Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/41/20230716

Path planning algorithms of sweeping robots

Zheyuan Chen[1,5], Jiani Lu[2], Yuqi Shang[3], Diwen Xu[4]

[1]Chongqing Depu Foreign Language School, Chongqing, 401320, China
[2]Changzhou Cardiff and Vale College, Changzhou, 231000, China
[3]Shanghai Caoyang No.2 High School, Shanghai, 200062, China
[4]Shanghai SZA, Shanghai, 200000, China

[5]lbrown83359@student.napavalley.edu

Abstract. Different categories of path planning algorithms for sweeping robot are introduced, including Dijkstra algorithm and A*Algorithm in Traditional path-planning Algorithm, PRM Algorithm and RRT Algorithm in sampling algorithm, and Ant Colony Optimization Algorithms and Genetic algorithms in Intelligent bionic algorithm. Each algorithm has its principles and features introduced. At the same time, several algorithms are compared, and summarized, each algorithm has its advantages and disadvantages, in the future development should be combined with their strengths to optimize the path planning algorithm of the sweeping robot.

Keywords: Computer Science, Path Planning Algorithm, Sweeping Robot.

1. Introduction

As technology continues to advance, various devices have become increasingly integral to our daily lives, such as washing machines, computers, TVs, and more. These devices have profoundly transformed our way of life. In this context, our discussion centers around the sweeper.

To begin, let's delve into the history of the sweeper. In 1996, the Swedish appliance manufacturer Electrolux introduced the trilobite, marking the world's maiden venture into sweeping robots. This innovative sweeper featured a single roller brush for floor cleaning, along with automatic recharge and anti-drop functions. Nonetheless, the Trilobite sweeping robot exhibited several shortcomings. Firstly, it suffered from sluggish reaction, operation, and travel speeds, resulting in lower cleaning efficiency. Secondly, its lofty design limited its ability to clean beneath furniture. Lastly, the steep price made it unaffordable for many.

In 2002, iRobot, an American technology company initially specializing in military and security robots, diversified its focus by introducing the Roomba, a sweeping robot. Originally dedicated to providing robots for diverse purposes like space exploration, battlefield rescue, explosive ordnance disposal (EOD), reconnaissance, security, and research and development for the US government, military, universities, and research institutions worldwide, iRobot made its first foray into the home automation sector with the Roomba in 2002 [1].

After Dyson, a renowned traditional vacuum cleaner brand, faced a sweeping robot project failure over a decade ago, it re-entered the sweeping robot market in 2014 with the launch of the high-end sweeping robot 360Eye. This advanced device employs a top-mounted camera to comprehensively

DOI: 10.54254/2755-2721/41/20230716

observe and analyze its surroundings. It utilizes sophisticated algorithms to create a room map for navigation, adapting its current route based on positional changes among various landmarks in before-and-after images. It continually updates and fine-tunes its environmental model [2].

Despite undergoing numerous changes and improvements, sweepers still exhibit several drawbacks. Notably, they often collide with objects within our homes. To address this issue, it is essential to explore the path planning algorithms.

2. Traditional path-planning algorithm

In the field of computers, it is incredibly important to use algorithms to solve complex problems. Both A* and Dijkstra are used to find the shortest path. This section will be introduced from their principled logic calculation steps.

2.1. Dijkstra's Algorithm

Dijkstra's algorithm is an algorithm for solving the shortest path problem in a graph, conceived in 1956 by the Dutch computer scientist Dijkstra and published in 1959. This algorithm addresses the single-source shortest path problem in a graph by constructing the shortest path tree [3]. Its goal is to determine the optimal route (the shortest path) from the starting node to the ending node, considering numerous vertices and edges. Notably, the weights of the edges must be non-negative. The algorithm employs a greedy strategy, commencing from the starting node and iteratively moving to the neighboring node with the shortest distance from the starting node, which has not yet been visited, until it reaches the end node.

Similarly, Dijkstra's algorithm can be applied to sweeping robots. An essential function of sweeping robots involves designing an optimal cleaning path for an entire room. The quickest and most resource-efficient method for achieving this is to find the shortest path, making Dijkstra's algorithm a suitable choice. Furthermore, Dijkstra's algorithm finds applications in network routing, map navigation, game development, and more.

The process begins with a starting point. At each step, the algorithm identifies the optimal choice from the adjacent vertices of the current vertex. Subsequently, the optimal vertex becomes the current vertex. This sequence of steps continues until the algorithm reaches the end vertex, and the sum of all the distances between the optimal vertices constitutes the shortest path.

2.2. A Algorithm*

The A* algorithm is a type of heuristic search algorithm, and it extends the Dijkstra algorithm through heuristic search. This algorithm can determine the shortest path among multiple nodes (vertices) on a graph by identifying the lowest cost. It combines certain advantages of both Dijkstra's and greedy search algorithms. The A* algorithm was developed by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968 and has had a significant impact on the field of artificial intelligence [4].

This algorithm also finds applications in guiding sweeping robots to identify the shortest path. Due to its extension of the Dijkstra algorithm, it offers improved efficiency.

The A* algorithm takes into account the cost already incurred from the starting vertex to the current vertex and estimates the cost required to reach the end vertex. This is calculated using the formula

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

The process begins at the starting point, and at each step, it aims to identify the best vertex with the minimum value of 'f.' It then proceeds from the best vertex to find the next best vertex in a similar manner until it reaches the endpoint [5].

Using the A* algorithm to find the shortest path in an AB graph involves considering the entire graph as a grid of squares. It maintains two lists: the open list (which includes squares to be examined) and the close list (which contains squares that have already been identified). The A* algorithm takes into account both past and potential future costs, as expressed by (1).

DOI: 10.54254/2755-2721/41/20230716

Here's the process:

Start by placing the starting point 'A' in the open list.

Repeatedly:

Examine points adjacent to the current point.

Iterate through the open list and select the point with the smallest 'F' value as the current point to process.

Move that point to the close list (squares already identified).

Unreachable or points in the close list are no longer considered.

If a point is not in the open list, add it and calculate 'G,' 'H,' and 'F' values based on the current point (using it as the parent).

If a point is already in the open list, check if this path is better (based on 'F' = 'G' + 'H') and may need reordering.

Continue this process until the best route is found, an error occurs, or the open list is empty.

Finally, trace back through the identified path to determine the optimal route. Figure 1 shows an example of how to find the shortest path by A* algorithm.

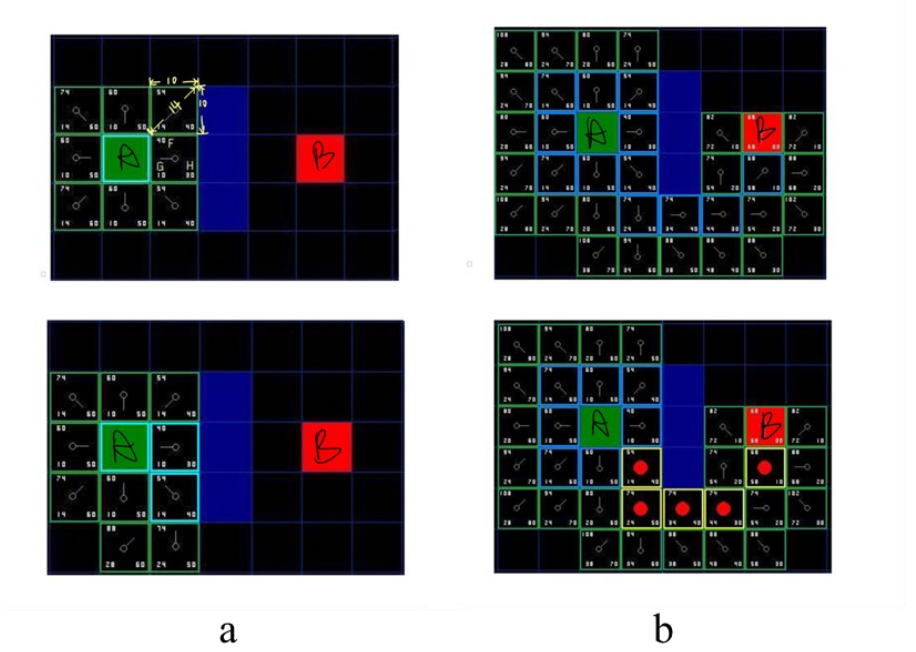

Figure 1. The example of how to find the shortest path by A* algorithm: a: configuration 1; b: configuration 2.

3. Based on the sampling path planning algorithm

3.1. PRM Algorithm

This algorithm plans a course of action through multiple samples of the environment. Its essence is to use the probability graphs to represent the path that the robot can move.

There are two main stages of PRM - Learning and Planning.

3.1.1. Learning phase

There are four steps in the learning phase: create sampling point; link each point bit; construct a network path diagram; clear an obstructed path.

When judging the path, the robot will determine the position of the obstacle according to the collision, and then screen out the wrong point, and routes associated with this point will not be

DOI: 10.54254/2755-2721/41/20230716

counted in the network diagram [6]. Figure 2 is a simple schematic diagram of the principle of PRM algorithm.

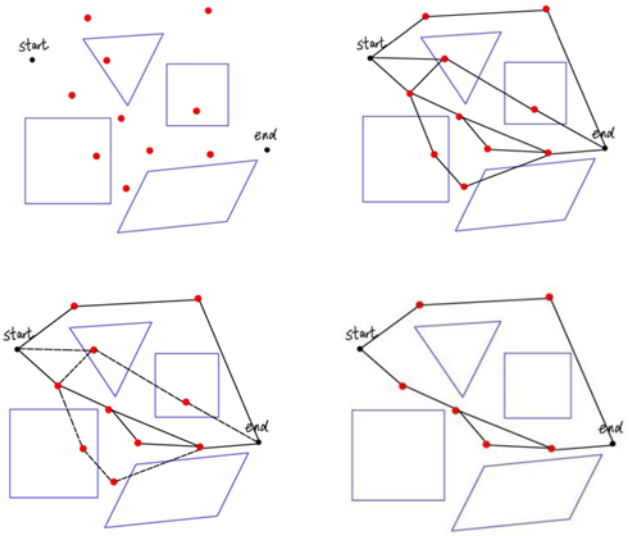

Figure 2. Schematic diagram of the principle of PRM algorithm.

3.1.2. Planning phase

There are two steps in the planning phase: link start and end; combine the mesh map to find the appropriate route. In all possible routes, the robot will find the optimal solution according to the point closest to the starting point and the target point.

PRM algorithm is very helpful for path planning, but it also has limitations. The essence of its planning is to rely on sample surveys. So, in the face of more complex terrain, it is easy to plan failure, or the sampling points are too few, and the planned route is not the optimal solution. And too much data will lead to too much computation and too low efficiency of the robot [7]. Figure 3 shows the results of the PRM algorithm.

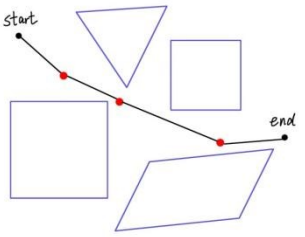

Figure 3. The results of the PRM algorithm.

3.2. RRT Algorithm

Similar to PRM, the RRT algorithm also employs random sampling. However, it differs significantly in that it creates a tree-like path graph. The RRT algorithm initiates by identifying the starting point, denoted as “S.” Next, it selects a random sampling point, “P0,” and establishes the step size, “R,” as the linear distance between points “S” and “P0.” This constitutes one iteration. The process is repeated by choosing another random point, “P1,” on the map, where the distance between “P0” and “P1” remains consistent at the step size “R.” Subsequently, the algorithm assesses whether each point is

DOI: 10.54254/2755-2721/41/20230716

within an obstacle, based on robot collision detection. This iterative approach continues until the final path is determined [8]. Figure 4 provides a simple diagram illustrating the RRT algorithm.

While the RRT algorithm is relatively straightforward, it has limitations when it comes to finding the optimal path. It relies on a fixed step size, which can pose challenges in complex, obstacle-rich environments. To navigate such environments using the RRT algorithm, multiple iterations are often required, involving node screening based on collision obstacles. This process can be time-consuming and may not yield the shortest path [9].

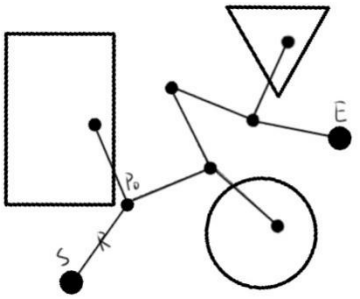

Figure 4. Illustration the RRT algorithm.

3.3. Summary

PRM algorithm relies on the establishment of network graph, the probability of the algorithm is complete, but the collision detection time is long and the efficiency is low. The RRT algorithm relies on the establishment of a tree graph, and the strategy is random sampling, so the efficiency is not high. They both need sampling to plan the path, and the advantage of both is obvious, they can get the final solution, but they need a lot of sampling points. This is also their disadvantage, too much sampling leads to a large number of operations, and they all need robot collision to eliminate the wrong line, which makes the work efficiency low. In practical application, once the complex environment is encountered, the two can get the route, but it will take a long time, and it is not necessarily the optimal solution.

4. Intelligent bionic algorithm

4.1. Ant Colony Optimization Algorithm

The ant colony algorithm is a biomimicry algorithm inspired by the foraging behavior of ants in nature. During their foraging process, ants can consistently discover the optimal path from their nest to a food source.

Ants are among the most common and abundant insects encountered by humans, often forming swarms in daily life. The biological intelligence exhibited by these insect populations has piqued the interest of scholars. Italian researchers, such as Dorigo and Maniezzo, observed ants' foraging habits and noted their ability to identify the shortest path to food sources. Research has shown that this population coordination among ants is achieved through the communication and coordination facilitated by a volatile chemical called pheromone, which they leave along their path [10].

The Ant Colony Optimization algorithm draws its inspiration from the behavior of ants, specifically their practice of releasing pheromones while foraging. In the context of a sweeping robot, the system constructs a comprehensive understanding of its surroundings and employs this algorithm to select and follow the most optimal path [11].

4.2. Genetic Algorithm

Proposed by John Holland in the 1970s, the genetic algorithm was developed based on the principles of biological evolution observed in nature. It simulates the biological evolution process described in

DOI: 10.54254/2755-2721/41/20230716

Darwin's theory of evolution and replicates the genetic mechanisms of natural selection. Essentially, it's a method for discovering the best solutions by mimicking natural evolutionary processes. Its core nature lies in being an efficient, parallel, global search method that autonomously acquires and accumulates knowledge about the search space during the exploration process. It also adaptively controls the search for the optimal solution.

In the operational range of an intelligent sweeping robot, obstacle data is typically three-dimensional, while environmental data is two-dimensional. Consequently, the robot itself is treated as a point, and the obstacle information is transformed from a three-dimensional representation to a flattened one. Additionally, the size of the obstacles is increased by half of the robot's radius. This transformation simplifies the robot's ability to identify and assess obstacles.

The simulation of the sweeping robot's surrounding environment involves fitting the working path T by connecting it with several tangential paths $\vec{l_\iota}$.

$$T = \overrightarrow{l_1} + \overrightarrow{l_1} + \ldots + \overrightarrow{l_{n-1}} \tag{2}$$

In the process of storing path data, a coordinate point storage method is employed. The value 'x' in two-dimensional coordinates represents the precise position of the sweeping robot within the current path. Following genetic principles, the process of path determination is referred to as encoding the acquired chromosomes. In the actual path planning process, the sweeping robot's path can be systematically generated through repetitive problem-solving operations [12].

$$f(\vec{l_\iota}) = \begin{cases} 1 \\ 0 \end{cases} \tag{3}$$

The mentioned function operates as follows: when a specific elementary motion unit doesn't intersect any obstacles, its value is set to 0. Conversely, when the chosen elementary motion unit intersects any spatial obstacle, its value becomes 1. Additionally, g_i is utilized to signify that during the path planning process, there may be a possibility of intersecting obstacle ranges between the 'i-th' and 'i+1-th' elementary motion units. Therefore, an expression for g_i can be formulated [13].

$$g_i = f(\overrightarrow{l_{\iota+1}}) - f(\vec{l_\iota}) \tag{4}$$

In different real-life situations, there can be three values for g_i.g_i takes the value 0 when neither the front side nor the back side intersects with the obstacle. It assumes the value 1 when the front side intersects the obstacle and -1 when the back side intersects the obstacle.

5. Conclusion

Above, various path planning algorithms have been introduced. Now, a question arises: which type of path planning algorithm should be chosen? This question is indeed challenging.

In the case of the Intelligent Bionic Algorithm, it possesses the ability to generate a path and adapt it as needed during operation. Essentially, it has memory and draws inspiration from ant behavior. On the other hand, other path planning algorithms rely on mathematical models to determine the most efficient route for the sweeping robot to follow. When comparing these two categories of path planning algorithms, each has its own set of advantages and disadvantages.

The Intelligent Bionic Algorithm requires time for planning and selecting the optimal path. However, it significantly reduces the occurrence of accidents once the plan is in place. On the other hand, algorithms using mathematical models can quickly identify the closest path, but the potential risk of accidents remains a concern, especially when unforeseen emergencies arise that neither the robot nor humans can predict.

In summary, it is essential to continually compare different types of path planning algorithms to find one that meets the specific requirements and needs of users.

To conclude, the optimal approach is to learn from each path planning algorithm and strive to identify the most suitable one. In summary, this covers the topic of path planning algorithms and their implementation in sweeping robots. In these aspects, there is still room for improvement in the

DOI: 10.54254/2755-2721/41/20230716

performance of sweeping robots. Looking ahead, continuous comparisons of different path planning algorithms and ongoing enhancements to the algorithms and robot functions are necessary for the future.

Authors Contribution

All the authors contributed equally and their names were listed in alphabetical order.

References

[1] Tan Dingzhong, Wang Qiming, Li Jinshan, Li Lin. (2004). Research and Development Status of Cleaning Robot.

[2] Cheng Qian, Gao Song, Cao Kai, Chen Chaobo. (2019). Path Planning of Mobile Robot Based on PRM Optimization Algorithm. Journal of Computer Applications and Software, 37(12).

[3] N. Jasika, N. Alispahic, A. Elma, K. llvana, L. Elma and N. Nosovic,"Dijkstra's shortestpath algorithm serial and parallel execution performance analysis," 2012 Proceedingsof the 35th International Convent.

[4] Bai Zhiqiang, Xin Zhou, Zhang Xueqi. (2022). Research on Robot Path Planning Based on Improved Traditional RRT Algorithm. Machine Building & Automation, pp. 177-179.

[5] Wang Hao, Fang Lu, Zhuang Kui, et al. (2021). Path Planning of Sweeping Robot Design based on Genetic Algorithm. China-Arab States Science and Technology Forum, 4(3).

[6] Yan Qin, Tian Zhumei, Ren Guofeng, et al. (2020). Path Planning of Intelligent Sweeping Robot Based on Genetic Algorithm. Journal of Science, 40(3), pp. 5.

[7] Zhang H., Hong W., and Chen M. (2019). A Path Planning Strategy for Intelligent Sweeping Robots. IEEE International Conference on Mechatronics and Automation (ICMA), Tianjin, China, pp. 11-15.

[8] Sen K. and Liqiang Z. (2019) A Path Planning Algorithm for Sweeping Robot Based on Improved Neural Network. The 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, pp. 359-362.

[9] Ge B., Hu S. and Zheng P. (2020). Research on Full Traversal Path Planning Based on Improved Reciprocating Algorithm. IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, pp. 922-926.

[10] Wang Z., Xie H., Lin Z., Wen T., Guo C. and Chen H. (2020). The Robot Path Planning Algorithm In Indoor Environment. The 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, pp. 5350-5355.

[11] Luo B., Huang Y., Deng F., Li W. and Yan Y., (2021). Complete Coverage Path Planning for Intelligent Sweeping Robot. IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC), Dalian, China, pp. 316-321.

[12] Hasan K. M., Abdullah-Al-Nahid and Reza K. J. (2014) Path Planning Algorithm Development for Autonomous Vacuum Cleaner Robots. International Conference on Informatics, Electronics & Vision (ICIEV), Dhaka, Bangladesh, pp. 1-6.

[13] P. Bogdan, "Diikstra algorithm in parallel- Case study" Proceedings of the 2015 16th international Carpathian Control Conference (ICCC), Szilvasvarad, Hungary, 2015, pp.0-53.