EAI Endorsed Transactions
on Scalable Information Systems Research Article **EAI.EU**

Dynamic Weighted and Heat-map Integrated Scalable Information Path-planning Algorithm

Shuhui Bi[1 2], Zhihao Li[1], Mackenzie Brown[3], Yuan Xu[1], Lei Wang[2 4,*]

[1]School of Electrical Engineering, University of Jinan, Jinan, 250022, China
[2]Shandong Guige Intelligent Technology co., Ltd, Jinan, 250000, Shandong, China
[3]School of Engineering, Edith Cowan University, Joondalup WA 6027, Australia
[4]HRG (Shandong) Intelligent Equipment Research Institute, No. 1268 Gongye 4 Road, Jinan, 250000, Shandong, China

Abstract

Smart storage is widely used for its efficient storage and applications. For making dynamic decisions when robots conflict and eliminating robot conflicts and improving efficiency from a global perspective, path-planning Algorithm will be analyzed and improved by integrating dynamic weighted and heat-map algorithm based on the scalable information of multi-robot in this paper. Firstly, a small storage grid model applicable to a variety of storage modes is established. Second, in order to solve the frontal collision problem of robots, an improved reservation table is established, which greatly reduces the storage space occupied by the reservation table while improving the operation efficiency; the A* algorithm is improved to achieve the purpose of avoiding vertex conflict and edge conflict at the same time; dynamic weighting table is added to solve the multi-robot driving strategy of intersection conflict and ensure that the most urgent goods are out of the warehouse firstly; the heat map algorithm is appended to reasonably allocate tasks, avoiding congested areas and realizing the dynamic assignment of tasks. Finally, the simulation was done by the proposed path planning method, the average transportation time was reduced by 14.97% comparing with the traditional path algorithm.

Received on 24 June 2022; accepted on 28 September 2022; published on 05 October 2022

Keywords: Improved A* algorithm, Improved Reservation Form, Dynamic Weighted Table, Heat Map Algorithm

doi:10.4108/eetsis.v9i5.1567

1. Introduction

The efficiency evaluation of the intelligent storage system mainly depends on the overall transport speed of the order task, and the transport speed mainly depends on the path planning algorithm for the shortest path to find the best, collision avoidance and reasonable allocation of orders. With the expansion of the scale of the intelligent storage system, the use of robots for transporting goods increases, and the avoidance of collisions and the reasonable allocation of orders become critical factors affecting the overall efficiency of the warehouse compared to the requirements for the shortest path.

As there is more than one robot in the warehouse, multiple robots will cause conflicts with each other, and the shortest path may not be the best route at this time. If multiple robots gather at the warehouse intersection, they need to pass the intersection in order to avoid collision, resulting in longer waiting time, which affects at efficiency instead.At the same time, if the materials are stored randomly, it will greatly reduce the efficiency of the warehouse, make it difficult to manage, and cause more chaos when the robots transport the materials out of the warehouse. The existing methods try to reduce the conflict range from the path planning, but cannot optimize the conflict from the global perspective, and can only qualitatively reduce the conflict but not quantitatively eliminate it.

*Corresponding author. Email: wangleisd@hitrobotgroup.com

The paper [1] proposed an ant colony path planning algorithm that can obtain higher quality globally optimal paths with a flexible trade-off between satisfactory solutions and the number of iterations. The paper [2] proposed an ant colony optimization-based path planning algorithm for robot clusters in dynamic environments. The paper [3] proposed a global path planning method based on path selection, which first used the ability of the swarm search optimization algorithm to cover multiple local optimal solutions to generate multiple paths at a time; then proposed two selection algorithms for multiple paths, which re-evaluate the intersecting paths and select the better path when passing the path intersection, and achieved path optimality; and finally quickly selected the appropriate path by heuristic search to path, which reuse the original search results, thus avoiding secondary planning. The paper [4] proposed a robot cluster path planning method based on the firefly method, which used the firefly social behavior to optimize the group behavior. The paper [5] investigated an improved bidirectional A* algorithm to reduce the path length of SAR UAV planning and the time required to plan the path, which can effectively improve the efficiency and safety of SAR UAVs, with important theoretical significance and application value.

When multiple robots are running, conflicts are bound to occur due to the interleaved routes of operation. For these conflicts, there are various heuristic rules that can be employed, such as setting up priorities for avoidance, waiting for robots with fewer tasks, or using channel protocols to avoid conflicts of robots within a channel [6]. The paper [7] improved the algorithm operation efficiency by improving the ant colony algorithm and improving the algorithm operation efficiency by the internal dynamic inertia weight and pheromone update strategy to obtain a global path planning algorithm that outperforms the ant colony algorithm in path planning results in different maps. The paper [8] proposed to build a flexible warehouse model, and proposed an improved A* algorithm and combined it with an improved adaptive genetic algorithm to achieve path planning for batch picking operations. In the paper [9], the shortest path for each robot to complete the task goal was planned and formed into a reservation table by an improved Q-Learning algorithm to reduce the standby state of robots without tasks and balance the workload among robots. In line planning, it is easy to fall into the local optimum problem by using a particular algorithm alone, and the paper [10] proposed a hierarchical path planning method based on a hybrid genetic particle swarm optimization algorithm, which can find the optimal path quickly and efficiently by avoiding obstacles in a complex environment.

Real-time scheduling is performed when the system is actually running, which requires high timeliness for scheduling. Therefore, some rules were mostly used to deal with various problems encountered in the system operation [11], and dynamic path planning was performed to avoid conflicts between robots [12] by the obtained current state of the robots in order to coordinate conflicts; there are also some problems related to system inputs, such as the appearance of urgent orders, which are assigned in preference to other orders, and in preference to other tasks being assigned to robots for completion [13]. In addition, intelligent algorithms can be used to obtain better task assignment results by iterative optimization, such as simulated annealing algorithms [14]. The paper [15] studied the task assignment problem of multiple logistics robots in an intelligent warehouse system, established a task assignment model using an improved ant colony algorithm[16] to solve it, and verified the effectiveness of the algorithm[17].

Based on the above research, the reservation table to solve the problem of path conflict is improved, which greatly reduces the storage space occupied by the reservation table and improves the operation efficiency. Aiming at the multi-robot driving strategy problem of intersection conflict, a dynamic weighted table method was proposed to ensure the most urgently needed goods to be sent out of the warehouse in priority. Aiming at the problem of reasonable assignment of tasks, the heat map algorithm is introduced into the task scheduling strategy to achieve dynamic assignment of tasks. Finally, the effectiveness of the model and algorithm is proved by comparative experiments.

The remaining construction of this study is outlined as follows. Section 2 established a raster model for the warehouse map, built an improved traditional reservation table; then innovatively proposed a dynamic weighted table and the heat map method; finally completely introduced the operation logic of the whole path planning algorithm constructed in this paper. Section3 tested the efficiency improvement of the A* algorithm with the addition of dynamic weighted table and heat map algorithm compared with the traditional A* algorithm through two sets of comparison experiments respectively.

2. Algorithm Design

2.1. Environment Construction

Raster Map Construction. This study first constructs a standardized warehouse model with a length of 25 meters and a width of 26 meters. The advantage of a standardized warehouse is that it can adapt to the needs of different companies in different scenarios. As shown in Figure 1, the picking table is arranged in the leftmost position of the warehouse, and each picking table occupies a space of 1m×2m. The black area is the location of the warehouse shelves. The warehouse shelves are arranged in the form of shelf groups in the warehouse, each shelf group has 8 shelf positions, arranged in the form of 2 rows and 4 columns, and the length and width of each shelf is 1 meter. The shelf groups are spaced 1 m wide from each other to serve as transportation paths for the robots. The robots represented by the icons deliver the goods to the designated shelves through the transport lanes. In the intelligent storage system model of this paper, each robot is set to carry only one cargo to avoid the impact on the path optimization algorithm by considering a robot transporting multiple cargoes at the same time.

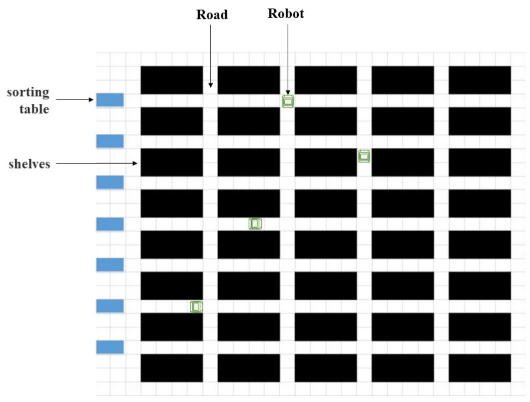

Figure 1. Warehouse planning diagram.

After the warehouse is planned, the path optimization algorithm does not plan the route by recognizing the real environment, so it needs to build the environment map that is convenient for the algorithm to recognize.There are many ways to build environment maps, such as visual map method, free space method, topology method, raster method, etc. The raster method is a form of data that divides the space into regular grids, each grid is called a cell, and the corresponding attribute values are assigned to each cell to represent the entity. This thesis intends to use the raster method to model the warehouse environment.

The map is divided into 650 grids using the grid method, each grid is 1 meter long and 1 meter wide, and the map is divided into 650 grids, as shown in Figure 2. Each picking table occupies two grids, each shelf group occupies 8 grids, and the width of the transport aisle is 1 grid. At the same time, in order to facilitate access, the warehouse generally manages all the shelves in the warehouse by partition, and chooses to place the frequently used goods close to the picking table. This paper adopts the classic ABC inventory classification management method, and points the storage area to the picking station according to the storage area. The average distance of the warehouse is used as the basis for division. The Euclidean selection method commonly used in distance selection divides the warehouse model into three parts, which store high-frequency goods, intermediate-frequency goods and low-frequency goods respectively, accounting for 40%, 40% and 20% of the storage area, convenient for the placement and transportation of goods, saving management investment.

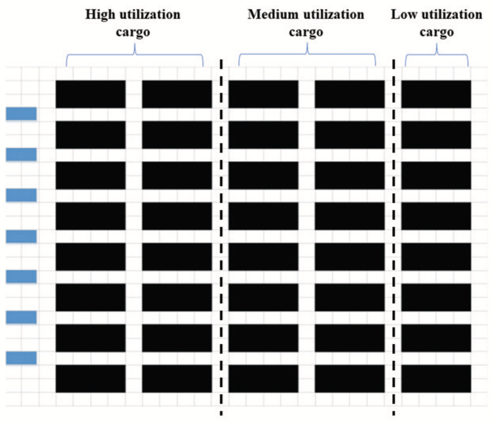

Figure 2. Grid map and shelf allocation schematic.

Improved Reservation Form. With the increase in the number of robots in the intelligent storage system, the path conflict problem gradually becomes an urgent problem to be solved. The prerequisite for solving the path conflict problem is the need to obtain the location information of the robots at each moment. To address this problem, this thesis uses an reservation table to monitor and record the motion trajectories of the robots, and improves the reservation table to record the robot motion data using a very small possible storage space.

The reservation table created is shown in Figure 3. The reservation table consists of several tables that record the robot location information. Each table in the reservation table represents the position information of all robots at a certain moment, namely the information of horizontal and vertical axes, as shown in the figure. The number of table rows is consistent with the number of robots used in the warehouse, and each row has two parameters, r_ix and r_iy, respectively. r_ix indicates the horizontal coordinate of the position of robot r_i at the moment of table record, and r_iy indicates the position of robot r_i at the moment of r_ix denotes the vertical coordinate of the robot r_i's position at the moment of the table record. The time interval between tables Δt is the time spent by the robot to move from the current grid center of the grid map to the adjacent grid center during normal operation. That is, the $k-c$th reservation table records the information at the $(k-c)\cdot$

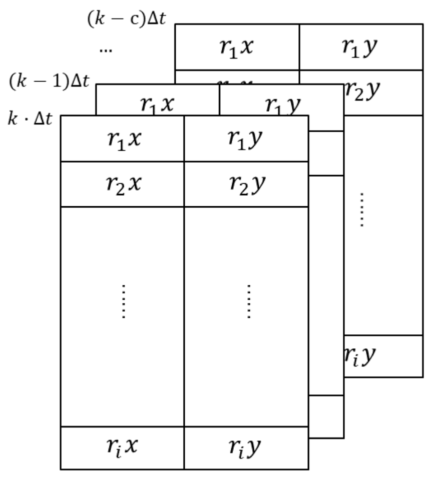

Figure 3. Improved reservation table.

Δt moment, where k and c are positive integers. For the convenience of simulation, the 90-degree steering time of the robot and the extra time consumed for start-stop are simplified in this paper with Δt as the smallest time unit, which is convenient for the reservation table to record the robot operation status.

In this paper, the reservation table form is improved. The original reservation table occupies a large amount of useless matrix space, causing matrix redundancy, and the query step is more cumbersome. Compared with the reservation table with the number of matrix rows and columns equal to the number of rows and columns of the raster map before the improvement, the improved reservation table in this paper reserves the number of matrix rows equal to the number of robots and the number of columns is 2, which occupies much less storage space. Taking the raster map model established in this paper as an example, the matrix size of the improved single reservation table is 10×2, which reduces the storage space by 96.92% compared with the reservation table with a size of 25×26 before the improvement. The original reservation table needs to locate the time, then locate the coordinate position, and then get the conflict information by querying the stored content of this position. When the improved reservation table is called, it is only necessary to compare whether the same coordinates appear at the same time to determine whether there is a conflict, which improves the query efficiency. To verify the improvement of the query efficiency of the reservation table, programs were written to simulate the query process of the traditional reservation table and the improved reservation table, respectively, as shown in Table 1. The results show that the query time of the improved appointment form is reduced by 23.34%.

2.2. Improved A* Algorithm

There are various algorithms for path planning, but in the raster map model established in this paper, the machine per capita only performs the commands of forward, stop, and 90-degree turn, which reacts to the path, i.e., only four directions of movement: up, down, left, and right. the A* algorithm, as the most effective direct search method for solving the shortest path in static maps, is suitable for solving the shortest path in this paper.

The A* algorithm prioritizes the expansion nodes based on the estimated cost function, selects the best node, and repeats the above steps at this node until it reaches the target point. The robot starts from the starting raster point and expands the surrounding raster at the current raster point. In this paper, the robot in the warehouse only moves up, down, left and right, so the four-neighborhood search method is chosen. The current position is called the parent node, and the estimated cost of the surrounding four directional grids is calculated and put into the candidate table for storing the extended nodes. After all the surrounding nodes finish estimating the cost, the grids with the smallest estimated cost are selected as the new moving position, and this position becomes the new parent node. The new parent node is used as the center to expand the grid, and this step is repeated until the robot reaches the target point. Finally, we obtain an optimal path from the starting point to the target point with the minimum cost.

The expression of the improved cost estimation function of the A* algorithm is :

$$f(n) = g(n) + h(n) + \sum_{j=1}^{p} t_{j(turn)} + \sum_{k=1}^{q} t_{k(wait)}$$

where $g(n)$ denotes the actual cost of moving from the starting grid to the current grid n. The actual cost is generally expressed by distance or time, and in this paper, time is used as a uniform scalar to compare the magnitude of the cost of the function $f(n)$. $\sum_{j=1}^{p} t_{j(turn)}$ is the sum of the extra time spent by the robot to turn from the starting grid to the current grid, p is the number of turns from the starting grid to the current grid, $\sum_{k=1}^{q} t_{k(wait)}$ is the extra time spent by the robot to wait in place due to path conflicts from the starting grid to the current grid, q is the number of turns from the starting grid to the current grid. The expression of $g(n)$ is:

$$g(n) = \frac{d}{v}$$

d is the actual distance traveled by the robot from the starting grid to the current grid n, and v is the speed at which the robot travels at a uniform speed. $h(n)$ denotes

Table 1. Comparison of the inquiry time of the reservation table

Experimental group	reservation table	improvement reservation table
Test 1	0.2018s	0.1518s
Test 2	0.2464s	0.1963s
Test 3	0.2330s	0.1743s
Average time	0.2271s	0.1741s

the heuristic estimation cost from the current grid n to the target grid point, and the expression is:

$$h(n) = \frac{d_n}{v}$$

where d_n is the estimated shortest distance of the robot from the current grid n to the target grid point, where the estimated distance is calculated using the Harmattan distance, expressed as the sum of the lateral and vertical distances between the current node n and the target point:

$$d_n = abs(n.x - goal.x) + abs(n.y - goal.y)$$

As the number of robots working in the warehouse increases, the A* algorithm expands the nodes in the parent node in a situation where the expanded nodes are already occupied by other robots, and if this problem is not solved, vehicle collisions are bound to occur. For this reason, a query step to the reservation table is required to be added to the A* algorithm. There are two main types of conflicts, vertex conflicts and edge conflicts. The vertex conflict is shown in Figure 4. Two robots arrive at the same grid at the same moment, and at this time the position information of the two robots is the same. By querying the position information of the robots in the reservation table, if the same coordinates appear twice in the same reservation table, it means that the current extended node will cause a vertex conflict, and it needs to return to the parent node to reselect the extended node.

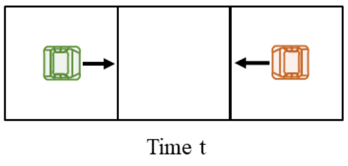

Figure 4. Vertex conflict diagram.

Meanwhile, since the expansion interval of the A* algorithm is a fixed value, it generally takes a raster as a cell and expands according to the raster. This will lead to the situation of edge conflict, namely, the reservation table is not queried to have robots in the same position at the same time, but the run will collide at the raster junction, as shown in Figure 5.

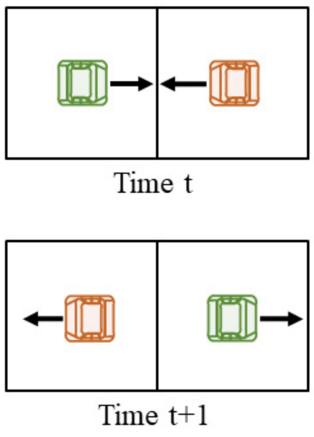

Figure 5. Edge conflict diagram.

Through the schematic diagram of edge conflict, it is obvious that the positions of two robots are switched at adjacent moments. In order to avoid the influence of edge conflict, this paper adds the operation of cross-call reservation table to the A* algorithm and brings in the judgment formula to determine the two robots to switch their positions at exactly two adjacent moments. The determination formula is as follows:

$$F_{g(\mathrm{n})}(x, y) \in R_{(k+1)} \cap N_{\mathrm{f(n)}}(x, y) \in R_{(k)}$$

where $R_{(k)}$ and $R_{(k+1)}$ are the matrices of the kth reservation table and the $(k+1)$st reservation table with all robot position information, respectively. $F_{g(n)}(x, y)$ denotes the horizontal and vertical coordinates of the parent node whose actual cost is $g(n)$ that the A* algorithm is currently expanding information, and $N_{f(n)}(x, y)$ denotes the information of the horizontal and vertical coordinates of the node with cost $f(n)$ whose parent node at the location of $F_{g(n)}(x, y)$ is trying to expand. When the path planning is finished, the robot runs to the position where the parent node $F_{g(n)}$ is located, it should correspond to the moment recorded in the kth reservation table, so how to locate the kth reservation table can be calculated by the following formula:

$$k = \frac{\text{length } (AllPath(i))\Delta t + g(n)}{\Delta t}$$

where $AllPath$ is a cell array that stores the walking paths of all robots. The number of cells is the same as the number of robots. Each cell stores the position and status information of the robot from the starting point to the end point of completing all tasks in a batch of orders. , Each row of the $AllPath(i)$ cell has four elements, which respectively represent the ordinate of the current position of robot i in the coordinate system, the abscissa of the current position of robot i in the coordinate system, and the current head direction of robot i (where the number 1 Represents the head of the vehicle towards the positive direction of the horizontal axis, clockwise and so on), the current time, and the number of rows of the $AllPath(i)$ cell is determined by the movement time of robot i. The time interval Δt between each row of the $AllPath(i)$ cell is the time it takes for the robot to move from the current grid center of the grid map to the adjacent grid center during normal operation, which is synchronized with the reservation table, so $length(AllPath(i))\Delta t$ represents the time required for the path that has been determined by the current robot. And $g(n)/\Delta t$ denotes the number of reservation table sheets to be recorded to run from the starting point of the current path planning to the parent node $F_{g(n)}$.

After the path finding by the A* algorithm added to the reservation table, an optimal path with the lowest cost from the start point of the current order task to the target point is obtained. The path information is sent to the robot to perform the transportation task and is also stored in the reservation table to facilitate the path planning for subsequent order tasks.

2.3. Dynamic Weighted Table

When multiple robots are about to pass through the same intersection at the same time, it is not possible to efficiently decide the order in which robots pass through by the control of reservation table only. Common warehouses commonly decide based on the remaining time for robots to reach the target point from the current conflict point, but this scheme may not be able to prioritize the most needed materials out of the warehouse and affect the efficiency of work progress.

In this paper, we innovatively propose a dynamic weighted table to solve the problem of passing order at intersections and to ensure that the goods with the highest demand are given priority to leave the warehouse. The dynamic weighted table is a matrix table composed of robot label and corresponding dynamic weights. The dynamic weights can be set differently according to different warehouse demands, and in this paper, the three dimensions, namely the demand degree of the type of goods delivered, the current transport status of the robot, and the remaining transport time of the robot's current task, are mainly analyzed for weights.

Queries are made on the demand degree of the type of goods delivered, the current transport status of the robot, and the remaining transport time of the robot for the current task, and stored to form a dynamic weighted table with a matrix consisting of the robot label and the corresponding dynamic weights, as shown in Table 2. The dynamic weights are denoted by w_i and refer to the weights of robot r_i at the current moment. The matrix composition of the weights w_i is as follows.

$$W_1 = [R_i, J_i, h_{(n)_i}]$$

where R_i denotes the priority level of the goods in the transport task currently performed by robot r_i. The robot is given a weight R_i based on the type of goods after receiving the task until the end of this task. According to the classification of shelves in Figure 2, here the goods are also divided into three types, high-frequency, medium-frequency, and low-frequency use of goods R_i are assigned a value of 1, 2, and 3, respectively. J_i indicates whether the robot r_i is carrying goods at the current moment, if it is, it means that the robot r_i is in the shipping stage at this time, assigned a value of 1, and if it is not, it means that the robot r_i is in the taking stage at this time, assigned a value of 2. $h(n)_i$ indicates the inspired time for robot r_i's current position to reach the target point.

When a conflict is encountered, the central controller invokes a dynamic weighting table to determine the order of passage by comparing the dynamic weights of the robots in the conflict situation. In this paper's warehouse model, the weight priorities are selected as follows: first, the more frequent the material demand, the more priority the corresponding robot has to pass, and the dominant factor; second, if the conflicting robots transport the same kind of material, query the shipment status, the robot in the shipment status has higher priority than the robot in the pickup status; third, when robots of the same material category and the same pickup/shipment status are in conflict, the remaining The shorter the inspiration time, the higher the priority. The paths of robots with higher priority remain unchanged; robots with lower priority need to recalculate their transport routes, as shown below.

2.4. Heat Map Algorithm

For the collision problem of robots in the warehouse system, this paper has been solved by adding the A* algorithm with improved reservation table and dynamic weighting table, but this is only a local conflict problem, and there is still a global congestion problem

Table 2. Dynamic weighting table

Robot ID(r_i)	Weight(W_i)	Resource(R_i)	Journey(J_i)	$h_{(n)_i}$
r_1	$W_1 = [R_1, J_1, h_{(n)_1}]$	R_1	J_1	$h_{(n)_1}$
r_2	$W_2 = [R_2, J_2, h_{(n)_2}]$	R_2	J_2	$h_{(n)_2}$
...	...	...	...	...
r_i	$W_1 = [R_i, J_i, h_{(n)_i}]$	R_i	J_i	$h_{(n)_i}$

Algorithm 1: Improved A* algorithm based on dynamic weighted table

```
1  initialize $Flag = 1$ to denote a conflict in the current extension position;
2  initialize $CP$ to denote the location of the conflict;
3  initialize "$ID1$, $ID2$ to denote two robot numbers that have encountered conflict, where robot $ID1$ has identified the transport route;
4  initialize $W_{ID1}$, $W_{ID2}$ to denote the information in the weighted table for these two robots at this point;
5  begin
6      while $Flag = 1$ do
7          for $i = 1$ to $3$ do
8              if $W_{ID1} > W_{ID2}$ then
9                  robot $ID1$ retains the route, robot $ID2$ re-routes;
10             end if
11             if $W_{ID1} < W_{ID2}$ then
12                 robot $ID2$ Select CP as the extended node; robot $ID1$ get it current locationre-route using the improved A* algorithm; The calculated new path is $NP$ new path time is $NPT$the original path time is $PT$;
13                 if $NPT<PT+Time for the robot to start in situ through a grid$ then
14                     robot $ID1$ runs on the new route; Update the NP to the reservation form; else
15                         robot $ID1$ choose to stay in place and wait for robot $ID2$ to pass;
16                 end if
17             end if
18             end if
19             else
20             end if
21         end for
22     end while
23 end
```

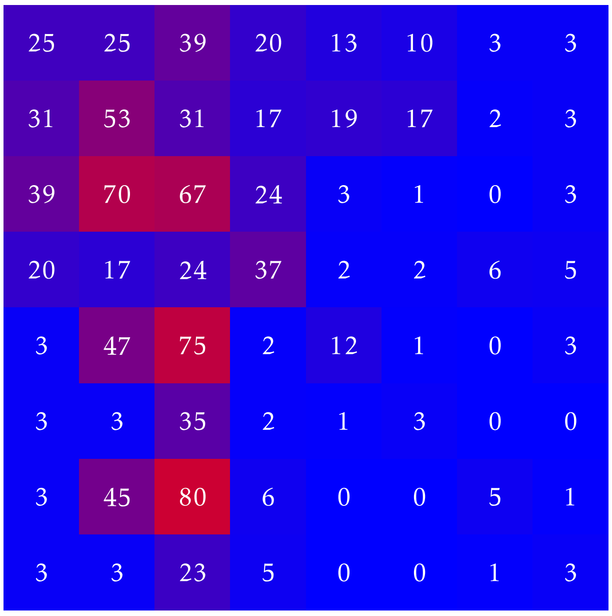

Figure 6. Diagram of the heat map reflecting the level of congestion.

in the warehouse system due to uneven assignment of tasks. With the increase in the number of robots and the centralization of order tasks, there may be local abnormal congestion and uneven distribution of vacancies in other locations of the warehouse, which greatly reduces the efficiency of the robot system in delivering goods.

To this end, this paper innovatively uses the heat map algorithm to monitor the congestion level of each aisle, picking table and shelf group of the warehouse in real time, as shown in Figure 6, and stores the congestion information in real time, and expresses the congestion level of each area in the form of heat value, which is used as the basis for sorting the remaining uncompleted tasks, prioritizing the assignment of tasks with low probability of conflict, dispersing the warehouse transportation pressure, and solving the global congestion problem caused by task assignment.

With the addition of the heat map algorithm, it is possible to obtain the number of robots passing through each aisle, picking table, and shelf group at each observation time as a reaction to the level of congestion. The data for the heat map comes from the reservation table. In the heat map algorithm, the congestion level of an aisle is reflected by the number of hours of robots present in this aisle over a period of time, and the formula for calculating the heat value of an aisle is as follows:

$$\text{Congestion} = \frac{\sum_{i=1}^{m} t((k-RI)\cdot\Delta t, k)}{RI\cdot\Delta t} * 100\%$$

where $RI \cdot \Delta t$ denotes the range of hours during which the congestion value is measured, expressed here by multiplying the number of sheets of recorded reservation tables by the unit time. $t((k-RI)\cdot\Delta t, k)$ denotes the $(k-RI)\cdot\Delta t$ recorded by a single robot from the $(k-RI)$th to the kth reservation table time to run in this lane. $\sum_{i=1}^{m} t((k-RI)\cdot\Delta t, k)$ denotes the duration of all robots running in this lane during the $(k-RI)\cdot\Delta t$ time recorded from the $(k-RI)$th to the kth reservation table. *Congestion* then denotes the thermal value of the lane during the current measured time range , expressed through the percentage form.

The congestion degree of the shelf group cannot be judged directly by recording the vehicles within a certain range. In this paper, the current congestion value of the shelf group is obtained by first calculating the congestion degree of the aisles, and then adding up the congestion values of two short aisles and two long aisles around the shelf group for a total of four aisles and taking the average value. Firstly, the aisles are coded to facilitate the query and retrieval of the thermal value. The leftmost position near the access warehouse is taken as the initial position, and the short aisles and long aisles are coded separately with the top-down and left-to-right rules. The formula for calculating the thermal value of the shelf group is shown below.

$$C_h(i,j) = \frac{C_w(i,j) + C_w(i,j+1) + C_l(i,j) + C_l(i+1,j)}{4}$$

where $C_h(i,j)$ denotes the thermal value of the shelf group in which the i-th row and j-th column are located. $C_w(i,j)$ and $C_w(i,j+1)$ denote the thermal values of the short aisles on the front and back sides of the shelf group, respectively, and $C_l(i,j)$ and $C_l(i,j+1)$ then denote the thermal values of the long aisles on the left and right sides of the shelf group, respectively.

The unassigned tasks correspond to the shelf groups where their target points are located, and each unassigned task is paired with a picking table one by one in shelf groups. After the pairing, the corresponding shortest path is found by the A* algorithm, and the congestion level of the aisles passed in the path is summed with the picking table thermal value and the shelf group thermal value as the overall thermal value of this picking table-shelf group. All the thermal values are sorted in ascending order, and the tasks with smaller thermal values are placed first, and if the thermal values are the same, the task point with the closest distance to the matching transport robot takes precedence. The reordered task list avoids orders for goods that are in the congested range of shelves, as well as choosing to avoid picking tables with long waiting times, enabling dynamic assignment.

2.5. Process of the Path Planning Method

After the above research, the processing of the local conflict problem is completed by adding the improved A* algorithm of reservation table with the dynamic weighted table, and the processing of the global conflict problem is completed by the heat map algorithm with the real-time assignment of task orders. Through the above methods, a complete set of multi-robot path planning method is formed, and the overall operation flow is as follows:

Step 1: Check the order remaining, if there is no order remaining then wait, no new orders generated for a long time then end the run; if there is an order remaining then enter step 2.

Step 2: Get information about the start and target points of the remaining tasks.

Step 3: Using a heat map algorithm, tasks are assigned to each robot that is in an idle state.Specifically, the following steps are included.

Step 3.1: Call the kth-RI to kth reservation table record centered on the current moment, and calculate the heat value of the aisle around the shelf group and the shelf group itself.

Step 3.2: Create a visualized heat map based on the heat value data of all shelf groups to show the distribution of congestion at the current time.

Step 3.3: Correspond the non-assigned tasks to the shelf groups where their target points are located, sort the tasks according to the heat value size, and assign them to the idle robots.

Step 4: Using the improved A* algorithm, path finding is performed for each robot r_i that has an order transportation task.

Step 5: Obtain the location information of each robot at each moment, and store the location information uniformly to form a reservation table.

Step 6: The reservation table is called to query vertex conflict and edge conflict, and if there is a conflict, go to step 7, and if there is no conflict, go to step 8.Specifically, the following steps are included.

Step 6.1: Call the reservation table and check whether the same coordinates appear in the reservation table at each moment, if so, we can determine that this extended node will have vertex conflict and go to step 7; if not, go to step 6.2.

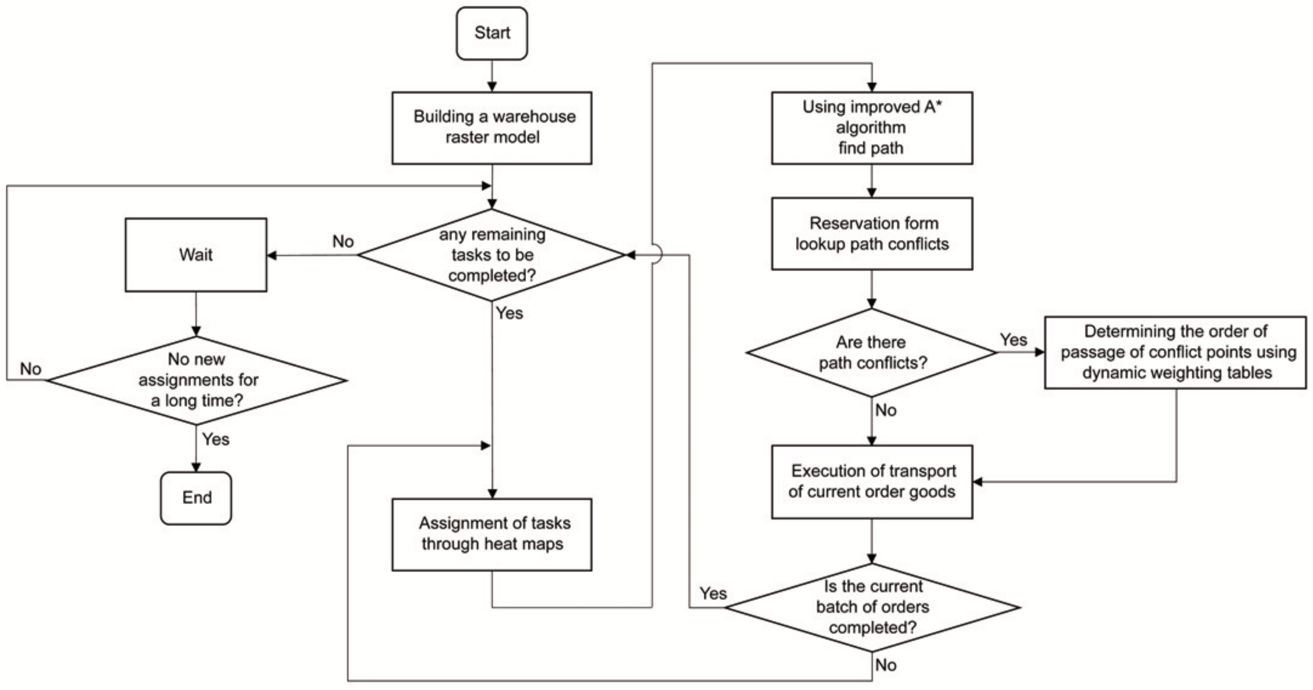
Start
Building a warehouse raster model
any remaining tasks to be completed?
No
Wait
No new assignments for a long time?
No
Yes
End
Yes
Assignment of tasks through heat maps
Using improved A* algorithm find path
Reservation form lookup path conflicts
Are there path conflicts?
Yes
Determining the order of passage of conflict points using dynamic weighting tables
No
Execution of transport of current order goods
Is the current batch of orders completed?
Yes
No

Figure 7. Flow chart of the optimization algorithm.

Step 6.2: Cross-call the reservation table and bring in the decision formula to determine whether the two robots are exactly aligned at two adjacent moments.

Step 7: Using a dynamic weighting table, the order of passage of conflict points is determined.Specifically, the following steps are included.

Step 7.1: A query is made for the demand degree of the type of goods to be transported, the current transport status of the robot, and the remaining transport time of the robot for the current task, and stored to form a dynamic weighting table with a matrix of robot labels and corresponding dynamic weights.

Step 7.2: When a conflict is encountered, the central controller calls the dynamic weighting table and determines the order of passage by comparing the dynamic weights of the robots in the conflict situation. The path of the robot with higher priority remains unchanged; the robot with lower priority needs to return to step 4 and recalculate the transport route.

Step 8: Robot r_i executes the current order task according to the route of merit search.

Step 9: Check the remaining status of the current batch of orders, if there is no order remaining, go to step 1; if there is an order remaining, go to step 3.

The overall flow chart is shown in Figure 7.

3. Simulation Results

In order to verify the effectiveness of the dynamic weighting table and the heat map algorithm in improving the efficiency of warehouse goods transportation respectively, this paper conducts a comparative analysis through two sets of experiments. In both sets of experiments, five groups of transport orders were selected, and the number of tasks in each of the five groups was 100, 150, 200, 250 and 300, all transported by 10 robots.

(i) In order to verify the improvement of transportation efficiency by dynamic weighting table, five groups of transportation tasks were run by the traditional A* algorithm and the dynamic weighting table-based optimization algorithm in this paper, and the improvement of transportation efficiency of goods with high priority by dynamic weighting table was compared and analyzed by recording the final transportation completion time of all high-frequency goods in the tasks. The simulation results are shown in Figure 8. Compared with the traditional A* algorithm model, the improvement in the efficiency of transporting high-frequency goods in 5 groups of tasks is more significant with the increase in the number of goods after the optimization of the dynamic weighting table-based optimization algorithm. The average reduction in HF cargo transportation time is 13.83%, which is a significant improvement in efficiency.

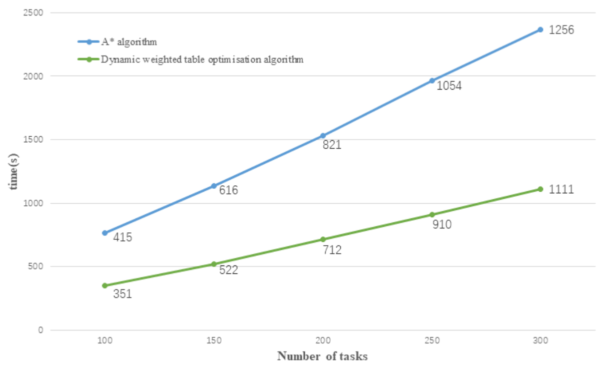

Figure 8. Efficiency comparison chart using dynamic weighted table.

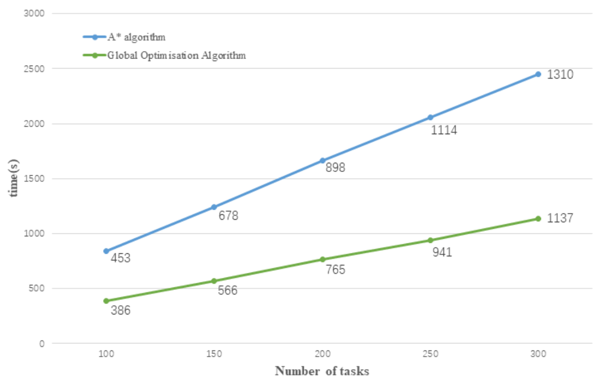

Figure 9. Efficiency comparison chart using the global optimization algorithm.

(ii) In order to verify the improvement of transportation efficiency by the heat map algorithm, five groups of transportation tasks were run by the traditional A* algorithm and the optimization algorithm based on dynamic weighting and heat map algorithm (referred to as the global optimization algorithm) in this paper, and the improvement of global transportation efficiency by the heat map algorithm was compared and analyzed by recording the final transportation completion time of all order tasks. The simulation results are shown in Figure 9. After the optimization of the global optimization algorithm, the final transport completion time of the order tasks is improved significantly compared with the traditional A* algorithm model, and the average transport completion time is reduced by 14.97%, which verifies the effectiveness of the heat map algorithm for the improvement of global cargo transport efficiency.

4. Conclusion

In this paper, a multi-robot path planning method based on dynamic weighting and heat map algorithms is proposed for intelligent storage systems for the multi-robot vertex and edge collision problem and the matching rule problem between robots and order and picking tables in storage systems. The following work was accomplished: a warehouse raster map was created; the reservation table was improved to greatly reduce the storage space occupied by the reservation table while increasing the operational efficiency. Improvements were made to the A* algorithm to achieve the objective of avoiding vertex conflicts and edge conflicts at the same time. Dynamic weighting tables are added to solve multi-robot driving strategies for intersection conflicts and to ensure that the most urgently needed goods are prioritised for release. Innovative heat map algorithms are used to monitor the congestion level of each aisle, picking table and shelf group in the warehouse in real time, avoiding congested areas and enabling dynamic assignment of tasks. Finally, the feasibility and effectiveness of the solution is verified through simulation.

Acknowledgement

This work by S. Bi was supported by the Natural Science Foundation of Shandong Province(ZR2020KF027), the National Key R & D Program of China (2018AAA0101703).

References

[1] S. Garnier, J. Gautrais, and G. Theraulaz, “The biological principles of swarm intelligence,” *Swarm Intelligence*, vol. 1, no. 1, pp. 3–31, 2007.

[2] J. Peng, “The robot path optimization of improved artificial fish-swarm algorithm,” *Computer Modelling and New Technologies*, vol. 18, no. 6, pp. 147–152, 2014.

[3] Y. Tian, “Study of two firefly algorithms,” *Scientist*, vol. 4, no. 6, pp. 21–29, 2016.

[4] A. Hidalgo-Paniagua, A. Vega-Rodrguez, M, and J. Ferruz, “Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach,” *Soft Computing*, vol. 21, no. 4, pp. 949–964, 2017.

[5] S. Li, W. S, P. Guo, S. Zhang, and X. Pengfa, “Research on sar drone global path planning based on improved a* algorithm,” *Chinese Medical Equipment Journal*, vol. 41, no. 12, pp. 16–20, 2020.

[6] D. Roy, S. Nigam, R. Koster, and et al, “Robot-storage zone assignment strategies in mobile fulfillment systems,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 122, pp. 119–142, 2019.

[7] J. Hu, X. Wang, Q. Zhang, and R. Quan, “Optimized multi-step ant colony algorithm for robot path planning problem solving,” *Transducer and Microsystem Technologies*, vol. 40, no. 10, pp. 121–124, 2021.

[8] C. Pan and M. Guo, "Batch picking path planning simulation of warehouse mobile robot," *Computer and Modernization*, vol. 2017, no. 2, pp. 12–16, 2017.

[9] M. Chen, T. Qian, S. Zhang, and J. Wang, "Obstacle avoidance and cooperative path planning method of warehouse logistics robot cluster," *Modern Electronics Technique*, vol. 42, no. 22, pp. 174–177+182, 2019.

[10] H. Ouyang, Y. Quan, L. Gao, and D. Zou, "Hierarchical path planning method for mobile robots based on hybrid genetic particle swarm optimization algorithm," *Journal of Zhengzhou University(Engineering Science)*, vol. 41, no. 4, pp. 34–40, 2020.

[11] Y. Xu, J. Cao, S. Yuriy, S, and Y. Zhuang, "Distributed kalman filter for uwb/ins integrated pedestrian localization under colored measurement noise," *Satellite Navigation*, vol. 2, no. 1, pp. 305–314, 2021.

[12] H. Yoshitake, R. Kamoshida, Y. Nagashima, and et al, "New automated guided vehicle system using real-time holonic scheduling for warehouse picking," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1045–1052, 2019.

[13] Y. Zhang, "Advances in multimodal data fusion in neuroimaging: Overview, challenges, and novel orientation," *Information Fusion*, vol. 64, no. 0, pp. 149–187, 2020.

[14] Y. Zhang, "Improved breast cancer classification through combining graph convolutional network and convolutional neural network," *Information Processing and Management*, vol. 58, no. 2, p. Article ID: 102439, 2021.

[15] S. Wang, "Secondary pulmonary tuberculosis recognition by rotation angle vector grid-based fractional fourier entropy," *Fractals*, vol. 30, no. 1, pp. 2240047, 2022.

[16] P. Liu, M. N. Huda, L. Sun, and H. Yu, "A survey on underactuated robotic systems: bio-inspiration, trajectory planning and control," *Mechatronics*, vol. 72, pp. 102443, 2020.

[17] M. N. Huda, P. Liu, C. Saha, and H. Yu, "Modelling and motion analysis of a pill-sized hybrid capsule robot," *Journal of Intelligent & Robotic Systems*, vol. 100, no. 3, pp. 753–764, 2020.