Research on an AGV path planning method

Zhihui Chen[1], Xiaoyan Chen*, Shiming Wang

Tianjin University of Science and Technology, China;

E-mail: 1594838831@qq.com

www.tust.edu.cn

Abstract

AGV is an acronym for Automatic Guided Transport Vehicle. At present, the key technologies of AGV mainly include navigation and positioning technology, path planning technology, multi-AGV coordinated control technology and multi-sensor information fusion technology, etc. This paper studies A* algorithm in path planning. The A* algorithm can find the shortest path between two points. It mainly studies the principle of A* algorithm and simulates it in Matlab. By comparing the path length of ant colony algorithm and A* algorithm in Matlab grid graph, the excellent performance of A* algorithm is proved.

Keywords:AGV, Path planning, Ant colony algorithm, A algorithm*

1. AGV Introduction

AGV (Automated Guided Vehicle) refers to a transport vehicle equipped with electromagnetic or optical automatic navigation devices, capable of driving along a prescribed navigation path, with safety protection and various transfer functions.

AGV can be divided into two types according to the guidance method. One is the guidance of a fixed path, and the other is the guidance of a free path. AGV can also be divided into loading type, assembly, towing, fork and suspension types according to the applications. The earliest AGV was transformed from a tractor-type excavator in 1953 by Barrett Electronics Company of the United States. It can be automatically and conveniently connected to the logistics system, which improves production efficiency to a certain extent. Until the beginning of the 21st century, some Western countries have developed AGV systems with network guidance, laser guidance technology and composite guidance technology.

2. AGV Path Planning

Path planning means that the mobile robot searches for an optimal or suboptimal path from the starting position to the target position according to a certain performance index (such as distance, time, etc.)[1]. According to the degree of grasp of environmental information, path planning can be divided into global path planning based on prior information and local path planning based on sensor information.

Global path planning includes environmental modeling method, probabilistic road map method, Dijkstra algorithm, A* algorithm (A-star algorithm) and fast search random tree method.

Local path planning mainly includes artificial potential field method, genetic algorithm, and DWA (Dynamic Window Algorithm).

3. A* Algorithm

3.1. *A* algorithm formula*

The A* algorithm is essentially a heuristic algorithm[2]. The A* algorithm combines the strengths of the breadth first search algorithm and the Dijkstra algorithm. According to the evaluation function formula, evaluate each position that the robot can reach, and find the optimal position from it. Then the optimal path can be found. The A* algorithm is commonly used in npc moves in various games.

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

In Eq.(1), F(n) is a function to calculate the cost from the starting point to the ending point. G(n) represents the actual cost from the starting node to the node where it is. H(n) represents the estimated cost from the node where it is to the end. The key to the shortest path is to choose the cost function, so it is considered the most important part. In the A* algorithm, because the Manhattan distance is relatively simple, the Manhattan distance formula is usually used as the distance estimation algorithm.

3.2. *A* algorithm specific steps*

Simplify the search area into a set of quantifiable nodes.

STEP1: Put the starting point S into the Open list as the node to be viewed.

STEP2: Search for nodes in eight directions near the starting point S (except for obstacles), put these nodes in the Open list, and then make S node the parent node.

STEP3: Move the starting point S from the Open list to the Close list.

STEP4: Compare the node X with the lowest F value in the current Open list, and move the X node from the Open list to the Close list.

STEP5: Search for all reachable nodes around X node.

(1) If these nearby searched nodes are still not in the Open list, add these nodes to it, calculate their F value, and then set the parent node of these nodes.

(2) If these nearby searched nodes are in the Open list, calculate the path from S to the target node through X point. Determine whether F is higher or lower than before.

If the F value is higher than before, it means that the cost of the new path is higher, and there is no need to change it.

If the F value is lower than before, change the parent node to square X, and calculate the F value of the nodes around the X node again, and repeat the cycle

STEP6: Repeat the above search steps until the target node is found

STEP7: End judgment. Finally, if the target node appears in the Open list, it means that a suitable path has been found.The algorithm flow chart is shown in the Fig1.

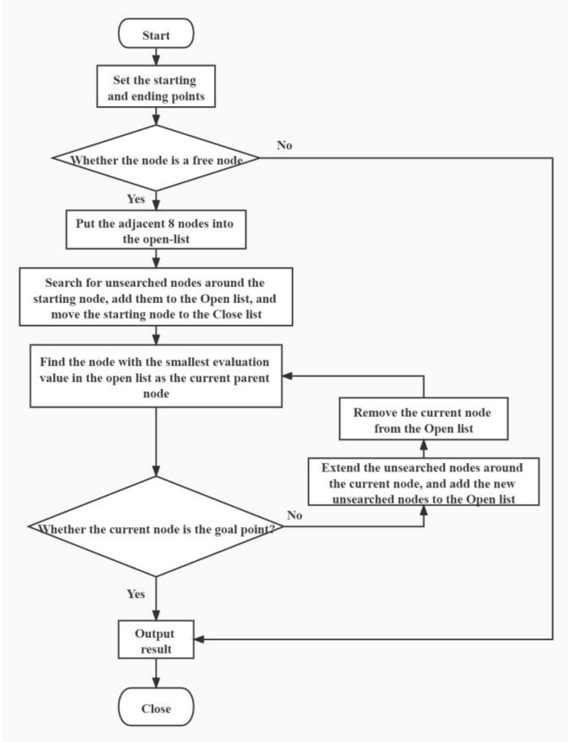


Fig.1. Flow chart of A* algorithm

4. Simulation on Matlab

In this paper, the A* algorithm is simulated on Matlab. Because all the researches in this paper do not need to consider the height of obstacles on a two-dimensional plane, the grid method is adopted for modeling. If there is an obstacle, it is "1" and if there is no obstacle, it is "0".As shown in Eq.(2).

$$\mathrm{cell}(x, y) = \begin{cases} 1 & \text{occupied grid} \\ 0 & \text{free grid} \end{cases} \tag{2}$$

According to the designed two-dimensional space environment, it is ensured that the robot can move freely in the preset environment map. Use Cartesian coordinate system to express the designed workspace in Matlab.

Construct a grid map with an environment of 20×20 and randomly generate 160 obstacles in the map, fix (1, 1) as the starting point, (19, 19) as the end point and use A* algorithm to simulate in Matlab. The simulation results are shown in Fig 2.

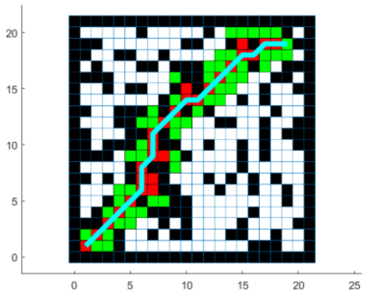


Fig.2. Simulation example of A* algorithm

Next, in order to compare the advantages of the A* algorithm. Compare the path length between A* algorithm and ant colony algorithm.

AG (Ant Colony Algorithm) is an optimization algorithm that simulates ants looking for food[3]. The principle of the ant colony algorithm is that each ant will secrete pheromone inversely proportional to the length of the path on the road on which it has crawled after exiting the hole to tell the subsequent ants. The pathfinding ends when they search for the target node. In the end, what the ant colony finds is the optimal foraging path.

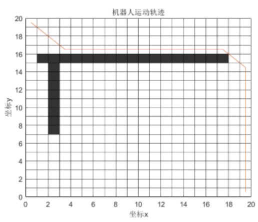
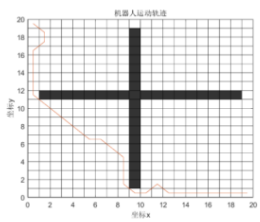
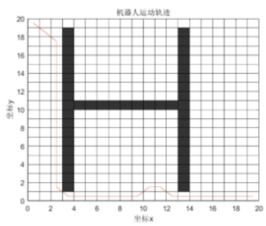
In the Matlab simulation, the basic parameters of the ant colony algorithm should be set first, including the pheromone intensity Q and the evaporation coefficient ρ. This paper adopts the ant colony algorithm to set the number of iterations as K=100; the number of ants M=50; the importance of pheromone Alpha=1; the importance of heuristic factor Beta=7; the pheromone evaporation coefficient Rho=0.3; the pheromone increase intensity coefficient Q =1. The parameter settings are shown in Fig 3.

```
Tau=ones(MM*MM,MM*MM);     % Tau initial pheromone matrix
Tau=8.*Tau;
K=100;                     %Number of iterations
M=50;                      %Number of ants
S=1;                       %Starting point of the shortest path
E=MM*MM;                   %Destination point of the shortest path
Alpha=1;                   % Alpha Parameter indicating the importance of pheromone
Beta=7;                    % Beta A parameter indicating the importance of the heuristic factor
Rho=0.3 ;                  % Rho Pheromone evaporation coefficient
Q=1;                       % Q Pheromone increase intensity coefficient
```

Fig.3. Ant Colony Algorithm Parameters

Establish two identical obstacle maps in the grid map and then use the A* algorithm and the ant colony algorithm to simulate the results as shown in the table 1and table 2 below.

Table 1 Simulation results

	Model 1	Model 2	Model 3
Ant Colony Algorithm			
A* Algorithm			

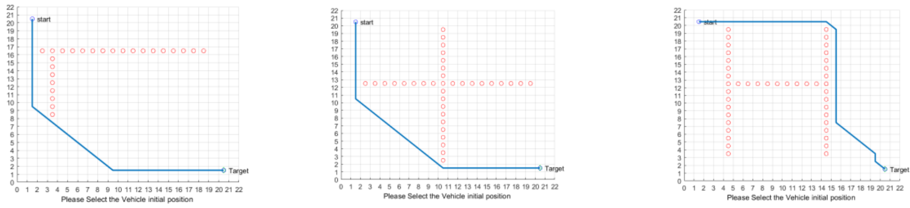


Table 2 Length comparison

Algorithm	Ant colony	A*	Short path algorithm
Model 1	37.272	33.3137	A*
Model 2	38.555	32.7279	A*
Model 3	38.772	34.4853	A*

The three sets of experimental data all show that the shortest path of the A* algorithm is shorter than that of the ant colony algorithm, so the A* algorithm is better than the ant colony algorithm.

5. Conclusion

The key issue of intelligent robot research all over the world is the path planning problem of mobile robot from beginning to end. The A* algorithm is already one of the commonly used algorithms for global path planning of mobile robots, and the A* algorithm is also effective in scientific and technological applications. But the A* algorithm still has many problems. Firstly, the current research of A* algorithm is mostly an ideal flat two-dimensional plane, and obstacles can only go around. In reality, this is not always the case[4]. When encountering obstacles that can overcome the past, the A* algorithm is not so good. Therefore, it is necessary to improve the A* algorithm so that the A* algorithm can be applied in more practical scenarios. Secondly, in order to prevent the A* algorithm from falling into the local optimal solution problem, a large number of search nodes will be carried out, and there will be some unrelated nodes among these nodes. Therefore, the balance between the optimality of the algorithm path and the number of irrelevant search nodes searched in the search process is a problem to be studied and solved. Thirdly, although the performance of the A* algorithm is relatively good, many scenarios have higher requirements for problem solving. Sometimes the A* algorithm alone cannot solve the problem particularly well[5], so it is necessary to integrate the A* algorithm with other algorithms. Solve more practical problems better.

References

1. C. Wang and J. Mao, "Summary of AGV Path Planning," 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), 2019, pp. 332-335.
2. C. Ju, Q. Luo and X. Yan, "Path Planning Using an Improved A-star Algorithm," 2020 11th International Conference on Prognostics and System Health Management (PHM-2020 Jinan), 2020, pp. 23-26. 2.
3. Q. Su, W. Yu and J. Liu, "Mobile Robot Path Planning Based on Improved Ant Colony Algorithm," 2021 Asia-Pacific Conference on Communications Technology and Computer Science (ACCTCS), 2021, pp. 220-224.
4. A. Candra, M. A. Budiman and K. Hartanto, "Dijkstra's and A-Star in Finding the Shortest Path: a Tutorial," 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA), 2020, pp. 28-32.
5. Xiang Liu and Daoxiong Gong, "A comparative study of A-star algorithms for search and rescue in perfect maze," 2011 International Conference on Electric Information and Control Engineering, 2011, pp. 24-27. : .

Authors Introduction

Mr. Zhihui Chen

He received his bachelor's degree from the school of electronic information and automation of Tianjin University of science and technology in 2021. He is acquiring for his master's degree at Tianjin University of science and technology.

Prof. Dr. Xiaoyan Chen

She is professor of Tianjin University of Science and Technology, graduated from Tianjin University with PH.D (2009), worked as a Post-doctor at Tianjin University (2009.5-2015.5).

Mr. Shiming Wang

He graduated from Harbin Institute of Technology with a Ph.D in Control Science and Engineering, and is now the Director of the Automation Department of Tianjin University of Science and Technology.