Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Coordinated Path Planning Algorithm for Multi-Robot in Intelligent Warehouse *

Xin Chen, Yanjie Li, Lintao Liu
School of Mechanical Engineering and Automation
Harbin Institute of Technology, Shenzhen
Shenzhen, Guangdong Province, China
autolyj@hit.edu.cn

***Abstract* - In this paper, we study the coordinated path planning problem of multi-robot in intelligent warehouse. We first describe the problem of the multi-robot path planning and analyze the basic structure of the intelligent warehouse system. Then we discuss the adjustment of multi-robot path planning algorithm from two aspects. We reserve the path information of the continuous time period in the reservation table of the Windowed Hierarchical Cooperative A* (WHCA*) algorithm, and analyze whether to reuse the path information each time the algorithm is invoked. Then on the basis of this, we introduce the turning factor as a measure when expanding a path, and hope to find the path with fewer turns to reduce the travel time of the robot. Finally, we do some experiments and make the analysis. The results show that the methods we used can improve the path planning efficiency of the robot and increase the order throughput of the intelligent warehouse system.**

Index Terms – multi-robot, coordinated path planning, adjusted reservation table, turning factor, intelligent warehouse.

I. INTRODUCTION

In recent years, with the vigorous development of the e-commerce industry, people are becoming more and more urgent for the pursuit of high-quality "online shopping" experience. Compared with traditional warehouses, more efficient intelligent warehouse systems play an increasingly important role in the supply chain. The first intelligent warehouse system is KIVA system [1], as shown in Figure 1(d). In the intelligent warehouse system, after each robot receives the corresponding order task, it needs to plan the path from the current position to the target pod, and then from the pod position to the working station. These paths are required to not conflict with the paths of other robots, and there is no deadlock between the robots during driving. Of course, we hope that this path is the shortest to improve the efficiency of the warehouse. Although the intelligent warehouse system has many advantages and market demands, the key to its success lies in whether multiple robots can operate in an effective and coordinated manner, avoiding mutual interference and conflict between robots and making them have the same purpose in task execution. Therefore, the path planning problem is an important research direction.

The multi-robot path planning problem refers to finding a collision-free path from a starting point to a target point for a set of robots in a known environment [2]. In the case of completing their respective tasks, we hope that the robot will reach the target point for the shortest time and the driving path is optimal. The path planning problem has certain research and application in computer games [3], drone formation [4], automated straddle carrier and intelligent warehouse [1], see Figure 1. This article mainly studies the coordinated path planning problem among multiple robots in the intelligent warehouse system.

For the multi-robot path planning problem, many scholars have studied this in recent years, and there are many optimal and sub-optimal solutions. In the suboptimal solution, a standard admissible method is the A* algorithm [5]. Based on A* algorithm, a new search-based sub-optimal solution is the Hierarchical Cooperative A*(HCA*) proposed by [6]. In this algorithm, the map is expanded into a three-dimensional space-time map, and the path of the robot is planned according to some rules that have been defined. The planned paths are stored in the global reservation table, and subsequent robots avoid the occupation of the same location based on the previous information in the reservation table when they are performed path planning. A further enhancement, the Windowed-HCA*(WHCA*) algorithm limits the search process to a time window after which the robot will be igno-

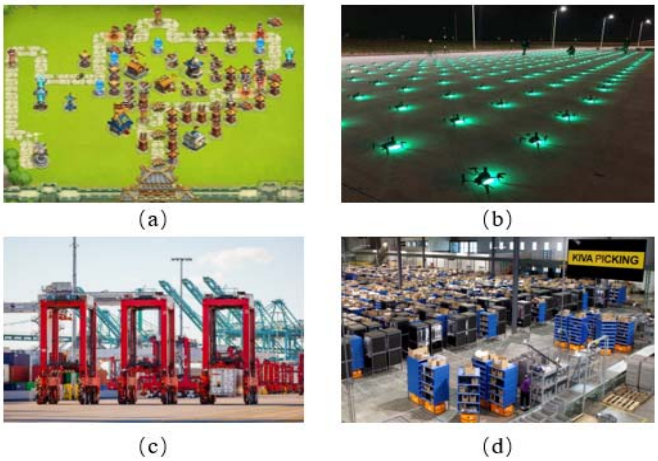

Fig. 1 Research and application of path planning.

* This work is supported by Shenzhen basic research program JCYJ20180507183837726, JCYJ20170811155028832 and National Natural Science Foundation (NNSF)of China under Grant U1813206 and 61977019.

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

-red. After that, in [7] it focuses the reservation windows on areas and time-steps where conflicts between agents occur, and reservations are made most of the time only by a single agent. There are other sub-optimal methods such as Biased Cost Pathfinding (BCP) [8] and Flow Annotation Replanning (FAR) [9]. The optimal solution to find multi-robot path planning problems proved to be NP-hard [10] [11]. Some optimal methods have also been studied in the literature, such as the conflict-based search (CBS) proposed in [12]. CBS is a two-level search algorithm. At the high-level search, algorithm find conflicts between robots in the constraint tree and add constraints to them. At the low-level search, only one robot is searched for the path at a time, and the robot's path is updated according to the newly added constraints. Other optimal methods based on A* algorithm are Operator Decomposition (OD) [13], Independence Detection (ID) [14] and M* [15]. In our work, we modify the reservation table in the WHCA* algorithm, so that we save the path information of the continuous time period in the reservation table. At the same time, we use the turning factor as an evaluation index when expanding the path.

The rest of the paper is organized as follows. In section II, we describe the problem of multi-robot path planning and the basic structure of the intelligent warehouse system, discuss the constraints and evaluation indicators in the real environment. In section III, we introduce the algorithm we have used for different problems and gives corresponding explanations. Section IV introduces our experimental situation and analysis the experimental results, and last section summarizes the whole work.

II. Problem And System Description

The problem of the multi-robot path planning can be expressed as $\{Map, ROB, S, G\}$. The Map is the environment that robots are located in, and there are one or more obstacles in it. We assume every location of the Map is marked by a QR code. The set ROB contains N robots, which is expressed as $ROB=\{r_1, r_2, ..., r_N\}$. S is a set of $\{s_1, s_2, ..., s_N\}$, which denotes the starting points of each robot. G is a set of $\{g_1, g_2, ..., g_N\}$, which denotes the target points of each robot. The input of every robot is $< Map, s_i, t_i >$, and the output of every robot is a path that leads the robot to complete their tasks. The robot needs to plan a collision-free path from the starting point to the target point. If a feasible path is found for all robots, then such a solution is called a feasible solution. While finding the feasible solution, we hope that some of the evaluation indexes will be optimal. If necessary, we will add constraint to the robot.

In the intelligent warehouse system, it generally consists of storage area, picking stations, replenishment stations, pods, and robots. The intelligent warehouse system is shown in the Figure 2. The items are stored in the movable pod, and the pods are randomly placed in the storage area. There are picking stations and replenishing stations on both sides of the storage area, and the robot transports the corresponding pods

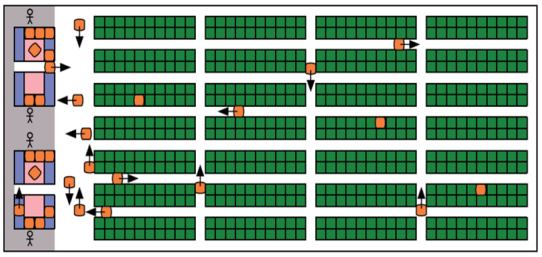

Fig. 2 Intelligent warehouse system.

to the station according to the order tasks. In general, we want pods that are often executed for picking and replenishment operations to be placed closer to the station.

There are generally multiple decision problems in intelligent warehouse system. For example, task assignment, path planning, resource allocation, etc. In the task assignment problem, the central controller transmits the received order task information to the task assignment controller. The task assignment controller assigns a new task to the robot for execution according to the location of the robot in the warehouse and the completion of the task through a certain assignment method, for example an auction-based method [16]. The path planning problem is roughly the same as that discussed earlier. In the resource allocation problem, it involves the sub-problems of pod selection, pod storage, robot allocation and so on [17]. If a robot has just performed the picking task and the pod needs to be replenished immediately, then the robot can simply carry the pod from the picking station to the replenishing station. Therefore, the order of execution of the replenishment and picking tasks, and the selection of the required pods will affect the completion of the final task to some extent. The control block diagram of the intelligent warehouse system is shown in the Figure 3.

During the actual driving process, the robot will be given some constraints such as maximum driving speed, whether it can drive in both directions, and so on. The robot cannot occupy the same position at the same time. For an aisle with a fixed direction of travel, the robot can only travel in the specified direction. In the actual warehouse environment, we also consider the acceleration rate and deceleration rate of the robot. The output of the system is the travel path of the robot and the completion of the order task. The objective function for evaluating path planning usually include *makespan* [11], *planning time*, *travel distance*, etc.

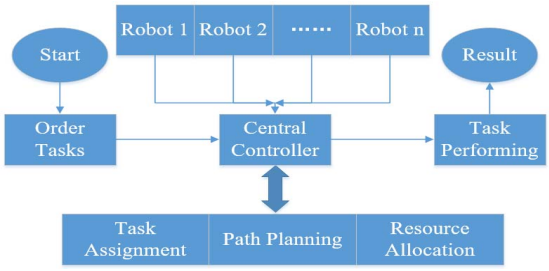

Fig. 3 Control block diagram of the intelligent warehouse system.

III. ALGORITHM

In this section, we will discuss some of the approaches we took when studying multi-robot path planning problems.

A. *Consider Actual Movement*

In the previous problems of multi-robot path planning research, most methods are on the grid map environment, and the robot moves in a single step, and the cost per step is the same. The A* algorithm is suitable for the path planning problem of a single robot. When multiple robots perform path planning, collisions and conflicts will occur between multiple paths. Therefore, based on the basic A* algorithm, [6] proposed the WHCA* method, which has achieved very good results in the path planning problem in the game field. We can also apply this algorithm to multi-robot path planning problems in intelligent warehouse.

The WHCA* algorithm takes into account the paths that other robots have planned during the search. The main idea of this method is to expand the two-dimensional space map into a three-dimensional space-time map by increasing a time variable. The algorithm records the path information of the robot through a global reservation table, so as to avoid occupying the same location at the same time in the subsequent robot planning path process. In order to improve the efficiency of path planning and reduce long-term reservations for a certain location point, the depth of the space-time search is limited by adding a time window.

In an actual warehouse environment, the robot tends to run continuously during operation and also satisfies some kinematic conditions. For example, if a robot travels a long distance, it will experience acceleration, uniform speed, and deceleration. If a robot goes to the picking station to perform the picking task immediately after completing the replenishment task at the replenishment station, it may only have the acceleration and deceleration process. In these cases, the path planning process of the robot is often measured in continuous time. A safe interval path planning (SIPP) algorithm is proposed in [18], which is a single robot continuous time path planning algorithm. SIPP is a path planning algorithm based on A* algorithm. It combines multiple collision-free continuous time steps into a safe interval and searches for the state represented by (configuration, safe interval) pair in the search space. SIPP theoretically provides guarantee of optimality and completeness. Therefore, under the condition of continuous time, we can make a certain adjustment to the reservation table, and record the continuous time interval reserved for each position. Each interval has a start time and an end time. Only robots with the correct number in this interval can go through this position. The adjusted reservation table is shown in the Figure 4.

After modifying the reservation table in the WHCA* algorithm, we discuss the use of existing path information when invoking the algorithm. The algorithm each time it is invoked will plan paths from start point to target point for each

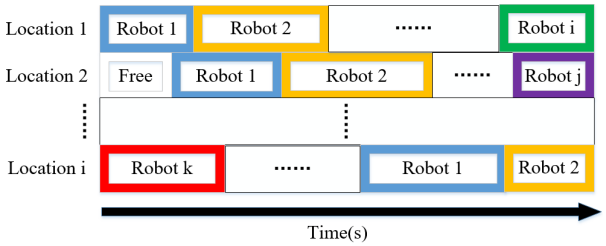

Fig. 4 Adjusted reservation table.

robot that has been assigned task. In the process of other robots performing tasks, if a robot in an idle state is assigned a new task, the algorithm is invoked again to plan the path for the robots. Whether the algorithm reuses the path information in the reservation table at each call is related to the execution efficiency of the algorithm. Intuitively, reusing existing path information will help us reduce the time it takes to calculate a new path. However, the path of the previous robot cannot be updated in real time. It is possible that the robot will miss a chance to drive a more perfect path in the remaining travel process before reaching the target point.

In general, the method of reusing path information is more suitable for the case where the obstacle in the environment is fixed, and the method of not reusing the path information is more suitable for the case where the obstacle in the environment is dynamically changed. Using different methods can have different effects on the overall efficiency of the intelligent warehouse system.

B. *Consider Turning Factor*

As shown in the Figure 5, when the starting point and the target point of a robot are determined, it can have multiple paths with the same distance to reach the target point on the grid warehouse map. When the path of the robot is searched by the A* algorithm, it is expanded randomly. When the robot travels along path 1, there are only 2 times of turning, there are 4 times of turning when traveling along path 2, and there are 5 times of turning when traveling along path 3. Different driving paths will take different driving time. If you choose a path with more turns, it will undoubtedly increase the driving time of the robot. For this reason, we need to improve the evaluation function in the A* algorithm and take the turning factor into account to find a path with fewer turns.

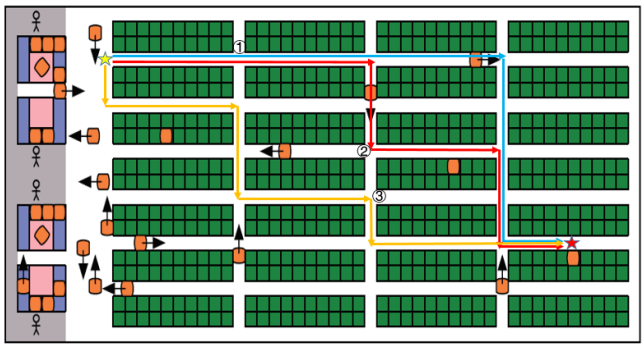

Fig. 5 Paths of the same length with different turns.

When judging whether a turn is generated, we judge by the positional relationship between the current node and its parent node, and the positional relationship between the current node and its neighboring nodes. Two situations in which a path generates a turn are: 1) if the current node has the same x-direction coordinate as its parent but the x-direction coordinates of its neighbor are not the same, or 2) the current node has the same y-direction coordinates as its parent, but the y-direction coordinates of its neighbor are not the same. A turn is generated by satisfying one of the conditions.

We can introduce a turning factor τ as a measure. The coordinate of the current node C is expressed as $(current.x, current.y)$. The coordinate of the parent node P is expressed as $(parent.x, parent.y)$. The coordinate of the neighbor node N is expressed as $(neighbor.x, neighbor.y)$. In the first situation, the schematic diagrams of turning and no turning is shown in the Figure 6, and the turning factor in this case is expressed by the equation (1). In the second situation, the schematic diagrams of turning and no turning is shown in the Figure 7, and the turning factor in this case is expressed by the equation (2). We add a turning factor to the original evaluation function of A* algorithm and expand the path by selecting the node with the smallest evaluation function. The evaluation function that considers the turning factor is shown in the equation (3), where $g(n)$ is the cost function, $h(n)$ is the heuristic function.

$$\tau = \begin{cases} 1 & if \ \tan \dfrac{|neighbor.x - current.x|}{|current.y - parent.y|} \neq 0 \\ 0 & otherwise \end{cases}. \quad (1)$$

$$\tau = \begin{cases} 1 & if \ \tan \dfrac{|neighbor.y - current.y|}{|current.x - parent.x|} \neq 0 \\ 0 & otherwise \end{cases}. \quad (2)$$

$$f(n) = g(n) + h(n) + \tau . \quad (3)$$

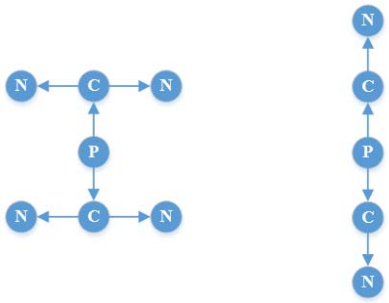

Fig. 6 The schematic diagrams of turning and no turning in situation one.

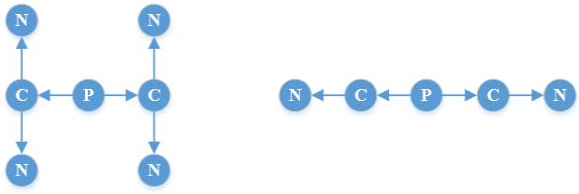

Fig. 7 The schematic diagrams of turning and no turning in situation two.

IV. Experimental Analysis

Since we mainly study the path planning problem, we do not consider the impact of other controller's strategies on the path planning in the experiment, or we adopt a default control strategy for other controllers. We use the WHCA* algorithm that considers actual movement for multi-robot path planning, whose reservation table reserves continuous-time path information of the robot. We analyze from two aspects.

A. The Usage of Path Information

We do the experiment for whether reusing path information. We set up different numbers of robots in the warehouse, and observe the impact of the two methods on the overall path planning effect. The time window of the algorithm is set to 30 seconds. In each simulation, the maximum travel speed of the robot is 1.5m/s, the acceleration rate and deceleration rate are both 1m/s^2, and the robot performs 24 hours of work. The relevant experimental data is shown in the table I and table II. The comparison of the two methods in average path planning time is shown in the Figure 8. From the experimental data, we can see that as the number of robots increases, the average time required to plan the path will increase, and the overall driving distance will also increase. The method of reusing path information is less in overall planning time, planning count, average planning time than the method of not reusing path information. However, the two methods show different effects in the overall driving distance due to the number of robots. When the number of robots is between 5 and 19, the method of reusing the table path information allows the robot to travel a smaller distance than the method of not reusing the table path information in the overall travel distance. When the number of robots is between 20 and 40, the method of reusing the table path information allows the robot to travel a longer distance than the method of not reusing the table path information in the overall travel distance. When the number of robots is between 41 and 50, it may be due to the influence of other decision controllers, and the two methods will alternately lead.

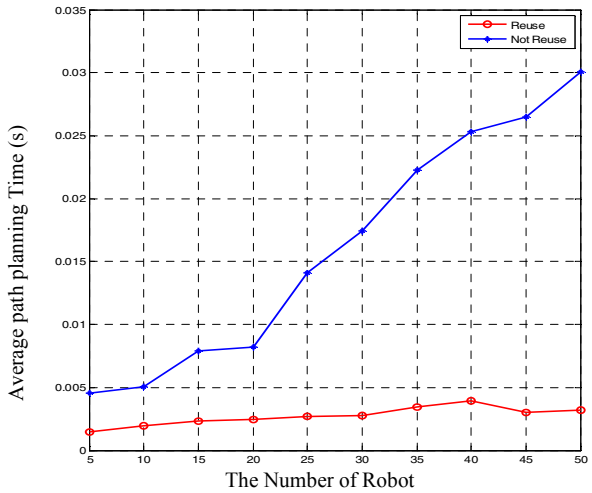

Fig. 8 Comparison of average path planning time between the two methods.

TABLE I
TEST OF REUSING PATH INFORMATION

Robot Number	Index			
	Overall Planning Time(s)	Planning Count	Average Planning Time(s)	Overall Distance Traveled(m)
5	23.87785430	16221	0.0014720334	310470.835
10	55.21246469	28488	0.0019380955	505970.434
15	84.33417589	36222	0.0023282584	628361.679
20	94.05264989	38164	0.0024644338	691357.138
25	106.24174929	39489	0.0026904138	731619.736
30	116.19998649	42275	0.0027486691	781488.137
35	162.54191819	47067	0.0034534157	865781.978
40	193.06734340	49065	0.0039349300	900024.569
45	150.22180419	49623	0.0030272616	905905.698
50	163.53304039	51113	0.0031994412	937513.703

TABLE II
TEST OF NOT REUSING PATH INFORMATION

Robot Number	Index			
	Overall Planning Time(s)	Planning Count	Average Planning Time(s)	Overall Distance Traveled(m)
5	35.91374130	15377	0.0045749033	312893.777
10	138.71865579	27541	0.0050368053	514582.302
15	279.40013110	35513	0.0078675452	635010.643
20	315.57937120	38429	0.0082120110	688489.948
25	556.81646550	39584	0.0140667054	724631.682
30	755.32981119	43244	0.0174666962	776654.984
35	1100.78493229	49393	0.0222862538	859806.599
40	1311.25279369	51818	0.0253049673	896680.720
45	1364.82839769	51585	0.0264578539	909073.240
50	1618.20704609	53818	0.0300681379	935459.071

Generally, we want to obtain a larger order throughput in the intelligent warehouse system, so a faster speed of algorithm is important. But according to the computing power of the modern computer, the calculation increment of the microsecond level is not a big problem. The shorter driving distance in the actual warehouse is the goal we are pursuing, therefore we will adopt different methods according to the number of robots in the warehouse. From the comparison between Table I and Table II we know, in most cases, the method that does not reuse path information when performing WHCA* algorithm will get a shorter distance. However, the method that reuse path information is faster. There is a trade-off between travel distance and planning time.

B. The Impact of Turning Factor

In the case of a single robot with the same starting point and target point, we compare the results of the path planning algorithm that considering turning factor and not consider turning factor. We set up two map environments during the experiment, one is a random obstacle environment, and the other is a regular warehouse environment, in which the influence of turning factors is compared. Figure 9 shows the two planned paths in a random obstacle environment. Figure 10 to Figure 12 show the paths planned by the two algorithms in a regular warehouse environment with different pod conditions. Among them, the blue path is a path that does not consider a turning factor, and the yellow path is a path that considers a turning factor. Since the path is compared in two cases, the two paths generated in the figures may have coverage.

As shown in Figure 9, the path lengths of the two algorithms are the same. The total number of turns for the path planned by the method without considering the turning factor is 13, and the total number of turns for the path planned by the method considering the turning factor is 7, the turning rate has decreased by nearly 50%. Therefore, for the path planning of a single robot, the algorithm considering the turning factor effectively reduces the travel time of the robot.

As shown in Figure 10 and Figure 11, in a regular warehouse environment, the path distances planned by the two algorithms are the same, and the number of path turns are the same. In the comparison of Figure 12, the path planned by the algorithm considering the turning factor has a slightly smaller number of turns than the path planned by the algorithm without considering the turning factor. In the regular warehouse environment of our experiment, the difference between the two methods is not particularly obvious.

In a large-scale warehouse environment, the position of the pod often changes since the replenishment tasks and the picking tasks are continuously in progress. The pods placed in the storage area are transported by robots to the replenishment stations or picking stations for corresponding operations. Thus, the original storage position of pod can be free to travel by other robots. In this case, the algorithm that considers the turning factor are effective and can plan a path with fewer turns. In the condition that the overall pod layout is regular but it is random in the local pod placement, which is more in line

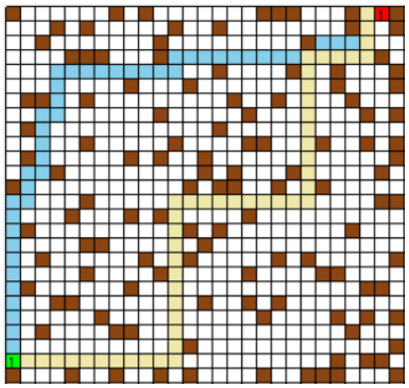

Fig. 9 Path comparison in random obstacle environment.

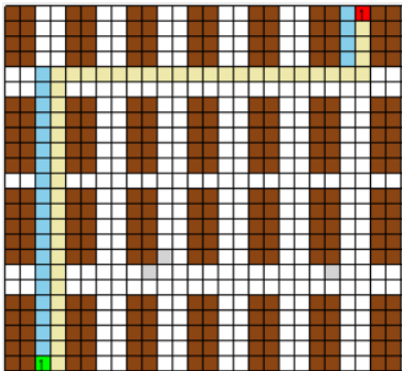

Fig. 10 Path comparison in a regular warehouse environment one.

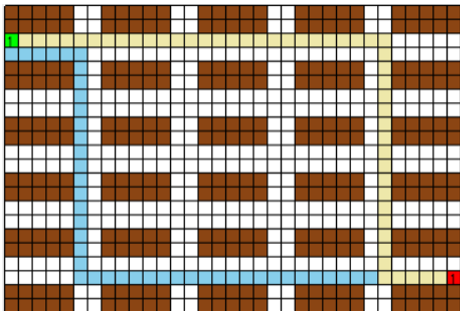

Fig. 11 Path comparison in a regular warehouse environment two.

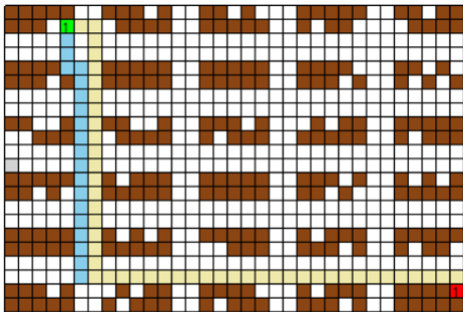

Fig. 12 Path comparison in a regular warehouse environment three.

with the actual situation of the intelligent warehouse, the algorithm that considering turning factor is more effective and can improve the efficiency of the warehouse to some extent. For the method of considering the turning factor, we can generalize from the case of two robots to the case of multiple robots in the future.

V. Conclusion

In this paper, we study the coordinated path planning problem of multi-robot in intelligent warehouse. We describe the problem of the multi-robot path planning and the basic structure of the intelligent warehouse system, and gives a system block diagram. Since the driving process of the robot is often continuous in the actual warehouse environment, it is proposed to make a certain adjustment of the structure of the reservation table in the WHCA* algorithm, and the reservation of the continuous time period of each position is used as the storage information. By experimenting with whether to reuse the path information of the robot each time the algorithm is invoked, we know that in most cases dynamically adjusting the planned path each time the algorithm is invoked will allow the robot to travel a shorter distance. By introducing a turning factor as a metric, we reduce the number of turns of the path and decrease the travel time while ensuring that the distance traveled by the robot does not increase. Through these methods, we make the intelligent warehouse system run more efficiently.

References.

[1] P. R. Wurman, R. D' Andrea, and M. Mountz, “Coordinating hundreds of cooperative autonomous vehicles in warehouses,” *AI Magazine*, vol. 29, no. 1, pp. 9-9, Spring 2008.

[2] L. Cohen, T. Uras, and S. Koenig, “Feasibility Study: Using Highways for Bounded-Suboptimal Multi-Agent Path Finding,” *Eighth Annual Symposium on Combinatorial Search*, 2015.

[3] P. Yap, N. Burch, Robert C. Holte, and J. Schaeffer, "Any-Angle Path Planning for Computer Games." *AAAI Conference on Artificial Intelligence & Interactive Digital Entertainment* AAAI Press, pp. 201-207, 2011.

[4] M. Soto, P. Nava, and L. Alvarado, "Drone Formation Control System Real-Time Path Planning." *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, pp. 2007-2770, May 2007.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100-107, 1968.

[6] D. Silver, “Cooperative Pathfinding,” *AIIDE*, vol. 1, pp. 117-122, 2005.

[7] Z. Bnaya, and A. Felner, “Conflict-Oriented Windowed Hierarchical Cooperative A*,” *IEEE International Conference on Robotics & Automation (ICRA)*, pp. 3743-3748, 2014.

[8] A. Geramifard, P. Chubak, and V. Bulitko, “Biased Cost Pathfinding,” *AIIDE*, pp. 112-114, 2006.

[9] K-H. C. Wang and A. Botea, “Fast and Memory-Efficient Multi-Agent Pathfinding,” *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS)*, pp. 380-387, 2008.

[10] J. Yu and S. M. LaValle, “Structure and Intractability of Optimal Multi-Robot Path Planning on Graphs,” *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, pp. 1443-1449, 2013.

[11] P. Surynek, “An Optimization Variant of Multi-Robot Path Planning is Intractable,” *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligences*, pp. 1261-1263, 2010.

[12] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, “Conflict-based search for optimal multi-agent pathfinding,” *Artificial Intelligence*, vol. 219, pp. 40-66, 2015.

[13] T. S. Standley, “Finding Optimal Solutions to Cooperative Pathfinding Problems,” *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligences*, pp. 173-178, 2010.

[14] T. S. Standley and R. E. Korf, “Complete Algorithms for Cooperative Pathfinding Problems,” *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pp. 668-673, 2011.

[15] G. Wagner and H. Choset, “Subdimensional expansion for multirobot path planning,” *Artificial Intelligence*, vol. 219, pp. 1-24, 2015.

[16] B. P. Gerkey and M. J. Matarić, “Sold!: auction methods for multi-robot coordination,” *IEEE Transaction on Robotics and Automation*, vol. 18, no. 5, pp. 758-768, 2002.

[17] J. J. Enright and P. R. Wurman, “Optimization and Coordinated Autonomy in Mobile Fulfillment Systems,” *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pp. 33-38, 2011.

[18] M. Phillips and M. Likhachev, “SIPP: Safe Interval Path Planning for Dynamic Environments,” *IEEE International Conference on Robotics & Automation (ICRA)*, pp. 5628-5635, 2011.