Path Optimization Using an Improved APF-RRT* Algorithm

Yongyang Zheng [1,2], Monsak Pimsarn [3], Varesa Chuwattanakul [3,*], Suriya Chokphoemphun [4], and Smith Eiamsa-ard [1]

[1] School of Industrial and Engineering Technology, Mahanakorn University of Technology, Bangkok, Thailand
[2] Sino-German College of Intelligent Manufacturing, Shenzhen City Polytechnic, Shenzhen, China
[3] School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
[4] Department of Mechanical and Manufacturing Engineering, Faculty of Science and Engineering, Kasetsart University Chalermphrakiat Sakonnakhon Province Campus, Sakonnakhon, Thailand
Email: 6717740005@mut.ac.th (Y.Z.); monsak_pi@kmitl.ac.th (M.P.); varesa.ch@kmitl.ac.th (V.C.); suri-ya.cho@ku.ac.th (S.C.), smith@mut.ac.th (S.E.)
*Corresponding author

***Abstract*—Path planning remains a critical research area in mobile robotics, yet current approaches often suffer from suboptimal path quality, limited sampling efficiency, and inadequate adaptability across diverse operational scenarios. To address these issues, this paper proposes an improved algorithm combining Artificial Potential Field (APF) and Restricted Path Time (RRT*) approaches. This algorithm employs an optimization model that combines dynamic sampling with potential field guidance, constructing a two-stage dynamic sampling mechanism. During sampling, candidate nodes with Gaussian noise are generated along the resultant force direction. Finally, path cost comparison and parent node reselection are performed within the dynamic optimization radius to ensure asymptotic optimality of the path. Experimental results show that in complex maps, path length is reduced by 33.41% and 26.64%, respectively, and planning time is reduced by 21.36% and 86.32%, respectively; in narrow passages, path length is reduced by 49.6% and 49.8%, respectively. The results confirm the effectiveness of the two-stage dynamic sampling mechanism, which not only preserves the probabilistic completeness of the RRT* algorithm but also adaptively adjusts the sampling strategy, improving both planning length and time.**

***Keywords*—artificial potential field, Restricted Path Time (RRT*) algorithm, path planning, dynamic sampling**

I. Introduction

With the continuous development of artificial intelligence, mobile robots have been widely used in various industries, the military, and service sector. Path planning is a core component of autonomous mobile robot navigation [1]. The key to path planning is to efficiently avoid obstacles and find a better path. Therefore, related algorithms have always been a research hotspot. Existing methods can be roughly divided into three categories: (1) geometric model algorithms, such as Dijkstra [2], A* [3], Probabilistic Roadmap (PRM) [4], Rapidly-exploring Random Tree (RRT) [5]; (2) swarm intelligence algorithms, such as ant colonies [6], genetic algorithms [7], Genetic Algorithm-Particle Swarm Optimization (GA-PSO) [8]; (3) self-learning algorithms, such as Q-learning [9] and deep learning [10]. To this end, researchers have proposed improved methods such as Restricted Path Time (RRT*) [11], Informed RRT* [12] and Informed RRT*-Connect [13], which have made progress in terms of speed and path quality. Among them, the RRT series based on random sampling has attracted widespread attention because of its ability to adapt to complex environments. The PRM algorithm in geometric models performs well in global path searches in high-dimensional spaces, but lacks adaptability in dynamic environments. The PSO algorithm [14] in swarm intelligence offers advantages in global optimization and complex environment search, but its convergence speed and stability are inferior to sampling-based methods.

Despite significant progress in the field, contemporary algorithms continue to face two fundamental challenges: (1) achieving an optimal balance between solution quality and computational efficiency; and (2) maintaining convergence stability in geometrically constrained environments such as narrow passages or dynamically evolving scenarios. In response to these issues, this article proposes an improved RRT* algorithm (APF-RRT*) that combines artificial potential fields to improve convergence efficiency and path optimality.

The main innovations and contributions of this paper are as follows:

(1) Introducing Artificial Potential Field (APF) guidance during the sampling process to make the search more directed and efficient;
(2) Constructing a dynamic sampling and rewiring optimization mechanism to improve path quality while maintaining asymptotic optimality;

Manuscript received August 5, 2025; revised September 15, 2025; accepted October 20, 2025; published February 9, 2026.

(3) Conducting multi-environment simulation experiments to verify the superior performance of the proposed APF-RRT* algorithm in both convergence speed and path smoothness.

II. Related Work

In the development of search path technologies, LaValle [15] first proposed the RRT algorithm. It is based on the sampling path planning algorithm and satisfies the need to efficiently find feasible paths from the starting point to the target point in a high-dimensional and complex constraint space. The main emphasis is on feasibility, the disadvantages of poor search path quality and long search time. Kuffner *et al.* [16] proposed the RRT-Connect algorithm to solve the disadvantages of long search time. It introduces bidirectional fast expansion technology and builds a search tree simultaneously from the start point and the end point, thereby significantly reducing the search time. It only shortens the search time, cannot guarantee the quality of the search path, and lacks engineering application significance. Karaman [11] and LaValle [15] proposed the RRT algorithm to improve the path quality, which introduces asymptotic optimality through rewiring and maintains probabilistic completeness. However, a large number of samples is required to obtain a high-quality path, resulting in slow convergence speed and large memory consumption. Jeong *et al.* [17] proposed the Quick-RRT* algorithm to address the disadvantages of RRT, which solves the problem of slow convergence speed by introducing a dynamic step size method. However, the ability of Quick-RRT* to maintain parameter optimality is weakened, and rapid extension may skip potential low-cost neighborhoods. Jeong *et al.* [17] have also adopted an interdisciplinary combination method to obtain a better path solution. By combining force field modeling from physics with sampling-based motion, and integrating local strategies with global sampling, an artificial potential field and sampling algorithm was developed. However, local minima may occur. In summary, RRT uses a one-way tree search, resulting in moderate convergence speed. While simple to implement, it suffers from poor path quality and cannot guarantee optimality. The RRT-Connect algorithm improves search efficiency through a two-way tree search, achieving faster convergence, but its path quality remains poor. In contrast, Quick-RRT* employs a dynamic step size and a target-oriented guidance strategy, improving convergence speed, path quality, and optimality to some extent, but overall, there is still room for improvement.

Currently, mainstream improved RRT algorithms can be broadly categorized into several core ideas: First, represented by RRT*, these algorithms introduce asymptotic optimality guarantees through reconnection mechanisms, continuously improving path quality while ensuring feasible solutions. Second, represented by RRT-Connect and Informed RRT*, these algorithms improve search efficiency in high-dimensional spaces through bidirectional expansion and ellipsoidal domain sampling. Finally, there are algorithms emphasizing deep combinations of system dynamics and constraints, such as Linear Quadratic Regulator Rapidly-exploring Random Tree (LQR-RRT), which consider dynamic constraints and control costs during expansion to generate more perfect paths. Although these improvements have achieved good performance across different environments and runs, they still have shortcomings in specific situations: Most algorithms fail in narrow channels due to strategy failures and severe "blind" search phenomena, making it difficult to guarantee consistently high-quality solutions. Additionally, relying on a single strategy may perform well in a specific scenario but fail to meet the needs of multiple scenarios. To address these shortcomings, this paper adopts a hybrid sampling strategy to enable operation in multiple scenarios, compensating for the lack of scenario adaptability in traditional RRT* improved algorithms.

III. RRT* Algorithm Integrated with APF

A. Algorithm Principle

RRT* is an enhanced variant of the RRT algorithm that converges towards an increasingly optimal solution over time. Its core principle is to converge the path cost to the optimal through rewiring (r_{rewire}) and parent node reselection mechanisms. In a newly generated x-node, a circle with a defined radius is used to compare the distances of each point enclosed in the circle to generate the shortest route, as shown in Eq. (1), to calculate the neighborhood radius r_n . The purpose of the rewire function is to detect nodes in existing trees, check the lengths of paths, compare different routes, and compare path costs to obtain the solution that yields the minimum cost of the search tree connection as shown in Eqs. (2) and (3). The RRT* algorithm will find a better path cost as the number of iterations increases, as depicted in Fig. 1.

$$r_n = min\left(\gamma\left(\frac{log(n)}{n}\right)^{\frac{1}{d}}, \eta\right) \tag{1}$$

$$C(x_{new}) + C(x_{new}, x_i) < C(x_i) \tag{2}$$

$$parent(x_i) = x_{new} \tag{3}$$

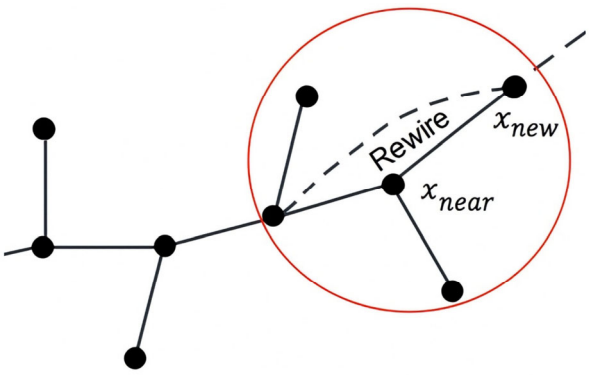

Fig. 1. Schematic diagram of new node expansion of the RRT* algorithm.

B. APF Algorithm Principle

The APF algorithm is an established way to plan a path. Its core idea is to guide a mobile robot to avoid obstacles

and reach a target point by simulating a "potential field" in physics. The algorithm has two potential fields, gravitational $U_{att}(q)$ and repulsive $U_{rep}(q)$ fields. The target point makes the gravitational field, which pulls the robot toward it. The obstacle produces a repulsive field that pushes the robot away. If q is the robot's current position, Eqs. (4) and (5) are the gravitational potential field and the gravitational functions, respectively.

$$U_{att}(q) = \frac{1}{2} k_a \times d^2(q, q_{goal}) \tag{4}$$

$$F_{att} = -\nabla U_{att}(q) = k_a \times (q_{goal} - q) \tag{5}$$

In Eq. (1), k_a represents the gravitational gain coefficient, and $d(q,q_{goal})$ shows the distance from the robot's current position q to the goal in Euclidean space q_{goal}. As the robot searches in space, the function is directly related to the distance between the robot and the obstacle. The function for the repulsive potential field $U_{rep}(q)$ and the repulsive function $F_{rep}(q)$ are shown in Eqs. (6) and (7), where k_r represents the repulsive gain coefficient, $d(q,q_{obs})$ is the distance from the robot to the obstacle, and d_0 represents the repulsive influence range threshold.

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_r (\frac{1}{d(q,q_{obs})} - \frac{1}{d_0})^2 & if\ d(q,q_{obs}) \le d_0 \\ 0 & if\ d(q,q_{obs}) > d_0 \end{cases} \tag{6}$$

$$F_{rep} = k_r \left(\frac{1}{d(q,q_{obs})} - \frac{1}{d_0}\right) \times \frac{1}{d^2(q,q_{obs})} \times \nabla d(q,q_{obs}) \tag{7}$$

The robot's total force is the vector sum of all the forces that push it away and gravity, as shown in Eq. (8).

$$F_{total} = F_{att} + \sum F_{rep} \tag{8}$$

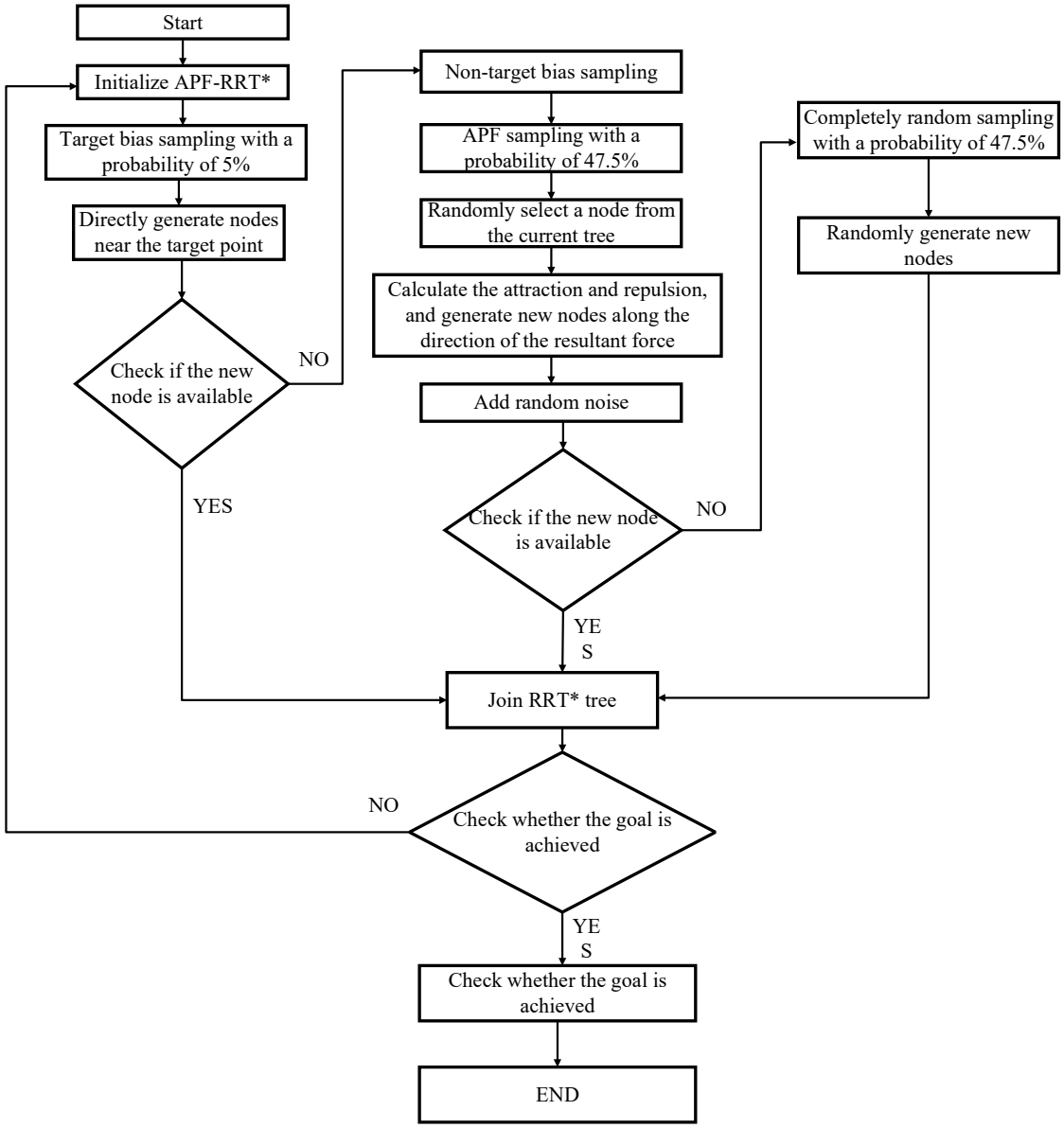

Fig. 2. Dynamic sampling strategy program flow chart.

C. RRT* Algorithm Optimization Integrated with APF

1) Dynamic sampling strategy

In this study, influenced by an artificial potential field and a consequent force F_{total}, the robot will move in the direction of x_{goal}. If the force that pushes the robot away from the barrier is stronger than the force that pulls the robot toward the obstacle, it will not reach the target point. In response to this phenomenon, the current approach adopts a hybrid sampling strategy. First, target bias sampling is directly performed with a probability of 5%, forcing the algorithm to explore in the target direction, accelerating convergence, and avoiding the inefficiency of pure random sampling. Second, a non-target configuration search is performed with a probability of 95%, in which APF-guided sampling and completely random sampling are included in a non-target bias. Finally, APF-guided sampling with a probability of 47.5% is used to randomly select a node from the current tree, calculate the gravitational and repulsive forces it receives, create additional nodes in the direction of the force that results, and add random noise to avoid falling into local minima. If the target bias or APF guidance is not met, random sampling with a probability of 47.5% is performed as shown in Fig. 2. To ensure the algorithm's capability to explore unknown areas and avoid missing feasible paths, these two algorithms adopt a dynamic balance, which not only ensures global exploration but also accelerates convergence to feasible paths. Combining the two algorithms achieves the synergy of "guidance" and "randomness", making APF-RRT* a more robust and efficient path planning algorithm in situations with complex environments, high real-time requirements, and dynamic obstacle scenarios.

2) Collision detection optimization

The efficiency of collision detection in path planning algorithms has an important impact on real-time performance. This paper adopts a hierarchical collision detection framework and an adaptive step size adjustment strategy. Compared with the traditional stepwise interpolation method, the algorithm must point-by-point check collisions between line segments and obstacles. This has high computational complexity. In the collision detection operation, the distance field pre-calculation and the Bresenham-ray method are combined to achieve two-stage detection. In the course detection stage, the distance field is used to quickly determine the safe distance near the endpoint of the line segment. If it is greater than the threshold, the possibility of collision is directly excluded. In the fine detection stage, the Bresenham algorithm is employed to generate accurate grid coordinates for high-risk areas, avoiding the performance loss caused by floating-point operations as shown in Algorithm 1.

Second, in complex environmental conditions, fixed step sizes are prone to computational redundancy or detection omissions. A dynamic step size adjustment method is proposed to address this limitation. A smaller step size is used in areas with dense obstacles to ensure detection accuracy. A larger step size is employed in free space to improve traversal efficiency. The step size parameter is automatically changed based on the density of local obstacles, which strikes a compromise between safety and efficiency, while significantly improving the computational efficiency of the APF-RRT* algorithm.

Algorithm 1: Line Segment Collision Detection

Data: *$node_1 = [y_1, x_1]$: Start point coordinates (y, x).*
$node_2 = [y_2, x_2]$: End point coordinates (y, x).
map: 2D occupancy grid map, where 2 denotes an obstacle.
param: Structure containing parameters, e.g., param.resolution.
Result: *flag: Boolean value; TRUE if collision detected, FALSE otherwise.*

Function is_collision($node_1$, $node_2$, *map, param*)
flag → TRUE *Assume collision until proven otherwise
$dy \leftarrow node_2(1) - node_1(1)$ *y-coordinate difference $dx \leftarrow node_2(2) - node_1(2)$ *x-coordinate difference $\theta \leftarrow$ atan2(*dy, dx*)*Angle of the line segment
$distance \leftarrow \|node_1 - node_2\|_2$ *Euclidean distance $n\_step \leftarrow$ [*distance/param.resolution*] *Number of steps for scanning
for $i \leftarrow 0$ **to** *n_step* **do**
  $current\_y \leftarrow node_1(1) + i \cdot param.resolution \cdot \sin(\theta)$
  $current\_x \leftarrow node_1(2) + i \cdot param.resolution \cdot \cos(\theta)$
  *Check for out-of-map boundaries **if** $current\_x < 1$ $current\_x > size(map, 2)$ $current\_y < 1$ $current\_y > size(map, 1)$ **then**
    return *TRUE* *Path extends beyond map limits
  end
  *Check for collision with an obstacle **if** $map(round(current\_y), round(current\_x)) == 2$ **then**
    return *TRUE* *Obstacle detected at current point
    end
  end
flag ← FALSE *No collision found along the entire path **return** *fla*
end

D. Computational Complexity Analysis

To comprehensively evaluate the proposed APF-RRT* algorithm, this section analyzes its time complexity. For each expansion in the RRT algorithm, the algorithm randomly generates a node and connects it to other nodes in the tree. The computational complexity of this process depends on the number of nodes in the tree. Due to the lack of an optimization mechanism, the RRT algorithm has a time complexity of $O(N)$, where N is the number of iterations. Each expansion requires checking for obstacles, and the path is not optimal, requiring multiple iterations to find a better path. Secondly, the RRT* algorithm improves upon the basic RRT algorithm by gradually optimizing the path through a rewiring mechanism. Each time a new node is expanded, neighboring nodes must be recalculated and the parent node reselected, increasing the computational effort for each expansion. Therefore, the RRT algorithm has a time complexity of $O(N \log N)$, where N is the number of nodes in the tree. As the number of nodes increases, the computational time of the algorithm grows logarithmically. Finally, the APF-RRT* algorithm combines an Artificial Potential Field (APF) with RRT*, guiding the growth of the search tree through attractive and repulsive forces. In each iteration, in addition to checking

neighboring nodes, the attractive and repulsive forces on each node are calculated, resulting in a time complexity of $O(N \log N + M)$, where N is the number of nodes and M is the APF computation overhead for each node. Although APF-RRT* increases the potential field computation, it accelerates convergence through a guided sampling strategy, avoiding the inefficiency of pure random sampling.

IV. Simulation Analysis

The RRT* method is combined with the artificial potential field and evaluated. It is then compared with the original algorithm and other algorithms. This is done to verify whether the improved algorithm can more effectively search for the target point and for paths in different scenarios. This work solely looks at two-dimensional space. The size of the simulation map is a 40×40 grid. Three scene maps are set, a simple map, a complex map, and a narrow channel to ensure the accuracy of the experiment. The simulation scene is shown in Fig. 3. Simulation parameters of different algorithms are set uniformly to ensure data accuracy, as shown in Table I.

TABLE I. Experimental Parameter Settings

Name	Value	Unit
r_{rewire}	30	grid
r	0.5	grid
n	20000	frequency
p_{target}	5%	-
p_{random}	47.5%	-
$p_{non-target}$	47.5%	-
k_a	0.5	-
k_r	0.5	-

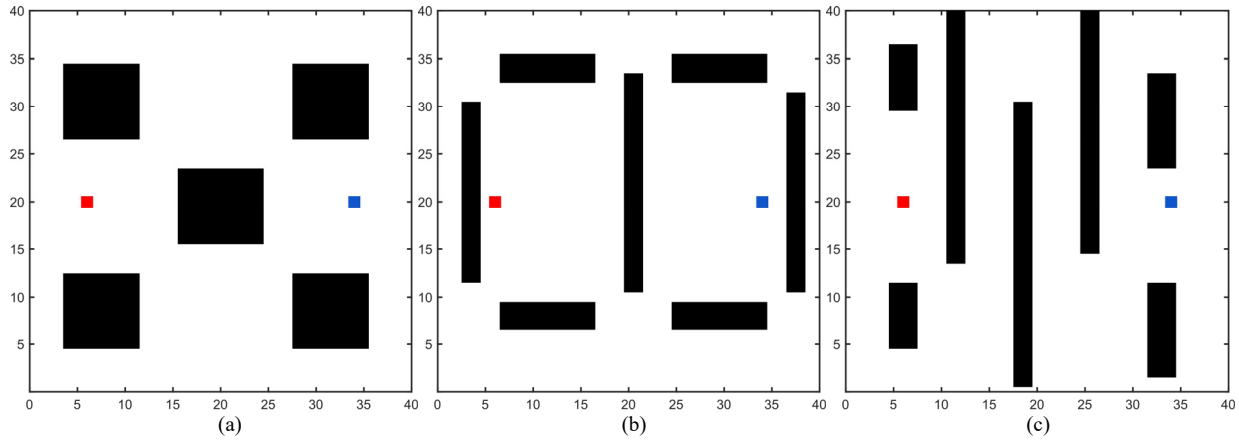

Fig. 3. Simulation scene maps. (a) simple map; (b) complex map; (c) narrow passage.

The specific parameters are the starting point (20, 6) and endpoint (20, 34). The starting point is marked in red, and the endpoint is in blue. The step size is 30. In the experiments, if the robot's search trajectory cannot intersect the obstacle, a collision results. Since the fast search tree has a defined degree of randomness, every algorithm is tested 50 times in each scene map to ensure data accuracy. At present, the quantitative indicators are the search time T/s, path length L/m, and average number of iterations n/times. These are used as algorithm performance indicators.

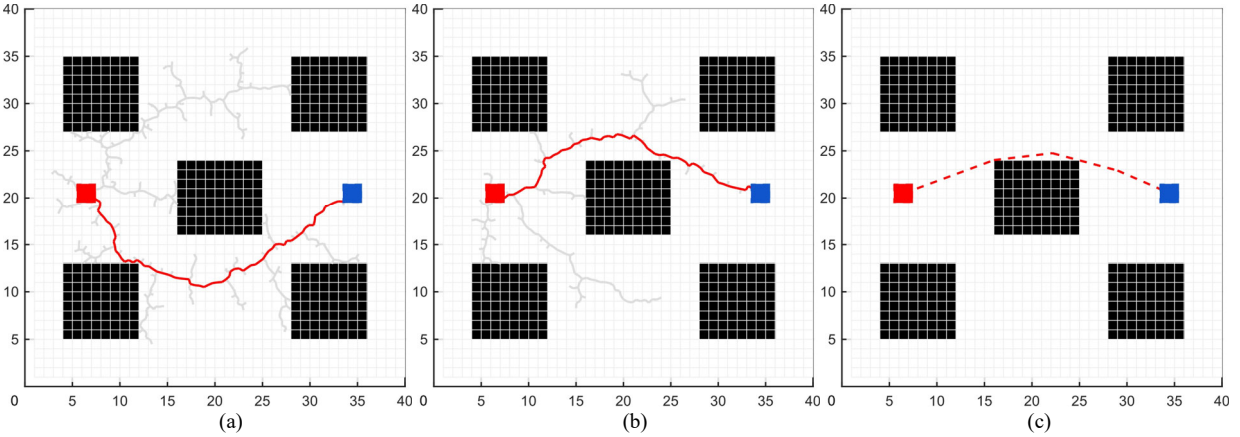

Fig. 4. Simulation of simple map comparison. (a) RRT algorithm; (b) RRT* algorithm; (c) APF-RRT * algorithm.

During the random tree algorithm search process, the red lines in Fig. 4 represent the algorithm planning paths, and the gray lines show the extended node routes. Fig. 4(a) illustrates the simulation trajectory diagram for the RRT algorithm. The search tree makes an excessive number of path points, and movement toward the goal location is

slow. Fig. 4(b) is the simulation trajectory diagram of the RRT* algorithm. The obtained path diagram significantly reduces the path length, but the path is still curved with poor convergence.

Fig. 4(c) shows the APF-RRT* algorithm results. There are fewer duplicate path points, which makes the path points much smaller. The tree expands toward the goal. The average length of the search path, the average cost of the search time, and the standard deviation of the three algorithms are examined in a simple map to further compare their search conditions. If the algorithm runs more than 20,000 times during the simulation, it is considered a failed search.

As shown in Table II, the better algorithm finds the shortest path for robot planning, the smallest number of iterations, and a clear target for the search path. Since the random tree iterations of the RRT and RRT* algorithms lack directionality, the iteration directions are scattered, which causes the failure of these two algorithms. The improved APF-RRT* method reduces the average path length by 31.6% and 27.77% compared to the RRT and RRT* algorithms, and the average iteration time by 50% and 93.47%, respectively. The algorithm results are closer to the average values, standard deviations are smaller, while the target is clearer and more quickly reached compared with the RRT and RRT* algorithms.

TABLE II. COMPARISON DATA OF SIMPLE MAP EXPERIMENTAL RESULTS OF THREE ALGORITHMS

Algorithm	Performance Indicators	Average Value	Standard Deviation	Maximum	Minimum	Average Number of Iterations
RRT	L/m	39.43	2.6	44.75	34.42	1093
	T/s	0.69	0.6	2.76	0.13	
RRT*	L/m	38.28	2.34	46.39	34.13	610
	T/s	0.89	0.17	1.36	0.59	
APF-RRT*	L/m	29.96	0.5	31.12	29.29	583
	T/s	0.46	0.15	1.06	0.24	

In complex maps, obstacles with different shapes are added, as shown in Fig. 5. The RRT* and RRT algorithms have large path node redundancy, slow convergence, and search failures. However, the APF-RRT* algorithm is more efficient and follows a straighter path.

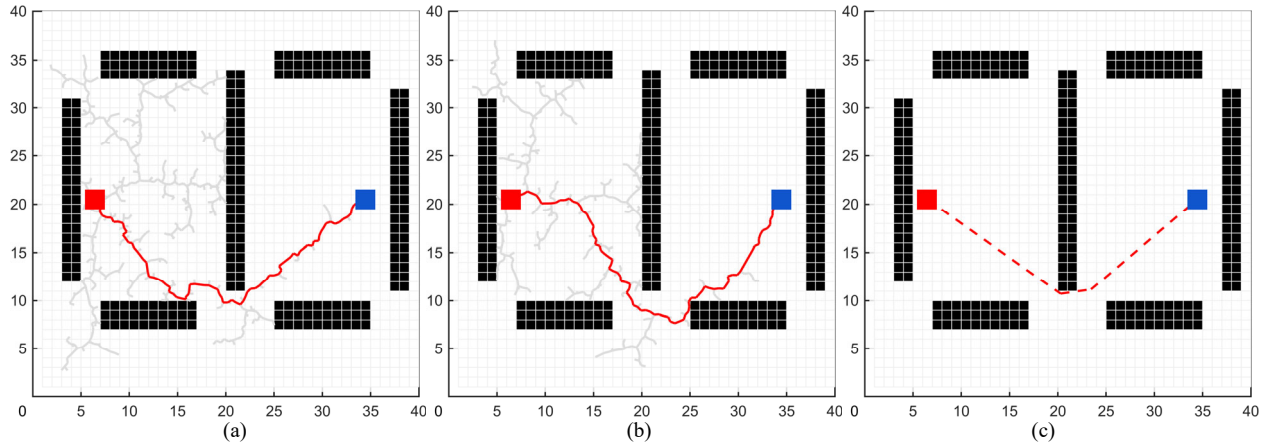

Fig. 5. Comparison of simulated complex maps. (a) RRT algorithm; (b) RRT* algorithm; (c) APF-RRT* algorithm.

The experimental data of the three algorithms for complex maps are shown in Table III. The enhanced algorithm shortens the average path length by 33.41% and 26.64% compared to the RRT and RRT* algorithms, while the average iteration time is reduced by 21.36% and 86.32%, respectively. Additionally, in comparison to the RRT* and RRT algorithms, the proposed approach exhibits a smaller standard deviation, with its operational results being more tightly clustered around the mean value. This characteristic enables more explicit and rapid convergence toward the target objective.

TABLE III. COMPARATIVE EXPERIMENTAL RESULTS OF THREE ALGORITHMS ON COMPLEX MAPS

Algorithm	Performance Indicators	Average Value	Standard Deviation	Maximum	Minimum	Average Number of Iterations
RRT	L/m	46.71	4.94	60.97	38.26	1892
	T/s	1.42	1.11	5.12	0.25	
RRT*	L/m	44.34	2.69	54.72	39.4	1204
	T/s	2.18	0.91	5.39	1.05	
APF-RRT*	L/m	35.01	1.25	39.76	34.38	1310
	T/s	1.17	0.55	2.81	0.54	

Obstacles in narrow channels are added, as shown in Fig. 6. The first two algorithms have large path node redundancy and long planning times. This is especially true for the rewiring of the RRT* algorithm in the search, which leads to a much longer planning time than for the other two algorithms for a better path. The improved algorithm has a short and linear path with fast convergence.

The experimental data of the three algorithms in narrow channels are shown in Table IV. The new approach cuts the path length by 49.6% and 49.8% compared to the RRT and RRT* algorithms, and the average iteration time by 30.81% and 605.35% correspondingly. Additionally, the outcomes generated by our algorithm show a stronger tendency to gather around the mean, with standard deviations being more reduced. Furthermore, it features a more definite targeting direction and attains the objective at a faster pace.

TABLE IV. COMPARISON OF NARROW CHANNEL EXPERIMENTAL RESULTS USING THREE ALGORITHMS

Algorithm	Performance Indicators	Average Value	Standard Deviation	Maximum	Minimum	Average Number of Iterations
RRT	L/m	89.07	6.95	115.39	74.08	14052
	T/s	49.38	19.07	104.07	20.82	
RRT*	L/m	89.15	7.9	113.52	80.59	13372
	T/s	287.97	104.17	512.82	100.88	
APF-RRT*	L/m	59.51	0.71	61.37	58.38	10971
	T/s	37.74	12.52	75.08	16.26	

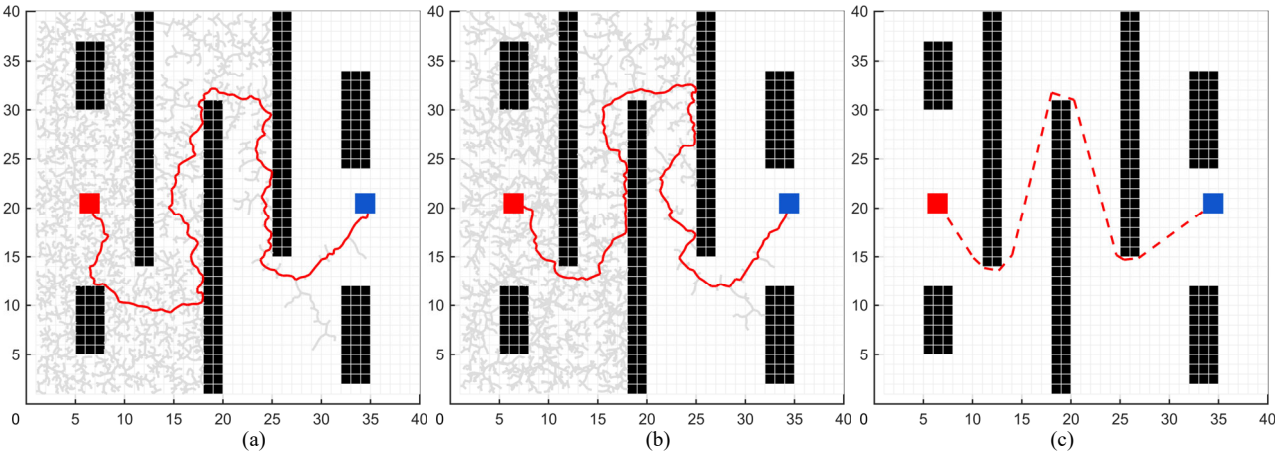

Fig. 6. Comparison of simulated narrow channels. (a) RRT algorithm; (b) RRT* algorithm; (c) APF-RRT* algorithm.

Secondly, we compared the path lengths and iteration times of the RRT, RRT* and APF-RRT* algorithms in different map scenarios through one-way Analysis of Variance (ANOVA). The results showed that the differences in path lengths and iteration times between the algorithms were statistically significant across all three map scenarios ($p \leq 0.05$). The specific F-values and p values are shown in Table V.

TABLE V. ONE-WAY ANOVA

Map Name	Algorithm	Path Length		Iteration Time	
		F-value	p-value	F-value	p-value
Simple map	RRT	38.13	4.03×10^{-5}	38.12	4.04×10^{-5}
	RRT*	39.51	3.2×10^{-5}	40.5	2.2×10^{-5}
	APF-RRT*	29.96	3.5×10^{-6}	32.34	3.2×10^{-6}
Complex maps	RRT	46.71	5.33×10^{-8}	48.12	5.5×10^{-7}
	RRT*	47.38	4.3×10^{-8}	44.8	1.2×10^{-6}
	APF-RRT*	35.01	3.8×10^{-7}	34.9	4.1×10^{-7}
Narrow passage	RRT	89.07	2.5×10^{-7}	34.9	4.1×10^{-7}
	RRT*	91.15	1.5×10^{-6}	93	2.5×10^{-5}
	APF-RRT*	59.51	3.1×10^{-8}	60.23	4.5×10^{-8}

Table V summarizes the performance comparison of three path planning algorithms (RRT, RRT*, and APF-RRT*) across three representative map environments. The results demonstrate that APF-RRT* consistently outperforms the other algorithms in terms of both path length and iteration time. In the simple map, APF-RRT* achieves the shortest path (29.96) and the lowest iteration time (32.34), outperforming both RRT and RRT*. In the complex map, APF-RRT* again shows clear superiority, with a significantly shorter path length (35.01) and reduced iteration time (34.9). For the narrow passage scenario, APF-RRT* achieves the most optimized path (59.51), markedly shorter than RRT and RRT*, although its iteration time (60.23) is slightly higher than RRT but still considerably lower than RRT* (93). Furthermore, all p-values are far below 0.05, indicating that the observed differences are statistically significant. Overall, APF-RRT* demonstrates a robust balance between path quality and computational efficiency across various environments, with particular advantages in complex and constrained scenarios.

To better validate the APF-RRT* algorithm, it was compared with the Informed RRT* algorithm in the same environment. Informed RRT*, after finding an initial

solution, continues searching for a better path within the ellipse, instead of randomly sampling across the entire free space. The average path lengths of Informed RRT* on simple maps, complex maps, and narrow channels were 35.55, 48.78, and 84.88, respectively. Compared to RRT and RRT*, it has an advantage in search length, but it is time-consuming due to the elliptical sampling. APF-RRT* reduces the relative path length by 15.72%, 39.33%, and 42.63% compared to Informed RRT*. In narrow channels, the disadvantage of Informed RRT*'s elliptical sampling is more pronounced, highlighting the advantage of APF-RRT*'s hybrid sampling strategy.

V. Conclusions

This paper proposes an improved APF-RRT* algorithm for robot path planning in a two-dimensional environment. Firstly, aiming at the random problem existing in the RRT* algorithm, it is proposed to integrate the APF and combine it with the RRT* algorithm. Secondly, aiming at the problem of local minima existing in the artificial potential field, a dynamic sampling strategy is proposed to adaptively search in different environments. Finally, for collision detection, a linear collision detection is proposed, which adaptively changes the step size to improve the narrow search ability. Despite these advantages, the algorithm still has some limitations. The potential field parameters need to be carefully adjusted to avoid local minima in a highly chaotic environment. In addition, the current methods mainly conduct evaluations in two-dimensional static scenarios. Its scalability in 3D scenes or dynamic environments still needs to be verified, and relevant research will be conducted in the next step.

Nomenclature

X_{free}: State space.
C_{obs}: Initial state space.
X_{start}: Random tree starting point.
X_{goal}: Random tree target point.
X_{rand}: Random points.
X_{near}: The point in the random tree closest to the random point (Xrand).
X_{new}: New node.
$U_{att}(q)$: Gravitational potential field.
$U_{rep}(q)$: Repulsive potential field.
k_a: Gravitational gain coefficient.
r_{rewire}: Relocation radius.
r: Extended distance.
q_{goal}: Target point.
$F_{rep}(q)$: Repulsion function.
q_{obs}: Obstacle configuration.
F_{total}: Combined force.
p_{target}: Target bias sampling probability.
$p_{non-target}$: Non-target bias sampling probability.
p_{APF}: APF guides sampling probability.
p_{random}: Random sampling probability.
N: Number of iterations.
r: Maximum extension distance.
k_r: Repulsion gain coefficient.
x_i : Represents a node in the search tree.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

YZ conducted the research, draft preparation, data curation, validation, methodology, formal analysis; MP conceptualization, writing, reviewing and editing, formal analysis; VC writing, formal analysis; SC writing, formal analysis; SE conceptualization, formal analysis; all authors had approved the final version.

Acknowledgment

The authors would like to express their gratitude to the Shenzhen City Polytechnic for their support with facilities.

References

[1] L. Liu, X. Wang, X. Yang *et al*., "Path planning techniques for mobile robots: Review and prospect," *Expert Syst. Appl.*, vol. 227, 120254, 2023.
[2] S. M. Langbak, C. Schou, and K. D. Hansen, "Multi-autonomous mobile robot traffic management based on layered costmaps and a modified Dijkstra's algorithm," in *Proc. IEEE Int. Symp. Multimedia*, 2023, vol. 232, pp. 53–63.
[3] F. Duchon, A. Babinec, and M. Kajan, "Path planning with modified A* algorithm for a mobile robot," *Procedia Engineering,* vol. 96, pp. 59–69, 2014.
[4] L. E. Kavraki, P. Svestka, J. Latombe *et al.*, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, 1996.
[5] C. Yuan, G. Liu, W. Zhang *et al.*, "An efficient RRT cache method in dynamic environments for path planning," *Robotics Auton. Syst.,* vol. 131, 103595, 2020.
[6] Z. Li, M. Du, J. Qin *et al.,* "Research on robot path planning based on multi-strategy genetic ant colony optimization algorithm," *Inf. Sci.*, vol. 718, 122407, 2025.
[7] Y. Huang, X. Ke, and Y. Jiang, "Theme park greening VR design based on entertainment robots and genetic algorithm optimization: Digital entertainment design experience," *Entertain. Comput*., vol. 52, 100840, 2024.
[8] L. Abualigah, "Particle swarm optimization: Advances, applications, and experimental insights," *Computers, Materials & Continua*, vol. 82, no. 2, 2025.
[9] Z. Sun, Y. Zhou, W. Xu *et al*., "Walking control of semi-passive robot via a modified Q-learning algorithm," *Int. J. Non-Linear Mech.*, vol. 161, 104691, 2024.
[10] Karthikeyan and D. B. Rani, "An innovative approach for obstacle avoidance and path planning of mobile robot using adaptive deep reinforcement learning for indoor environment," *Knowl.-Based Syst.*, vol. 326, 114058, 2025.
[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.
[12] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2014, pp. 2997–3004.
[13] R. Mashayekhi, M. Y. Idris, M. H. Anisi *et al*., "Informed RRT*-connect: An asymptotically optimal single-query path planning method," *IEEE Access*, vol. 8, pp. 19842–19852, 2020.
[14] B. K. Patle, S. Chen, A. Singh *et al*., "Optimal trajectory planning of the industrial robot using hybrid S-curve-PSO approach," *Robot. Intell. Autom*., vol. 43, no. 2, pp. 153–174, 2023.
[15] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Annu. Research Report*, 1998.

[16] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, 2000, pp. 995–1001.
[17] I. B. Jeong, S. J. Lee, and J. H. Kim, "Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate," *Expert Syst. Appl.*, vol. 123, pp. 82–90, 2019.