and PRM reach 98%–100% in most scenes, and P-RRT increases from 10% to 98% in Scene 5 and from 10% to 87% in Scene 6. Thus, repeated stochastic sampling improves robustness, but at the cost

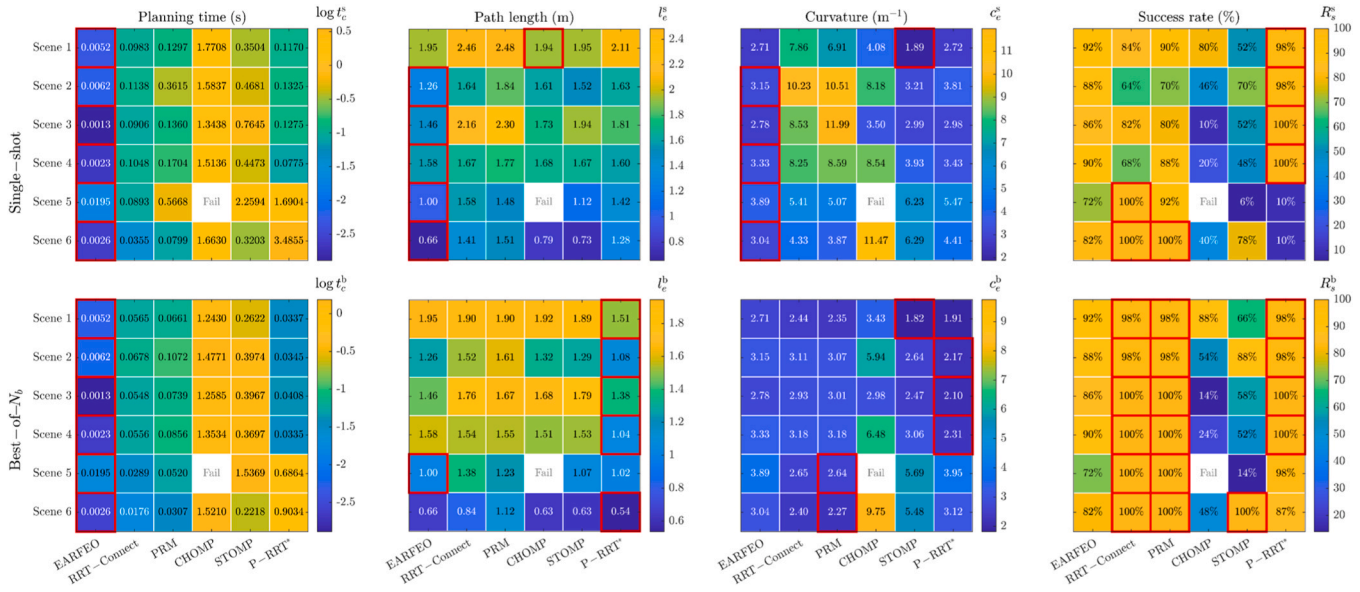

Fig. 7. Scene-wise comparison of planning time, path length, curvature, and success rate under the single-shot and best-of-N_b settings. Cell values denote the original metric values, while planning-time colors are log-scaled for readability. Red boxes mark the best method in each scene.

Table 3
Time to first successful path generation t_c^t (TTFS) with mean value and standard deviation of multiple sets of start-goal pairs for different path planning methods under 6 scenes.

TTFS (s)	EARFEO		RRT-Connect		PRM		CHOMP		STOMP		P-RRT*	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Scene 1	0.0052	0.0040	0.1172	0.0402	0.1485	0.0570	3.5145	7.4400	3.1946	9.9092	0.1170	0.1211
Scene 2	0.0062	0.0039	0.1980	0.1542	0.6902	1.2881	3.0162	4.1803	9.4000	19.3309	0.1325	0.2005
Scene 3	0.0013	0.0002	0.1302	0.0778	0.1785	0.0809	11.4555	36.3422	13.2083	28.1644	0.1275	0.1668
Scene 4	0.0023	0.0009	0.1442	0.0647	0.2073	0.1059	1.7146	0.7436	11.4560	35.9523	0.0775	0.0328
Scene 5	0.0195	0.0049	0.0899	0.0869	0.9698	1.8196	Fail	Fail	172.0900	166.0310	81.2802	86.3913
Scene 6	0.0026	0.0009	0.0361	0.0099	0.0805	0.0345	10.6213	37.0993	1.4720	3.9399	62.9064	88.9493

of additional execution time, as reflected by the TTFS results in Table 3. To evaluate the balance between computational efficiency and robustness, Table 4 reports the ratio between the time to first successful path generation (TTFS) t_c^t and the best-of-N_b success rate R_s^b. A smaller value indicates faster first-solution generation with higher success probability. As shown in Table 4, EARFEO achieves the smallest trade-off values across all six scenes. The values of EARFEO are on the order of 10^{-5} to 10^{-4} while those of the baseline methods are mostly on the order of 10^{-3} to 10^{-1}, and even reach 10^{1} in difficult cases such as STOMP in Scene 5.

Table 4
Trade-off between time to first successful path generation t_c^t and best-of-N_b success rate R_s^b of multiple sets of start-goal pairs for different path planning methods under 6 scenes.

t_c^t/R_s^b	EARFEO	RRT-Connect	PRM	CHOMP	STOMP	P-RRT*
Scene 1	5.7×10^{-5}	1.2×10^{-3}	1.5×10^{-3}	4.0×10^{-2}	4.8×10^{-2}	1.2×10^{-3}
Scene 2	7.0×10^{-5}	2.0×10^{-3}	7.0×10^{-3}	5.6×10^{-2}	1.1×10^{-1}	1.4×10^{-3}
Scene 3	1.5×10^{-5}	1.3×10^{-3}	1.8×10^{-3}	8.2×10^{-2}	2.3×10^{-1}	1.3×10^{-3}
Scene 4	2.6×10^{-5}	1.4×10^{-3}	2.1×10^{-3}	7.1×10^{-2}	2.2×10^{-1}	7.8×10^{-4}
Scene 5	2.7×10^{-4}	9.0×10^{-4}	9.7×10^{-3}	Fail	1.2×10^{1}	8.3×10^{-1}
Scene 6	3.2×10^{-5}	3.6×10^{-4}	8.1×10^{-4}	2.2×10^{-1}	1.5×10^{-2}	7.2×10^{-1}

Overall, EARFEO provides the strongest computational efficiency and competitive single-shot path quality, making it suitable for applications requiring rapid path generation.

After presenting the comparative performance of all algorithms across the six scenarios, the mechanism of EARFEO is further illustrated using two representative scenarios: one involving obstacle clustering with virtual obstacle insertion, and the other involving obstacle clustering without virtual obstacles, as shown in Fig. 8(a) and (b). The core idea of EARFEO lies in iteratively pushing the path away from obstacles with repulsive forces through a series of obstacle expansion steps following obstacle clustering and, in some cases, virtual obstacle insertion. To alleviate the risk of local minima, virtual obstacles (gray cylinders) are inserted within the gaps between green clustered obstacles, as shown in Fig. 8(a). The colored arrows along the path illustrate these repulsive force directions, while color and length encode magnitude. The path is progressively pushed away from the narrow passage and escapes the local minima. In Fig. 8(b), no virtual obstacles are inserted, as the obstacles are connected without gaps. EARFEO incrementally adjusts the path across multiple steps. In the refinement phase, the path is adjusted until the repulsive forces at all waypoints converge to ϵ.

While Fig. 8(a) and (b) illustrate the general path-deformation mechanism of EARFEO, direct deformation based on the original clustered obstacles may still fail in dense obstacle clusters with approximately balanced repulsive fields. Therefore, the box-envelope strategy is further introduced to handle this representative failure mode. Fig. 8(c) presents a representative case in Scene 1. Without the box-envelope representation, the initial path traverses a dense obstacle cluster. During path deformation, the intermediate waypoints located inside or near

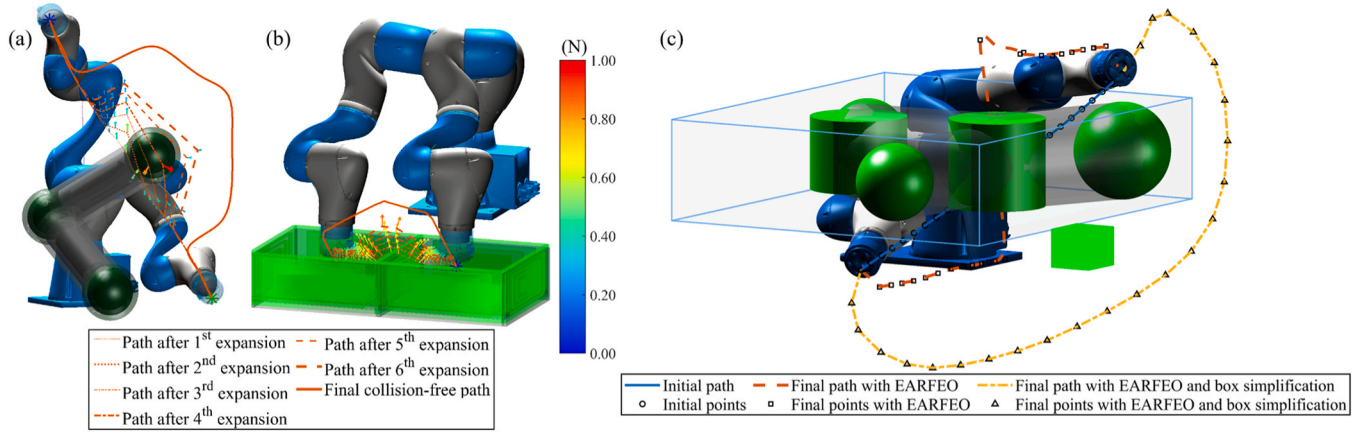

Fig. 8. Path deformation and box-envelope-guided obstacle avoidance. (a) Path expansion with virtual obstacle insertion. (b) Path expansion without virtual obstacle insertion. (c) Box-envelope representation guiding EARFEO around dense obstacle clusters to reduce local-minimum-induced failures.

the cluster are affected by the repulsive fields of multiple cylindrical and spherical obstacles. When the lateral repulsive components are nearly balanced, the waypoints may be displaced mainly in the vertical direction, producing disconnected collision-free segments and an infeasible path. This failure mechanism is associated with the local-minimum limitation of artificial-potential-field-based planning methods [44]. To alleviate this problem, each obstacle cluster intersected by the initial path is approximated by the box envelope before path deformation. The path is encouraged to bypass the obstacle cluster as an integrated region rather than evolving through its internal narrow regions. As shown in Fig. 8(c), EARFEO without the box-envelope representation is trapped by the obstacle cluster, whereas EARFEO with the cluster box envelope generates a collision-free path around the cluster. The cluster box-envelope representation is hard to eliminate all local minima, but addresses an important source of failure in direct EARFEO, and improves the practical success rate while maintaining the advantage of rapid path generation. Therefore, EARFEO is suitable for time-critical industrial scenarios where rapid planning is preferred.

To evaluate the robustness of EARFEO to the repulsive-force-related parameters, a one-factor-at-a-time sensitivity analysis is conducted [47]. Four parameters are examined: the repulsive-force coefficient η, the force saturation bound $F_{\max}$, the safety distance d_0, and the number of obstacle expansion iterations K. In each test, one parameter is varied while the others are fixed at their default values. For each parameter level in each of the six scenarios, EARFEO is evaluated using the corresponding 50 start-goal planning tasks. The 50 outcomes are grouped into ten 5-task blocks to obtain block-level success-rate samples. One-way ANOVA is then applied to these samples to assess whether parameter-level differences are significant. Across the six scenarios and four examined parameters, all ANOVA tests produce p-values greater than 0.05. The minimum p-value is $p = 0.834$, obtained for the variation of safety distance in Scene 4. These results indicate that no statistically significant parameter-induced differences are detected in the tested environments. As a representative cluttered 3D workspace, Scene 3 further confirms this trend. The success rate ranges from 84% to 90% for η, from 82% to 88% for $F_{\max}$, from 84% to 88% for d_0, and from 82% to 88% for K. The corresponding ANOVA results are $p = 0.902$, $p = 0.906$, $p = 0.957$, and $p = 0.918$, respectively. With the other five scenes, these results indicate that EARFEO maintains stable performance under the tested parameter ranges and environmental complexities.

To further investigate the computational characteristics of EARFEO, the number of discretized path points N_d is varied across all test scenes. The average success rate changes only slightly from 83.67% at $N_d = 15$ to 85.67% at $N_d = 35$, while remaining 85.00% for $N_d = 20$, 25, and 30. This marginal improvement indicates that denser path discretization provides limited benefit to planning success within the tested range. By contrast, the average planning time increases monotonically from 2.57×10^{-3} to 7.09×10^{-3} s as N_d increases from 15 to 35. This trend is mainly due to the additional distance queries, force calculations, path updates, and collision checks required by larger N_d. Therefore, $N_d = 25$ is adopted in the main experiments as a practical trade-off, achieving 85.00% success with an average planning time of 4.64×10^{-3} s. In addition, the runtime of EARFEO is decomposed into several algorithmic components. Distance query is the most time-consuming component, accounting for 69.5% of the total runtime. Force calculation is the second-largest component, accounting for 18.6%, followed by path smoothing and collision checking, accounting for 9.2%. The remaining components, including obstacle clustering and virtual obstacle insertion, uniform resampling, RDP path simplification, and other operations, each contribute less than 1% to the total runtime. Thus, repeated distance queries and force calculations dominate the computational cost of EARFEO.

3.3. Dynamic industrial environment experiment verifications

3.3.1. Dynamic bricklaying environment experiment

The performance of EARFEO in the dynamic bricklaying environment is evaluated in this subsection. Unlike static simulations, this task involves repeated pick-and-place operations with continuously changing target positions as the corner wall is constructed. Fig. 9 depicts a typical operation cycle. The robot picks up bricks from the stack on the left, navigates through a wall-like obstacle in the center, and places the bricks onto the corner wall structure on the right. The purple dashed line represents the polynomial-based path fitted from the key path points generated by EARFEO. EARFEO is compared with RRT-Connect, PRM, CHOMP, STOMP, and P-RRT* over 36 movements. The evaluated metrics include planning time, path length, and path curvature statistics. As shown in Table 5, EARFEO achieves the shortest average planning time among the non-reuse planners, with 0.0094 s on average and 0.3390 s in total over all 36 movements. Compared with RRT-Connect, the fastest baseline, EARFEO reduces both values by approximately 83%. EARFEO is also substantially faster than PRM, CHOMP, STOMP, and P-RRT*, indicating its suitability for time-sensitive structured manipulation tasks. In terms of path length, EARFEO achieves a mean path length of 1.2407 m and a total path length of 44.6651 m, shorter than those of RRT-Connect, PRM, and P-RRT*. This indicates that EARFEO provides a favorable balance between efficiency and path compactness.

To further enhance computational efficiency in repetitive motion planning tasks, a path reuse strategy is introduced, leveraging sequential similarity in the wall-building process. The path reuse strategy is mainly intended for structured dynamic scenarios with repetitive task sequences, progressive environmental changes, and similar adjacent start

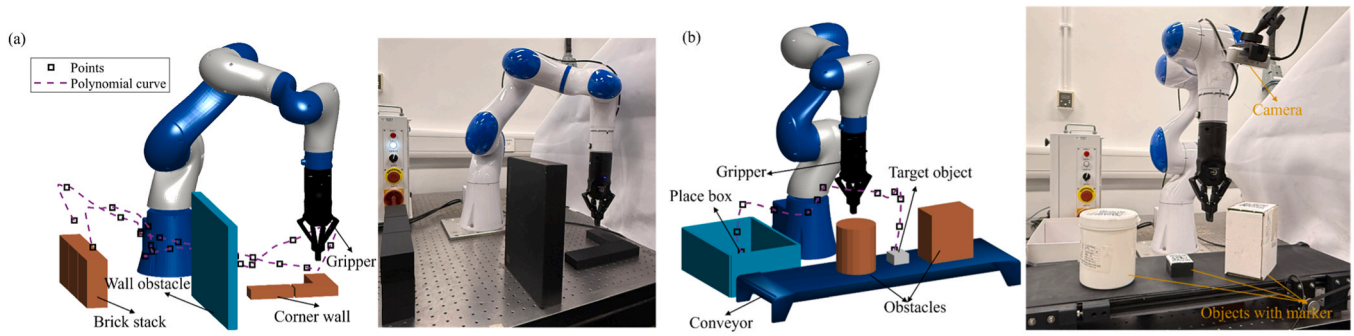

Fig. 9. Simulation and Experiment verification under (a) repetitive dynamic corner wall bricklaying tasks and (b) dynamic conveyor-belt-based robotic pick-and-place tasks with EARFEO path planning.

Table 5
Comparison of the proposed EARFEO (with and without path reuse) over the baseline methods on the 36 sets of corner-wall bricklaying tasks.

Methods	Mean planning time with SD (s)	Total time (s)	Mean path length with SD (m)	Total path length (m)	Mean path curvature with SD (1/m)
EARFEO	0.0094±0.0035	0.3390	1.2407±0.1497	44.6651	6.1264±1.1643
EARFEO-reuse	0.0051±0.0019	0.1842	1.3166±0.1335	47.3978	7.2692±1.2692
RRT-Connect	0.0567±0.0119	2.0425	1.6078±0.3102	57.8802	6.2767±1.4105
PRM	0.0755±0.0295	2.7162	1.6240±4.1241	58.4624	7.1384±2.4905
CHOMP	1.5034±0.1131	54.1222	1.3216±0.1240	47.5776	6.5058±1.2959
STOMP	0.4066±0.1646	14.6394	1.2078±0.1397	43.4811	5.6574±1.9964
P-RRT*	0.1895±0.3272	6.8221	1.5571±0.4242	56.0556	6.1840±0.5844

or goal configurations. Reusing part of the previously generated path can further reduce planning time while preserving feasibility. The transition is achieved by truncating the reused path near the end and appending a newly interpolated segment toward the updated target. To ensure structural consistency, the total number of waypoints is preserved during the update. The performance of the reuse-enhanced EARFEO is also summarized in Table 5. With path reuse enabled, EARFEO-reuse achieves the lowest planning time among all evaluated methods, with 0.0051 s on average and 0.1842 s in total. Compared with the non-reuse version of EARFEO, the reuse strategy further reduces the planning time by 46%. The slight increase in path length observed with the reuse strategy is an inherent trade-off. Specifically, the mean path length increases from 1.2407 m to 1.3166 m, corresponding to an increase of approximately 6%. Nevertheless, EARFEO-reuse still achieves a shorter mean path length than RRT-Connect, PRM, and P-RRT*. For path curvature, EARFEO achieves a mean curvature of 6.1264 m^{-1}, lower than RRT-Connect, PRM, and CHOMP, but slightly higher than STOMP and P-RRT*. With reuse, the mean curvature increases to 7.2692 m^{-1}, indicating a slight smoothness reduction caused by partial path inheritance and local terminal-segment regeneration. The reuse strategy improves computational efficiency significantly, while introducing moderate increases in path length and curvature.

To support the computation-efficient claim of the path reuse strategy, a brief computational complexity analysis is provided based on the standard asymptotic time-complexity analysis using Big-O notation. Let $N_{\mathscr{O}}$ denote the number of obstacles, K denote the number of obstacle-expansion iterations, L_r denote the number of path-refinement iterations after obstacle expansion, and N_r denote the number of repeated motion cycles. Since the dominant computational cost of EARFEO comes from evaluating the repulsive effects and collision-related distances between waypoints and obstacles, the cost of one complete planning process is defined as $C_s = O((K+L_r)N_{\mathscr{O}}N_d)$. If all N_r motions are planned independently, the complexity is $C_t = O(N_r(K+L_r)N_{\mathscr{O}}N_d)$. With the path reuse strategy, full EARFEO planning is only performed for the first motion, while the subsequent motions require local adjustment. Let βN_d ($0 < \beta < 1$ denote the number of waypoints requiring local adjustment in each reused motion. The total complexity is $C_p = O((N_r-1)(K+L_r)N_{\mathscr{O}}\beta N_d + (K+L_r)N_{\mathscr{O}}N_d)$. Therefore, the approximate speedup over independent replanning is

$$S = C_t \Big/ C_p = \frac{N_r}{1+(N_r-1)\beta}. \tag{54}$$

As indicated by (54), the benefit of path reuse increases with the number of repeated motion cycles N_r. For $0 < \beta < 1$, the speedup increases with N_r, and asymptotically approaches $1/\beta$ as $N_r \to \infty$. Thus, the path reuse strategy is beneficial for repetitive tasks with small locally affected path segments. These conditions are consistent with the bricklaying scenario considered in this work, where consecutive manipulation motions are highly structured and repetitive. Under these conditions, path reuse reduces redundant computation while preserving local collision-free feasibility.

3.3.2. *Dynamic conveyor-belt-based robotic pick-and-place environment experiment*

The performance of EARFEO in the conveyor-belt-based robotic pick-and-place experiment is evaluated in this subsection. Unlike the structured bricklaying task in Section 3.3.1, this experiment evaluates EARFEO under event-triggered planning in a less structured dynamic environment. As the conveyor moves, object distributions, obstacle layouts, target poses, and start-goal configurations change continuously. Once a target object is detected, the conveyor stops, and EARFEO replans the path using the latest perceived scene information rather than reusing a previous path. Therefore, each pick-and-place operation is treated as an independent dynamic replanning instance. Fig. 9(b) presents a representative planning result. The purple dashed line denotes the polynomial-smoothed path obtained from the key waypoints generated by EARFEO. The planned path guides the manipulator toward the detected target object while avoiding surrounding non-target objects and workspace constraints, demonstrating the feasibility of EARFEO in dynamic operations.

In this experiment, EARFEO is benchmarked against RRT-Connect, PRM, CHOMP, STOMP, and P-RRT* using planning time, path length, and path curvature statistics across multiple pick-and-place movements. As shown in Table 6, EARFEO achieves the shortest average planning time among all successful planners, reducing it by approximately 93% compared with RRT-Connect, the fastest baseline. EARFEO also

Table 6
Comparison of the proposed EARFEO with other baseline methods over multiple pick-and-place movements.

Methods	Mean planning time with SD (s)	Mean path length with SD (m)	Mean path curvature with SD (1/m)
EARFEO	0.0046±0.0010	1.3997±0.0515	5.6956±0.8985
RRT-Connect	0.0642±0.0029	2.3177±1.3938	7.3690±1.0003
PRM	0.1423±0.0591	2.1705±1.0378	6.5613±1.0867
CHOMP	Fail	Fail	Fail
STOMP	2.0465±1.4088	1.3208±0.0841	6.4756±1.7731
P-RRT*	0.8322±1.0822	1.5876±0.6854	5.5674±1.3287

generates shorter paths than RRT-Connect, PRM, and P-RRT*. Although STOMP produces the shortest mean path length, this method requires a longer planning time of 2.0465 s. In terms of path curvature, EARFEO has lower mean curvature than RRT-Connect, PRM, and STOMP, but slightly higher curvature than P-RRT*. CHOMP fails in this experiment. Overall, the results demonstrate that EARFEO provides fast planning in the conveyor-belt-based dynamic environment. EARFEO is suitable for time-critical dynamic robotic tasks requiring rapid replanning and reliable path generation.

4. Conclusion

This paper proposed a computation-efficient path planning method using evolving artificial repulsive force over expanding obstacles (EARFEO) for robotic manipulators. The proposed approach introduced a novel mechanism for mitigating local minima by clustering predefined obstacles using an undirected graph, strategically inserting virtual obstacles within each cluster, and utilizing the box-envelope strategy. The initial path was iteratively pushed away from the expanding obstacles with the artificial repulsive force. Once the obstacle expansion was finished, the path was continuously refined to be pushed away from the obstacles until the artificial repulsive force along the whole path fell below the threshold. To further enhance path quality, RDP path simplification and polynomial smoothing were applied to the refined path. The effectiveness of EARFEO has been validated through extensive static and dynamic simulations and experiments, demonstrating that EARFEO achieves superior performance in terms of planning time. These results highlight the potential of EARFEO for high-productivity tasks.

CRediT authorship contribution statement

Yi Zhou: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization, Writing – review & editing. **Bowen Huang:** Validation, Software, Methodology, Formal analysis. **Yim Ho Cheung:** Visualization, Validation, Software, Formal analysis. **Pengfei Ye:** Visualization, Formal analysis. **Molong Duan:** Writing – original draft, Supervision, Resources, Project administration, Methodology, Funding acquisition.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used ChatGPT and Grammarly AI in order to perform grammar checks. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors gratefully acknowledge financial support from the Research Grants Council (RGC) of the Hong Kong Special Administrative Region (HKSAR) Government through the RGC/ECS project (No. 26215824), the Frontier Technology Research for Joint Institutes with Industry scheme (Grant OKT26EG04), Guangdong Provincial Department of Science and Technology (2025A0505000030), and HKUST-BYD Embodied AI Joint Lab Research Project (BYD26EG02). The authors also thank the funding agencies and collaborating researchers for their assistance.

References

[1] Yu YH, Zhang YT. Collision avoidance and path planning for industrial manipulator using slice-based heuristic fast marching tree. Robot Comput Integr Manuf 2022; 75:102289. https://doi.org/10.1016/j.rcim.2021.102289.
[2] Shao S, Peng Y, He C, Du Y. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization. ISA Trans 2020;97: 415–30. https://doi.org/10.1016/j.isatra.2019.08.018.
[3] Tamizi MG, Honari H, Nozdryn-Plotnicki A, Najjaran H. End-to-end deep learning-based framework for path planning and collision checking: bin-picking application. Robotica 2024;42:1094–112. https://doi.org/10.1017/S0263574724000109.
[4] Yang Y, Zhou Y, Duan M. Contact-force-based closed-loop control of shell structure additive manufacturing with continuous-fiber-reinforced polymer composites. J Mater Process Technol 2024;331:118501. https://doi.org/10.1016/j.jmatprotec.2024.118501.
[5] Zhou Y, Huang B, Dong B, Wen Y, Duan M. Dynamic robotic bricklaying force-position control considering mortar dynamics for enhanced consistency. Autom Constr 2025;174:106090. https://doi.org/10.1016/j.autcon.2025.106090.
[6] Zhou Y, Duan M. Vibration Compensation of an Extendable Variable-Stiffness Boom-Lift-Mounted Robot. IEEE/ASME Trans Mechatron 2024;29:2812–20. https://doi.org/10.1109/TMECH.2024.3402054.
[7] Liu Z, Liu Q, Xu W, Wang L, Zhou Z. Robot learning towards smart robotic manufacturing: A review. Robot Comput Integr Manuf 2022;77:102360. https://doi.org/10.1016/j.rcim.2022.102360.
[8] Gasparetto A, Boscariol P, Lanzutti A, Vidoni R. Path planning and trajectory planning algorithms: A general overview. Motion Oper Plan Robot Syst Backgr Pract Approaches 2015:3–27. https://doi.org/10.1007/978-3-319-14705-5_1.
[9] Tao Y, Du J. Time-optimal global path planning and collision-avoidance local path planning for USVs in traffic separation scheme-implemented coastal waters. ISA Trans 2025;165:280–94. https://doi.org/10.1016/j.isatra.2025.06.030.
[10] Cohen BJ, Chitta S, Likhachev M. Search-based planning for manipulation with motion primitives. IEEE International Conference on Robotics and Automation; 2010. p. 2902–8. https://doi.org/10.1109/ROBOT.2010.5509685.
[11] Dijkstra EW. A note on two problems in connexion with graphs. Numer Math (Heidelb) 1959;1:269–71. https://doi.org/10.1007/BF01386390.
[12] Hart PE, Nilsson NJ, Raphael B. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern 1968;4:100–7.
[13] Pohl I. The avoidance of (relative) catastrophe, heuristic competence, genuine dynamic weighting and computational issues in heuristic problem solving. Proceedings of the 3rd international joint conference on Artificial intelligence. 1973. p. 12–7.
[14] Stentz A. Optimal and efficient path planning for unknown and dynamic environments. Intell Unmanned Ground Veh 1997:203–20. https://doi.org/10.1007/978-1-4615-6325-9_11.
[15] Zhang Z, Jiang J, Wu J, Zhu X. Efficient and optimal penetration path planning for stealth unmanned aerial vehicle using minimal radar cross-section tactics and modified A-Star algorithm. ISA Trans 2023;134:42–57. https://doi.org/10.1016/j.isatra.2022.07.032.
[16] Cohen B, Sucan IA, Chitta S. A generic infrastructure for benchmarking motion planners. IEEE International Conference on Intelligent Robots and Systems; 2012. p. 589–95. https://doi.org/10.1109/IROS.2012.6386228.
[17] Liu J, Yap HJ, Khairuddin ASM. Review on Motion Planning of Robotic Manipulator in Dynamic Environments. J Sens 2024:2024. https://doi.org/10.1155/2024/5969512.
[18] Kavraki LE, Svestka P, Latombe JC, Overmars M. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans Robot Autom 1996; 12:566–80. https://doi.org/10.1109/70.508439.

[19] LaValle SM. Rapidly-Exploring Random Trees: A New Tool for Path Planning. Res Report 1998.
[20] Kuffner JJ, La Valle SM. RRT-connect: an efficient approach to single-query path planning, 2. IEEE International Conference on Robotics and Automation; 2000. p. 995–1001. https://doi.org/10.1109/robot.2000.844730.
[21] Karaman S, Frazzoli E. Incremental sampling-based algorithms for optimal motion planning. Robotics Science Systems 2011;6:267–74. https://doi.org/10.15607/rss.2010.vi.034.
[22] Gammell JD, Srinivasa SS, Barfoot TD. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. IEEE International Conference on Intelligent Robots and Systems; 2014. p. 2997–3004. https://doi.org/10.1109/IROS.2014.6942976.
[23] Feng B, Jiang X, Li B, Zhou Q, Bi Y. An adaptive multi-RRT approach for robot motion planning. Expert Syst Appl 2024;252:124281. https://doi.org/10.1016/j.eswa.2024.124281.
[24] Ganesan S, Ramalingam B, Mohan RE. A hybrid sampling-based RRT* path planning algorithm for autonomous mobile robot navigation. Expert Syst Appl 2024:258. https://doi.org/10.1016/j.eswa.2024.125206.
[25] Elbanhawi M, Simic M. Sampling-based robot motion planning: A review. IEEE Access 2014;2:56–77. https://doi.org/10.1109/ACCESS.2014.2302442.
[26] Ratliff N, Zucker M, Andrew Bagnell J, Srinivasa S. CHOMP: Gradient optimization techniques for efficient motion planning. IEEE International Conference on Robotics and Automation; 2009. p. 489–94. https://doi.org/10.1109/ROBOT.2009.5152817.
[27] Kalakrishnan M, Chitta S, Theodorou E, Pastor P, Schaal S. STOMP: Stochastic trajectory optimization for motion planning. Proc IEEE Int Conf Robot Autom; 2011. p. 4569–74. https://doi.org/10.1109/ICRA.2011.5980280.
[28] Schulman J, Duan Y, Ho J, Lee A, Awwal I, Bradlow H, et al. Motion planning with sequential convex optimization and convex collision checking. Int J Robot Res 2014;33:1251–70. https://doi.org/10.1177/0278364914528132.
[29] Dong J, Mukadam M, Dellaert F, Boots B. Motion planning as probabilistic inference using Gaussian processes and factor graphs. Robotics Science Systems 2016;12. https://doi.org/10.15607/rss.2016.xii.001.
[30] Mukadam M, Dong J, Yan X, Dellaert F, Boots B. Continuous-time Gaussian process motion planning via probabilistic inference. Int J Robot Res 2018;37:1319–40. https://doi.org/10.1177/0278364918790369.
[31] Escande A, Mansard N, Wieber PB. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. Int J Robot Res 2014;33:1006–28. https://doi.org/10.1177/0278364914521306.
[32] Nie J, Wang Y, Mo Y, Miao Z, Jiang Y, Zhong H, et al. An HQP-Based Obstacle Avoidance Control Scheme for Redundant Mobile Manipulators Under Multiple Constraints. IEEE Trans Ind Electron 2023;70:6004–16. https://doi.org/10.1109/TIE.2022.3196390.
[33] Wang G, Ma H, Wang H, Ding P, Bai H, Xu W, et al. Reactive mobile manipulation based on dynamic dual-trajectory tracking. Rob Auton Syst 2024;172:104589. https://doi.org/10.1016/j.robot.2023.104589.
[34] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. Int J Rob Res 1986;5:90–8. https://doi.org/10.1177/027836498600500106.
[35] Bounini F, Gingras D, Pollart H, Gruyer D. Modified artificial potential field method for online path planning applications. IEEE Intelligent Vehicles Symposium; 2017. p. 180–5. https://doi.org/10.1109/IVS.2017.7995717.
[36] Wu Z, Dai J, Jiang B, Karimi HR. Robot path planning based on artificial potential field with deterministic annealing. ISA Trans 2023;138:74–87. https://doi.org/10.1016/j.isatra.2023.02.018.
[37] Qureshi AH, Ayaz Y. Potential functions based sampling heuristic for optimal path planning. Auton Robots 2016;40:1079–93. https://doi.org/10.1007/s10514-015-9518-0.
[38] Li Y, Wei W, Gao Y, Wang D, Fan Z. PQ-RRT*: An improved path planning algorithm for mobile robots. Expert Syst Appl 2020;152:113425. https://doi.org/10.1016/j.eswa.2020.113425.
[39] Feng Z, Zhou L, Qi J, Hong S. DBVS-APF-RRT*: A global path planning algorithm with ultra-high speed generation of initial paths and high optimal path quality. Expert Syst Appl 2024;249:123571. https://doi.org/10.1016/j.eswa.2024.123571.
[40] Ding ZJ, Meng XJ, Zhang LX, Guo YBin, Chen MQ, Zhen SH. Variable Sampling Area RRT Algorithm Based on Limiting Joint Angle for Robotic Arm Path Planning. IEEE Access 2025;13:30665–77. https://doi.org/10.1109/ACCESS.2025.3541645.
[41] Ericson C. Real-Time Collision Detection. CRC Press; 2005.
[42] Peucker T, Douglas DH. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature. Cartogr Int J Geogr Inf Geovisualization 1973;10:112–22.
[43] Aissaoui TD, Dahmouche R. Zoned Artificial Repulsion: Path Planning Through Local Minima for Multiple-Robot Dexterous Micromanipulation. IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE; 2025. p. 2014–21. https://doi.org/10.1109/iros60139.2025.11247568.
[44] Koren Y, Borenstein J. Potential field methods and their inherent limitations for mobile robot navigation, 2. IEEE International Conference on Robotics and Automation; 1991. p. 1398–404. https://doi.org/10.1109/robot.1991.131810.
[45] Nocedal J, Wright SJ. Numerical optimization. New York: Springer; 2006. https://doi.org/10.1007/978-0-387-40065-5.
[46] Chamzas C, Quintero-Pena C, Kingston Z, Orthey A, Rakita D, Gleicher M, et al. MotionBenchMaker: A Tool to Generate and Benchmark Motion Planning Datasets, 7. IEEE Robot Autom Lett; 2022. p. 882–9. https://doi.org/10.1109/LRA.2021.3133603.
[47] Saltelli A, Ratto M, Andres T, Campolongo F, Cariboni J, Gatelli D, et al. Global Sensitivity Analysis. The Primer. John Wiley & Sons; 2008. https://doi.org/10.1002/9780470725184.