Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 5
AIDML 2024

Path Planning Algorithm for Mobile Robots

Hanwen Jian *

School of Mechanical and Electrical Engineering, Zhongkai University of Agriculture and Engineering, Guangzhou, 510225, China

* Corresponding Author: 1808010426@stu.hrbust.edu.cn

Abstract. Path planning algorithm is a critical technology enabling mobile robots to realize autonomous navigation. In this paper, the path-planning technology of mobile robots is deeply discussed, and the operation mechanism and principle of different algorithms are analyzed in detail. Based on the understanding of the characteristics of mobile robot path planning algorithms, these algorithms are divided into three categories: traditional planning algorithm, intelligent planning algorithm, and sampling-based planning algorithm. This paper reviews and discusses the key research results in recent years, especially the advantages and limitations of various algorithms, which are analyzed in depth. Considering the current research status of mobile robot path planning technology, this paper also forecasts future research trends to provide new thinking direction for further development, such as the mixed-use of different types of path planning algorithms, the adoption of new technologies such as deep learning for path planning, and the application of multi-robot collaborative path planning.

Keywords: Mobile robot; Path planning; Algorithm optimization; Algorithm classification and summary.

1. Introduction

In the era of rapid development of technology, mobile robots have become increasingly widely used in various fields, including but not limited to automated logistics, disaster relief, and home services. One of the core technologies of mobile robots is path planning, that is, how to make the robot move from the starting point to the destination safely and efficiently. This paper is a comprehensive review of the current mobile robot path planning field of a variety of algorithms to provide a thorough understanding for readers and explore the advantages and limitations of these algorithms.

Firstly, this paper introduces the basic concepts of path planning, including global and local path planning. Global path planning is concerned with planning an optimal path in a known environment from the beginning to the end. In contrast, local path planning involves planning an optimal path in an unknown or partially unknown environment, while real-time path planning is based on current perceptual information.

Then, this paper discusses several classical path planning algorithms in detail, including the A * algorithm, the Dijkstra algorithm, the Rapidly expanding Random Tree (RRT) algorithm, and path planning based on a genetic algorithm. This paper introduces its basic principle and implementation for each algorithm and analyzes its applicable scenarios, advantages, and possible problems.

2. Traditional Programming Algorithm

2.1. Artificial Potential Field

Khatib proposed the artificial potential field (APF) method in 1985 to plan mobile robots to achieve obstacle avoidance capabilities [1]. The algorithm leverages the "potential field" concept in physics, equating obstacles to equivalent repulsive sources, generating "repulsion" forces for the mobile robot. It also equates the target point to an equivalent attractive source, generating "attraction" forces for the robot. During the robot's movement, the "attraction" and "repulsion" forces continuously change, and

the robot changes its movement direction in real-time based on the net force it experiences, thereby achieving obstacle avoidance during movement. This method has high real-time performance and a relatively simple implementation structure, giving it practical value. The schematic diagram of the artificial potential field method is shown in Figure 1.

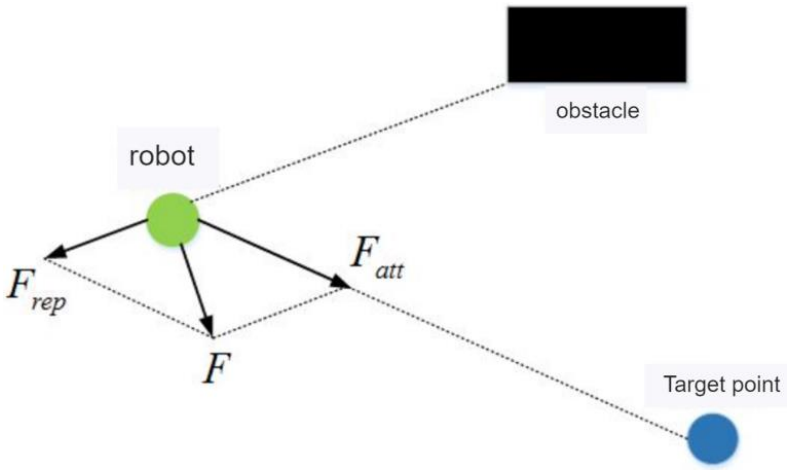

Figure 1. Schematic diagram of artificial potential field method. (Picture credit: Original)

The AFP method has certain limitations. Especially when mobile robots approach obstacles, their path planning may experience "oscillation" effects, affecting the robot's stability during task execution. This method is prone to local minima; for example, when the obstacle is located at the center of the mobile robot and the target point, the "gravitational" and "repulsive" forces acting on the mobile robot cancel out each other. Currently, the robot cannot determine the direction of movement, leading to path planning failure. Many scholars have made improvements to this method in different directions. LiG overcomes the oscillation effect in the artificial potential field method by setting virtual local targets [2]. Cong solved the problems of trajectory oscillation and unreachability by setting detection windows and distance influence factors [3]. Bo introduced variable gain coefficients into the repulsive potential energy function to solve the oscillation problem [4]. Wang Bing dynamically sets virtual target points based on environmental information and quickly escapes local minimum points in the path [5]. Shi proposed a method of concatenating local minimum regions to disperse obstacles for the problem of local minima in the field of influence [6]. Experimental results have shown that this method effectively solves the problem of local minima.

2.2. Grid Algorithm

2.2.1. Dijkstra.

In 1973, Johnson proposed the Dijkstra algorithm [7]. The Dijkstra algorithm finds the shortest distance from one vertex to another in a weighted graph. The algorithm saves the shortest distance from the starting point to each vertex by array D, and the vertex corresponding to the shortest path traversed by Set T. When executing the algorithm, every time the vertex with the smallest distance from the starting point is obtained from the vertex outside the set T, the vertex is added to the set T, and the value in the array D is refreshed through the vertex, until the collection T contains all vertices.

Dijkstra's algorithm is robust and can calculate the optimal path between two points. However, the algorithm is an undirected search algorithm; with the number of nodes, the algorithm's computational efficiency decreases. Based on the shortcomings of the Dijkstra algorithm, scholars have made many improvements [8]. At present, some research has been done to improve the performance of the Dijkstra algorithm by changing the weight criterion of the weighted graph. Still, most of them have improved the search efficiency of the Dijkstra algorithm by limiting the search area. For example, Zhang et al. [9] reduced the search area to a rectangular area with the starting point and the target point as diagonals and lines and gradually expanded the sampling area when the search area could not find the path; Gong et al. [10] respectively create adjacency matrix according to path weight and

time weight, which reduces the minimum path search domain and the optimal path update domain in the iterative process; Guo et al. [11] use the connected domain to represent obstacle region, and store the distance information of adjacent nodes in the connected domain by a linked list so that the search range is kept in the obstacle avoidance space; Jia et al. [12] constructed the convex hull model of the starting point, the target point and the obstacle by mathematical means, and transformed the whole map path solving problem into solving, and solved the shortest path problem from the starting and ending point to the target point in the convex hull.

2.2.2. A*.

In 1972, Hart et al. proposed a * algorithm [13]. A * algorithm is essentially a Heuristic algorithm, that is, through a certain number of indicators to guide the algorithm run. A * algorithm introduces the cost function as the evaluation index, and the cost function is shown in formula (1)

$$F(n) = g(N) + H(N) \tag{1}$$

where G (N) represents the cost from node n to the starting point, which is the current actual cost, and H (N) represents the cost from node n to the target point, which is the estimated cost, there are two methods to calculate the cost from node n to target point: Euclidean distance and Manhattan distance, and G (N) represents the total cost of node n.

Improvement of heuristic function: refinement of heuristic function: adjust the heuristic function according to the specific problem, make it closer to the actual situation, and reduce the search space.

(1) Dynamic heuristic: dynamically adjust the heuristic function according to the found path information during the search. (2) Data structure optimization: optimize the data structure of Open and Closed lists to speed up the insertion, deletion, and lookup of nodes.

Use efficient priority queue data structures such as Fibonacci heap or paired heaps to manage open lists.

(1) Incremental search: For the dynamically changing environment, using an incremental algorithm, such as D, etc., can reuse the previous search results and reduce re-calculation. (2) Parallel and distributed processing: parallel processing of multiple search branches using multi-core processors or distributed computing resources. Implement parallel versions of algorithms, such as parallel algorithm (PA).

Mnemonic search: stores the path and results searched to avoid repeating the calculation of the same sub-problem in the search process.

Integrate other algorithms: combine with other path planning algorithms, such as RRT or another heuristic, to improve performance or adapt to specific application scenarios.

(1) Preprocessing and spatial decomposition: preprocessing a map or search space, such as by layering or zonal decomposition, to simplify the search problem. Abstract graphs and hierarchical search are used to reduce the actual search space. (2) Dynamic weight: to balance the relationship between the heuristic estimation and the actual cost, the dynamic weight is introduced into the heuristic function to improve the search efficiency. (3) Schema Database: A schema database stores sub-problem solutions in a specific problem domain to reduce search time. (4) Limit the search space: apply a variety of heuristic strategies to limit the search space, such as limiting the search depth and using iterative depth methods.

2.2.3. D*.

In 1994, Stentz proposed the D * algorithm [14], which was developed from the A* algorithm and is suitable for situations where the environment is unknown or has dynamic changes. The D* algorithm makes two significant improvements to the A * algorithm.

1) when the dynamic obstacle appears, the D * algorithm can update the grid information around the dynamic obstacle.

2) the heuristic function of the D * algorithm is transitive. When the central grid extends to the surrounding grid, its heuristic value can be passed to the neighboring points.

The cost formula of the D * algorithm is developed from the A * function, as shown in Formula (2).

$$F(s) = G(s) + H(s) \tag{2}$$

where G (s) is the starting point to the S value, and H (s) is the heuristic value.

D* algorithm improves the A * algorithm to some extent; its search path speed is faster, and the path is better, but the D * algorithm needs to strengthen the path twists or the shortcomings. Moreover, the path planned by the D * algorithm is close to the edge of the obstacle. Similar to the A * algorithm, improving the D * algorithm mainly focuses on improving the smoothness of the path and improving the search efficiency. Some researches improve the search efficiency by reducing the search area. Some studies have optimized the D * algorithm single-time by optimizing the node selection method and other methods to improve the efficiency of the algorithm search. There are also studies to change the cost function by narrowing the search space. There are two main ideas for improving path smoothing: taking smoothing into account in path generation and smoothing after path generation.

3. Intelligent Planning Algorithm

3.1. ACO

Ant Colony Optimization (ACO) is a heuristic that simulates the foraging behavior of ants and is mainly used to solve path Optimization problems. The ant foraging diagram is shown in Figure 2.

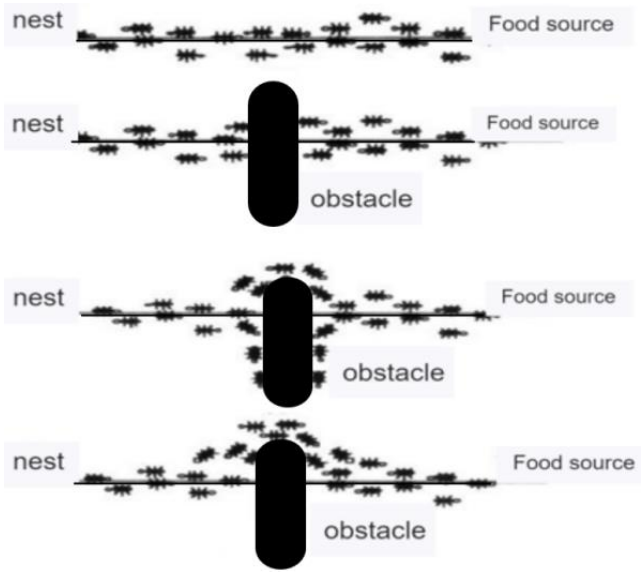

Figure 2. Ant foraging diagram. (Picture credit: Original)

Implementation mechanism and principle: (1) pheromone model: ants in the ant colony algorithm will leave pheromones on the path in the search process; the number of pheromones represents the excellent degree of the path. Other ants tend to choose paths with high concentrations of pheromones. (2) Heuristic information: ants rely not only on pheromone concentrations but may also refer to heuristic information, such as the length of a path or the distance to a target, when choosing where to go next. (3) Probability transfer rule: the probability of an ant moving from one node to the next is calculated based on pheromone concentration and heuristic information, usually as a function of both. (4) Pheromone update: after each algorithm iteration, pheromones on all paths evaporate to avoid

premature convergence to a locally optimal solution. At the same time, the pheromone is updated according to the path quality found by ants in this iteration. (5) Global Search and Local Search: The Ant Colony algorithm balances global search and local search by combining global update and local update of pheromone.

Advantages: (1) distributed computing: The Ant Colony algorithm is a naturally distributed algorithm in which each ant is independent and can process in parallel. (2) Positive feedback mechanism: The excellent path will become more prominent through accumulating pheromones, forming positive feedback, and contributing to rapid convergence. (3) Strong Adaptability: The Ant Colony algorithm can adapt to the dynamic changes of the problem, such as path interruption or target change, and ants can adjust the search strategy according to the new situation. (4) Good robustness: the algorithm is less dependent on the initial solution and, even in the complex search space, can also find a solution.

Limitations:(1)Speed of convergence: The Ant Colony algorithm may need more converging iterations, especially in large-scale problems. (2) Parameter sensitivity: the algorithm's performance mainly depends on parameter settings (such as pheromone evaporation rate, importance, etc.), and parameter adjustment usually requires empirical judgment. (3) Local optimal problem: although the pheromone evaporation mechanism can somewhat prevent premature convergence, the algorithm may still fall into local optimal. (4) Computational resource consumption: for large-scale problems, the ant colony algorithm may consume many computational resources.

The ant colony algorithm uses a distributed computing approach, computing multiple individuals simultaneously; each individual, in real-time, feels the changes in the environment through the release of pheromones to change the surrounding environment. The positive feedback mechanism of the ant colony algorithm makes the individual gradually close to the place of information and prime, which makes the algorithm converge continuously in the process of searching the path. However, in this algorithm, the initial pheromone value is the same, so the next node is chosen at random, which leads to the algorithm needing a long time to play a positive feedback role, thus reducing the algorithm's convergence rate. When the initial solution obtained by the ant colony algorithm is sub-optimal, the positive feedback mechanism can easily make the sub-optimal solution take advantage, even if the algorithm falls into the local optimal, and it is difficult to escape from the local optimal. In addition, the ant colony algorithm has many parameters and a certain degree of correlation; adjusting parameters can improve the performance of the ant colony algorithm, but the adjustment of parameters depends on experience and trial and error. Therefore, in addition to the above research, the optimization of the ant colony algorithm can also improve the initialization of pheromone, which provides a better direction for the algorithm to explore, and can not only solve the local optimization problem but also can improve the initialization of pheromone, it can also improve the convergence speed of the algorithm. In addition, the structure of the ant colony algorithm optimization is also of great significance; by improving the parameters of the association, optimization can increase the algorithm's tunability and enhance the algorithm's ability to optimize.

3.2. Genetic Algorithm

In 1958, Bremermann proposed the genetic algorithm (GA) [15]. This algorithm imitates organisms' genetic and evolutionary mechanisms in reproduction and carries out global optimization. It is implemented by coding the path and establishing the fitness function. A GA can search many areas of solution space simultaneously and gradually lead the search process to the optimal solution. This algorithm has been widely used in mobile robot path planning and other applications.

However, the genetic algorithm could be faster and needs a lot of computing resources, which limits its further development. Many researchers have conducted in-depth research to improve the genetic algorithm. For example, to solve the problem of slow convergence and weak local search ability of traditional genetic algorithms, Wang Hao and others proposed an improved adaptive genetic algorithm [16]. This algorithm improves the search efficiency by optimizing the fitness evaluation index and the selection mechanism. The experimental results show that the average path length is

reduced by 9.9%, and the number of iterations is also reduced. Facing the problem that genetic algorithm is easy to fall into the local minimum, Wang Jidai put forward a fuzzy adaptive genetic algorithm, which can dynamically adjust the operation of crossover and mutation by combining with fuzzy logic; the cosine function is added into the fitness function as the evaluation factor of the smoothness of the path, to reduce the turning point of the path and improve the smoothness of the path [17]. Experimental results show that the proposed algorithm converges faster than the adaptive genetic algorithm, reducing the number of sharp corners by 45.16%. In addition, the cooperative mechanism also helps avoid collision between robots so that the algorithm can be used for multi-robot path planning. Nazarahari proposed an enhanced genetic algorithm, which solves the problem that traditional genetic algorithms may need multiple iterations to find the optimal solution by designing five unique crossover and mutation operations; collision elimination is introduced to make the algorithm applicable to multi-mobile robot path planning [18].

The genetic algorithm and ant colony algorithm are similar; they are from the point of view of the whole population, multi-individual parallel computing, which, to a certain extent, improves efficiency. The genetic algorithm searches through probability iteration, which has a certain randomness and relies on the evaluation function to guide the searching process. Like the ant colony algorithm, the genetic algorithm involves many parameters. These parameters are highly correlated, and parameters set on the algorithm's performance have a significant impact, but the parameters are usually adjusted based on experience. To optimize the algorithm, many researchers have focused on improving the algorithm's performance by changing the parameters, as described in [18]. Compared with the ant colony algorithm, the genetic algorithm makes less use of network feedback information, resulting in slower convergence speed. Therefore, improving the structure of the gene algorithm and improving the utilization of feedback information will help to speed up the algorithm's convergence rate. The genetic algorithm also depends on the selection of the initial population, so using the heuristic method to select a better initial population can improve the algorithm's efficiency.

3.3. Particle Algorithms

Particle swarm optimization (PSO) is a particle swarm optimization optimization tool based on Swarm intelligence proposed by Eberhart and Kennedy in 1995 [19]. Inspired by the foraging behavior of birds, it solves optimization problems by simulating the social behavior of biota, such as flocks of birds or fish.

Principle: in the PSO algorithm, the potential solution of each optimization problem is treated as a "Particle" in the search space. Each particle has a location representing the possible solution and a velocity representing the direction and distance it moves in the solution space. Particles update their speed and position based on individual and group experiences. The workflow diagram of the particle swarm optimization algorithm is shown in Figure 3.

Operation flow: (1) Initialization: Randomly generate a group of particles, each with a random initial position and speed. (2) Evaluation: calculating each particle's fitness value is usually the optimization problem's objective function. (3) Update individual experience: if the current position is better than the best in the particle's history, update its best position. (4) Update the group experience: update the global best position if the current position is better than the best position found throughout the group's history. (5) Update speed and position: adjust the speed and position of each particle based on the current speed, individual best position, and global best position. (6) Iterations: the steps to evaluate, update individual experience, update group experience, and update speed and location are repeated until stopping conditions are met, such as iteration times, precision requirements, etc.

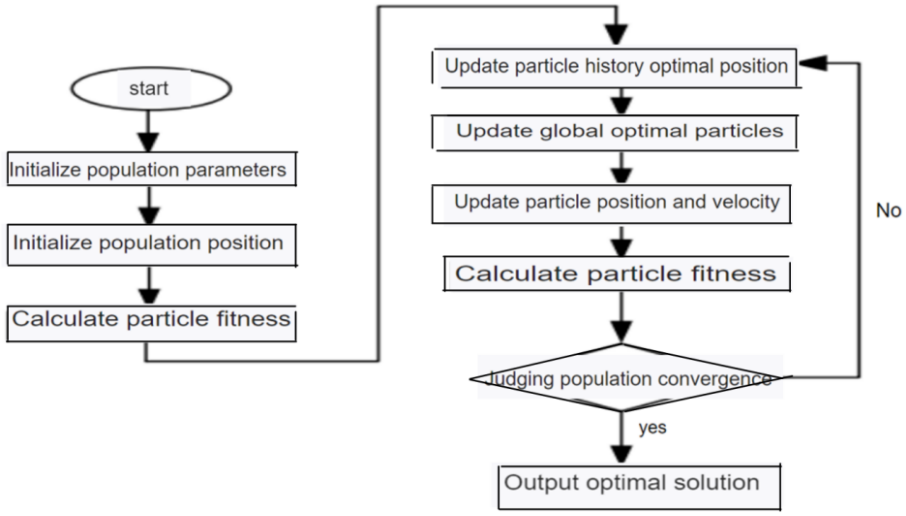

Figure 3. Workflow diagram of particle swarm optimization algorithm. (Picture credit: Original)

Features: (1) Simple and efficient: The PSO algorithm is simple, easy to understand and implement, and highly efficient. (2) No need for gradient information: unlike some optimization algorithms that require the gradient information of the objective function, PSO uses only the objective function values. (3) Broad applicability: can be used for continuous space or discrete space optimization problems

Optimization direction: (1) Parameter selection: the performance of PSO largely depends on the choice of parameters, such as particle velocity, acceleration coefficient, etc. Proper parameter selection can significantly improve the algorithm's performance. (2) Topology: information sharing among particles affects the performance of algorithms. Studying different social network topologies, such as global optimal and local optimal models, can help improve the algorithm. (3) Multi-objective optimization: PSO needs to be extended to accommodate multi-objective optimization problems, which involve how to define and handle trade-offs between multiple objectives. (4) Dynamic and uncertainty problems: real-world optimization problems are often dynamic changes; how to make PSO adapt to the dynamic environment is an important direction. (5) Hybrid Algorithm: PSO and other optimization techniques combined to form a hybrid algorithm can make full use of their respective algorithms' advantages, improving the optimization effect.

4. Sampling Planning Algorithm

4.1. PRM

In 1996, Kavraki and Svestka [20] proposed the PRM multi-query-based planning algorithm, including two stages: learning and query. In the learning stage, the free configuration space randomly generates the feasible sampling points, and the fast path planner connects the feasible sampling points to form the probability roadmap. An optimal path from the initial state to the target state in a probabilistic roadmap. The schematic diagram of PRM algorithm path planning is shown in Figure 4.

PRM algorithm performs well in high-dimensional space, so it has been widely considered and studied. However, in some applications, the amount of computation required to obtain a priori road sign may be too large or even lead to planning failure, which limits the online planning ability of the algorithm. When the number of sampling points is small, the PRM algorithm makes it difficult to plan the path through a narrow channel; increasing the number of sampling points can solve the problem, but the computational cost increases. To address this problem, Ravankar [21] proposed a probabilistic road map algorithm based on hybrid potential, which utilizes map segmentation to generate high-potential and low-potential regions to reduce samples during road map construction and set dispersion. The experimental results show that this method's local and global planning success rates exceed 95%. Zhong [22] By identifying narrow channels and increasing the density of road markers in the channels, the efficiency and success rate of path planning are improved. Cheng [23]

improved the efficiency of searching paths to a certain extent by changing the distance between connecting points. To solve the problem that PRM is challenging to plan when there are few sampling points, the generation function of random sampling points will fall, and the sampling points in obstacles will be transformed into free space to improve the benefits of sampling points, in addition, the path is optimized by node enhancement method. The experimental results show that, compared with the traditional PRM algorithm, the path planning time of the algorithm is shorter, and the path cost is correspondingly reduced.

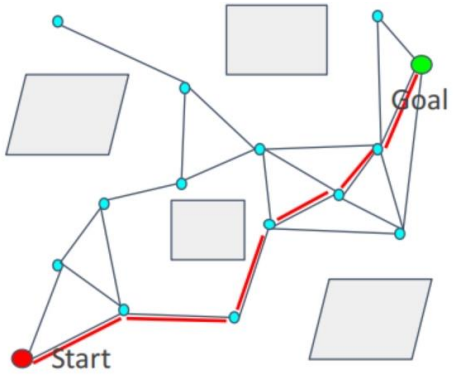

Figure 4. Schematic diagram of PRM algorithm path planning. (Picture credit: Original)

4.2. RRT

In 1998, Lavalle proposed the RRT algorithm [24], which obtains the collision-free path from the starting point to the target point by constructing a random tree. The RRT algorithm takes the starting point as the root node of the random tree and obtains the random points by random sampling. Then, the leaf node is added to the random tree until the target point is added to the random tree, and the iteration is finished. A schematic of the RRT algorithm and extension process is shown in Figure 5.

Principle: the basic idea of the RRT algorithm is to plant a random tree in the search space by random sampling. The tree grows and expands to explore the whole search space quickly. For each iteration, randomly select a point, then find the node closest to that point in the tree as the parent of the extension, and extend toward that point one small step to add new nodes.

RRT algorithm is widely used in robot path planning, autopilot, and other fields. Although the RRT algorithm can quickly find a path in a complex environment, it has some limitations, such as a non-optimal path, low efficiency (especially in high-dimensional space and narrow channels), the exploration of the target region may not be sufficient, and so on. Therefore, researchers have put forward a variety of improved schemes to optimize the RRT algorithm.

RRT Star (RRT*) is an improved version of RRT, which introduces a new node selection and reconnection mechanism in the hope of finding a better path with higher probability. In RRT, every time a new node is added, the algorithm considers expansion from the nearest node and looks for a better parent in a neighborhood of the new node. At the same time, adding a new node may also make other nodes in its neighborhood better parents, so attempts are made to reconnect those nodes.

RRT-Connect RRT-Connect is another popular RRT variant that uses two trees that grow simultaneously: one starts at the beginning and the other at the end. The two trees expand alternately, each time extending to random points and trying to connect directly to the nearest node of another tree. This two-way growth strategy dramatically speeds up the path search.

Informed RRT * Informed RRT builds on RRT by using heuristic information to narrow the search space. Once the first path has been found, Informed RRT * computes an elliptical search space based on the shortest path length currently found. The new sampling only occurs within this ellipse, thus improving the search efficiency and the probability of finding a better path.

RRT # further improves RRT * by introducing a new cost propagation mechanism to accelerate convergence to the optimal path. RRT # attempts to update the optimal path cost for all nodes in the tree in each iteration, not just the cost near the newly added nodes

RRT with Dynamic Domain: This method improves the RRT algorithm by dynamically adjusting the size of the sampling area. For those nodes in narrow channels or near obstacles, the sampling area will be reduced to improve the generation rate of new nodes in these areas, thus enhancing the algorithm's performance in a complex environment.

(1) Optimize content path quality: improve path quality through path smoothing and reprogramming. (2) Search Efficiency: to improve the search efficiency of the algorithm by enhancing the sampling strategy and search mechanism. (3) Computational complexity: reduce the computational complexity of the algorithm, especially for the Galway problem. (4) Environment Adaptability: Improve the adaptability of the algorithm to different types of environments, such as dynamic and unknown environments.

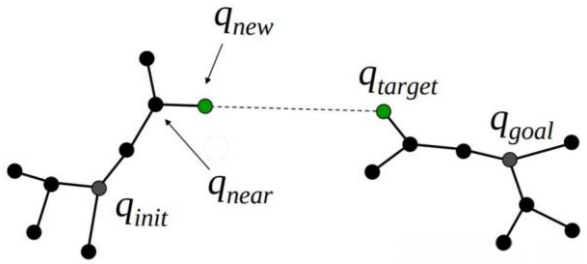

Figure 5. PiA schematic of the RRT algorithm and extension process. (Picture credit: Original)

5. Algorithm Summary and Comparison

This article summarizes the path planning algorithm introduced in a table, as shown in Table 1. This table summarizes the implementation mechanism, principles, advantages, and limitations of the algorithm mentioned in this article.

Table 1. A brief analysis of algorithms.

Algorithm	Implementation Mechanism and Principles	Advantage	Limitations
APF	The direction of motion of the mobile robot is changed by the combination rule of force	Planning is fast in two-dimensional space	Easy to fall into local minimum; produce oscillation problem
Dijkstra	Solve the distance between the vertices of the weighted graph	Strong robustness and fast computing speed	When there are too many nodes in the weighted graph, the planning efficiency is low
A*	The minimum estimated cost of finding the current node to the target point	The calculation method is simple; the planning path is short	There are many inflection points in the path
D*	Finding the minimum synthesis cost from the current node to the target point	Fast computing speed; short planning path	The path is closer to the edge of the obstacle; there are many inflection points in the path
ACO	Ants move to higher pheromones	High robustness	Easy to fall into local optimum

GA	Populations cross and mutate to produce new species	Progressive optimization; overcome local optimization	Low computing speed; large memory footprint
PSO	Based on the idea of swarm intelligence and information sharing. By continuously adjusting the particle's velocity and position, the whole particle swarm searches in the direction of the optimal solution	The principle is simple and easy to realize. It converges quickly and does not depend on the information of the problem. It is easier to find global optimality by leaps and bounds.	It is difficult to obtain very accurate values without ensuring optimality. Easy to fall into a local pole without getting the right result.
PRM	The path network graph is constructed by random sampling in the workspace	Apply to high-dimensional space	A large amount of calculation; not suitable for online planning
RRT	Random trees grow and spread	It is suitable for high-dimensional space, a relatively simple algorithm, fast expansion speed, and differential constraints	High randomness, high path cost, and low efficiency with the complexity of the environment

6. Trends

This paper comprehensively summarizes the leading algorithm theory, its advantages and limitations, and the corresponding improvement measures in mobile robot path planning; these algorithms are divided into three categories: classical planning algorithm, intelligent planning algorithm, and sampling-based planning algorithm.

Efficient implementation of algorithm integration: although the current research mainly focuses on the improvement based on the characteristics of each algorithm, there is relatively little research on the integration and fusion of algorithms, but these attempts have shown promising results. Because a single algorithm often has limitations, such as quickly falling into local optimal, high path curvature, planning efficiency is not high, and different algorithms do not share the same shortcomings, this paper can achieve more efficient planning methods through the combination of highly complementary algorithms. For example, combining the evaluation function of the A * algorithm to optimize the pheromone updating mechanism of the ant colony algorithm not only improves the convergence speed of the algorithm but also avoids the local optimization problem. Therefore, the efficient implementation of path planning in the future will depend on the comprehensive application of multiple algorithms.

Combined with multi-sensor dynamic path planning, the current path planning algorithms are mainly used for offline planning in static environments. However, in practical applications, the environment often changes, and the mobile robot needs to be able to avoid dynamic obstacles in real time. Therefore, future research needs to focus on developing a dynamic path-planning algorithm. In dynamic path planning, updating environmental information in real time is the key. Various sensors, such as depth cameras and radars, can provide information about the distance and shape of obstacles in the environment, helping to assess the distance between the robot and the obstacle in real time. By applying multiple sensors and integrating their data, more accurate path planning and even more complex tasks can be achieved.

Path planning in a complex environment: the current path planning algorithm usually simplifies the starting point and the target point into points, which is relatively simple, and many traditional path

planning algorithms are only suitable for two-dimensional environments when the environmental dimension increases or the obstacles increase, the planning efficiency will decrease. The real work scene is usually more complex, so improving the algorithm's efficiency in a complex environment is essential.

Collaborative planning of Internet of Things technology: In the current application scenario, mobile robots mainly rely on their computing units to complete path planning. With the growth of market demand, the mobile robot is applied in more complex scenes, which will significantly increase the amount of data to be processed by the robot and put forward higher requests to the core processing unit of the robot. With the development of Internet of Things technology, a large amount of data can be transferred to a high-performance remote server through the network for processing, improving the efficiency of path planning and simplifying the structure of mobile robots to some extent.

Performance optimization of existing algorithms: although the existing algorithms have some limitations, and most are applied to simple planning scenarios, these basic algorithms still have their application fields and constitute an essential basis for practical application. Therefore, optimizing different path-planning algorithms according to their limitations is still significant. In future research, this paper can explore the application of more mathematical theory to the optimization of path planning algorithms, for example, improving the pheromone updating mechanism of the ant colony algorithm, optimizing the cost function of the A *and D* algorithm, improving the fitness function of genetic algorithm, etc. Further enhancing the basic algorithm has essential research value for optimizing future practical applications.

7. Conclusion

This paper profoundly discusses and analyzes the path-planning algorithm of mobile robots. This paper summarizes the current mainstream of path planning algorithms, including traditional, intelligent, and sampling-based planning algorithms; their principles, advantages and disadvantages, and possible improvement methods are discussed in detail. Through comparative analysis, this paper finds that although there are many effective path-planning techniques, each has its scope of application and limitations.

The results of this paper show that path planning can be more efficient through algorithm fusion, especially in the face of complex environments and dynamic obstacles. This paper also discusses applying multi-sensor fusion technology in dynamic path planning, which is crucial for updating environment information and improving planning precision. In addition, the research also points out that integrating IoT technology will further enhance the mobile robot's ability to plan in complex scenes and simplify the robot's computing burden through remote processing of large amounts of data.

However, this study has some limitations. For example, the experiments in this paper are conducted mainly in a simulated environment and may differ from the real-world environment. In addition, the fusion effect of the specific type of sensor and algorithm needs further experimental verification.

Future research should focus on the following areas: (1) Developing more efficient algorithm fusion strategies to take full advantage of different algorithms. (2) Exploring more advanced multi-sensor fusion technologies to improve the real-time and accuracy of dynamic path planning. (3) The optimization research is carried out according to the limitations of the existing algorithms.Combined with the Internet of Things technology, explore the potential of distributed computing in path planning.

In short, path planning for mobile robots is a challenging field. With the continuous progress of technology, there will be more intelligent, efficient, and adaptable path-planning solutions in the future to meet the growing demand for applications.

References

[1] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research 2 (1985) 500 - 505.

[2] G. Li, A. Yamashita, H. Asama, et al. An efficient improved artificial potential field-based regression search method for robot path planning. International Conference on Mechatronics &. Automation. IEEE, 9 (8) (2012) 9 - 13.

[3] Y. Cong, Z. Zhao, C. Na, et al. Dynamic obstacle avoidance path planning for unmanned aerial vehicles based on improved artificial potential field. Journal of Weapon Equipment Engineering, 42 (9) (2021) 7.

[4] Q. Bo, S. Liang, C. Lu, et al., Dual robotic arm collision avoidance path planning based on improved artificial potential field method for bidirectional planning. Journal of Jiangsu University of Science and Technology: Natural Science Edition, 35 (5) (2021) 8.

[5] B. Wang, H. Wu, X. Niu, Robot path planning based on improved potential field method. Computer Science. Computer Measurement and Control, 27 (1) (2019) 1 - 12.

[6] W. Shi, X. Huang, W. Zhou, Mobile robot path planning based on improved artificial potential field method. Computer Application, (8) (2010) 3.

[7] D. B. Johnson, A note on Dijkstra's shortest path algorithm. Journal of the ACM, 20 (3) (1973) 385 - 388.

[8] Q. Li, B. Li, R. Zhang, et al. Research on AGV path planning based on improved Dijkstra algorithm CJ Mechanical Engineering and Automation, Computer Simulation. 2021 (1) (2021) 23 - 25.

[9] C. Zhang, X. Li, X. Zhao, Transmission line path planning based on improved Dijkstra algorithm. Electric Power Survey and Design, 2022 (2) (2022) 1 - 5.

[10] J. Gong, Z. Niu, Y. Zhang, Multi objective path planning for campus food delivery robots based on local dimensionality reduction Dijkstra algorithm. Journal of Shandong University of Technology (Natural Science Edition), 35 (4) (2021) 75 - 80.

[11] M. Guo, T. Shi, Research on robot path planning based on improved Dijkstra algorithm lj - electrical technology, Computer measurement and control 20 (20) (2020) 22 - 23.

[12] W. Jia, W. Wei, L. Zhu, et al. Research on path planning and visualization of mobile robots under improved Dijkstra algorithm. Journal of Xuzhou University of Engineering (Natural Science Edition), 36 (2) (2021) 34 - 38.

[13] P. E. Hart, N. J. Nilsson, Raphael B. A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science & Cybernetics, 4 (2) (1972) 28 - 29.

[14] A. Stentz, Optimal and efficient path planning for partially known environments. Robotics and Automation, 1994.Proceedings. IEEE International Conference on IEEE, 14 (6). (1994) 14 - 16.

[15] H. J. Bremermann, The evolution of intelligence: The nervous system as a model of its environment. University of Washington, Department of Mathematics, 1958.

[16] H. Wang, X. Zhao, X. Yuan, Robot path planning based on improved adaptive genetic algorithm. Electro Optics and Control (2022) 1 - 7.

[17] J.Wang, X. Wang, Q. Tian, et al. Path planning for mobile robots based on improved fuzzy adaptive genetic algorithm. Machine Tool and Hydraulic, 49 (23) (2021) 18 - 23.

[18] M. Nazarahari, E. Khanmirza, S. Doostie, Multi-objective multi-robot path planning in continuous environment using an enhanced Genetic Algorithm. Expert Systems with Applications, 115 (2018) 106 - 120.

[19] J. Kennedy and R. Eberhart, Particle swarm optimization, in Proceedings of IEEE International Conference on Neural Networks, pp. 1942–1948, IEEE, Perth, Australia, 1995.

[20] E. Kavraki, P. Svestka, J. C. Latombe, et al., Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation, 12 (4) (1996) 566 - 580.

[21] A. A. Ravankar, A. Ravankar, T. Emaru, et al. HP-PRM: Hybrid potential-based probabilistic roadmap algorithm for improved dynamic path planning of mobile robots. IEEE Access, 8 (2020) 221743 - 221766.

[22] J. Zhong, J. Su, Robot path planning in narrow passages based on probabilistic roadmaps. International Journal of Robotics and Automation, 28 (3) (2013) 29 - 32.

[23] Q. Cheng, S. Gao, K. Cao, et al. Path planning for mobile robots based on PRM optimization algorithm. Computer Applications and Software, 37 (12) (2020) 254 - 259.

[24] S. M. LaValle, Rapidly-exploring random trees: A new tool for path planning. IEEE International Conference on Robotics and Automation. IEEE. 1998.