IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON HUMAN-CENTERED SMART SYSTEMS AND TECHNOLOGIES

Received January 5, 2018, accepted February 9, 2018, date of publication March 26, 2018, date of current version June 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2819199

Collision-Free Route Planning for Multiple AGVs in an Automated Warehouse Based on Collision Classification

ZHENG ZHANG[1], (Student Member, IEEE), QING GUO[1], JUAN CHEN[1], AND PEIJIANG YUAN[2]

[1]College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China
[2]Intelligent Technology and Robotics Research Center, Beihang University, Beijing 100191, China

Corresponding author: Qing Guo (guoqing@mail.buct.edu.cn) and Juan Chen (jchen@mail.buct.edu.cn)

ABSTRACT An automated warehouse system contains a number of materials, workstations, and multiple Automated Guided Vehicles (AGVs). The automated warehouse is server-controlled. This paper proposes a collision-free routing method for AGVs based on collision classification. This method can deal with collisions arising in the automated warehouse. It first divides the warehouse environment into five areas, and then performs route planning. In this paper, the environment map for AGVs is described by using the grid method. The initial route of each task is predetermined by improved Dijkstra's algorithm. The server detects the potential collisions by comparing each workstation's ID and corresponding time window in every route. This paper presents four collision classifications and three solutions. Based upon the analyses and experiments, we select the corresponding solution for each type of collision. Presented case studies demonstrate the efficiency of the proposed collision-free route planning approach.

INDEX TERMS Multiple AGVs, route planning, time window, the status of AGV, collision classification.

I. INTRODUCTION

The demand for flexible automation has significantly increased the use of Automated Guided Vehicles (AGVs) in Automated Storage and Retrieval System (AS/RS). For example, it can get access to transport goods automatically without human labors. The AGVs in the warehouse environment travel among the multiple workstations by following a set of predetermined routes. However, the management becomes a challenging problem especially when bidirectional routes are used to gain efficiency and flexibility on the warehouse. When multiple AGVs are employed, the management problem such as collisions may happen. The problem of collisions results from competition for the manufacturing resources in the warehouse [1]. Collision solution is important in the automated warehouse systems. An effective and efficient routing algorithm for finding the collision-free routes is of importance for AGVs. The AGV will be assigned for the new transportation task with the minimum cost.

There are numerous researches on route planning algorithms for AGVs. Several collision-free route planning techniques have been offered in the literature such as genetic algorithm [2], particle swarm optimization [3], neural network [4], visibility graph [5], colony optimization [6] and regional control method [7]. One of the simplest approaches is the Regional Control Method (RCM) [8]. The operational space is divided into a number of non-overlapping regions and each region can only accommodate one traveling AGV. The goods are exchanged through an exchange station. Since the warehouse can be divided into a number of regions, the RCM can be used in routing problems for warehouse AGV. However, the productivity and efficiency of the system are reduced a lot by using the regional control method, because it is a waste of time to exchanging goods in exchange station. Another approach is the Artificial Potential Field (APF) method [9]. It is assumed that the vehicles are moving in an abstract artificial force field. The potential function is defined as a set of equations of the attractive potential of target and the repulsive potential of the obstacles. A vehicle can then find a collision-free route along the direction of the declining potential function. Besides, MC Chen [10] proposed a mathematical model of Vehicle Routing Problems (VRPs). Based on high complexity of the model, a Particle Swarm Optimization (PSO) with a self-learning approach is tailored to solve Vehicle Routing Problems. The APF and PSO method is suitable for solving route planning problem without route limitation for AGV and free-travel situation. A heuristics-

based approach and a dynamic polynomial-time sequential are proposed in [11], which enabled the task assignment and guaranteed collision-free route planning. Because the problem of finding a series of collision-free routes for multiple AGVs is still Non-deterministic Polynomial-time Hardness (NP-hard), Wu and Zhou [12], [13] presented a Colored Resource Oriented Petri Net (CROPN) modeling method. In this method, the problem of collision and deadlock is dealt with based on the CROPN model and the deadlock-free operation condition. The CROPN method is proposed based on the situation that the on-line routing is flexible while the total routes of AGV are determined. I n many real-life application domains, such as factories, container terminals, and airports, the environment map is quite unusual. Therefore, Mors [14] proposed an algorithm, which can eliminate delays by using re-planning of the AGVs in case one or more AGVs are delayed.

As the increasing number of the tasks, the AGV should make a task decision priority to determine which task to implement first and which to execute next. The purpose is to minimize the cost of all the tasks. Motivated by this problem, Shuai Hao *et al.* [15] combined task decision problems with routing algorithm, and proposed an optimal method to help the AGV make the optimal task decisions. Many scholars have draw attention to the whole time. The scheduling and routing algorithm of multiple AGVs is different from the conventional route selection algorithm [16]. It is required to determine the tasks and collision-free routes for different agents in the multi-AGVs system at the same time. Ivaylo *et al.* [17] employed the heuristic function for multiple AGVs, so they should decide the order of searching workstations. This method can reduce the execution time of a single task. The Monte Carlo Localization (MCL) [18] algorithm is a recently proposed probabilistic-based approach, which is used to estimate the position and orientation of an AGV in the given map. Whenever multiple AGVs travel, the MCL can shift the AGVs to predict their new goal simultaneously.

In the multi-AGVs warehousing environment, the AGVs travel along a number of mesh-like routes at the same time, the collisions will happen in the process of traveling. Several literatures also recognized some common collision types. Rong [19] introduced two common collisions types between two AGVs:

a) One AGV is parked on another AGV's route;
b) Two AGVs are traveling at the same cross road simultaneously.

Chengbao Liu *et al.* [20] mentioned three types of collisions:

a) Intersection collision;
b) Catching up with collision;
c) Facing collision.

They both employed waiting strategy — the traveling AGV should stop before the collision station — to solve all the collisions. Besides, Yuan *et al.* [21] introduced two types of collisions in warehouse environment:

a) Collision due to catching up;
b) Collisions in intersection.

Then he presented a collision-free routing method of multiple AGVs based on improved A* algorithm. However, these mentioned collision types are not completed. Besides, the collision avoidance algorithms are non-robust.

This paper is an extension of our previous work presented at the 2017 IEEE International Conference on Systems, Man, and Cybernetics (Zhang *et al.* 2017). The major additions are:

- Based on the multiple AGVs travelling in the mesh-like environment, this paper classify collisions into four types, i.e. head-on collision, cross collision, node-occupancy collision, and shelf-occupancy collision. Then, three feasible solutions are proposed, and then the corresponding solution is selected for each type of collision;
- Furthermore, the shelf-occupancy collision is classified into two types according to the status of AGV (i.e. on-load or non-load task).
- We introduced three solutions to solve the collision. According to the experiments and analyses, we employed corresponding solution for different types of collisions. It decreases the computing load of the upper system compared with our previous work. In the previous work, an adaptive approach was proposed. That is to say, when the collision happens, the server chooses the appropriate solution according to the travel time of AGV to solve the collision;
- We improve the traditional waiting strategy by delaying the starting of the AGV, which forms one of solutions.

This paper is organized as follows: section II describes the AGV system environment model, which includes description of the work environment and the configuration of the multi-AGVs control system; followed by section III presents the route planning policy for multiple AGVs and single AGV routing algorithm; section IV shows the collision classification and collision solution strategies. In addition, the simulation environment situations are included; section V shows the cases study. Besides, the aforementioned algorithmic solutions are compared with the existing approaches; finally some conclusive remarks are given in section VI.

II. DESCRIPTION OF THE WORK ENVIRONMENT

A. THE ARCHITECTURE OF MULTI-AGVS SYSTEM

The coordination of the AGVs along the warehouse-map can be performed through hierarchical control. Several existing AGV control systems in the warehouse are generally classified into two classes: the centralized route planning approach and the decentralized route planning approach. Centralized and decentralized control strategies can be used for optimal coordination of AGVs. The coordination traffic control was composed by a hierarchical control configuration, which consists of two layers [3], [22]. Although the decentralized approach does better in handling multiple AGVs, it features in the drawback such as collisions and deadlocks [23]. Since the

multiple AGVs are traveling in a dynamical warehouse environment, traffic management becomes a key factor. Because of the merit of more manageable, the centralized route planning approach is mainly used in the multi-AGVs system.

According to the real environment in warehouse, it is a key issue to establish the suitable environment model. There are many approaches for modeling environment of the warehouse, and the most common method is the topological theory to realize the visualization-numbered representation of the environment [24]. By modeling nodes (i.e. workstations) and lanes (i.e. routes), the proposed approach can implement collision-free route planning for multiple AGVs.

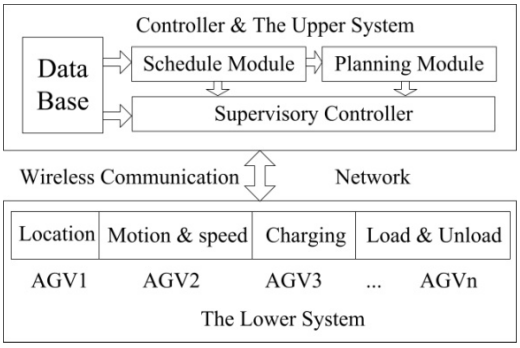

FIGURE 1. Layered system structure.

The whole system involves two layers (i.e. the upper-lower system structure), which are coupled with each other. In this paper, we decouple the upper layer planning from the lower level logical control. The control architecture for multi-AGVs system is shown in Fig. 1. The upper system consists of three modules: scheduling module, planning module, and supervisory module. The lower system layer consists of several AGVs, and each AGV includes a location module, a motion & speed module, and a charging module.

The scheduling and routing problem is solved at the upper system, also called server. The upper system layer determines and issues several pairs of feasible initial starting, destination, and calculates optimal routes between them. The server supervises the running state of the AGV. The upper system layer communicates with the lower system layer by using Wi-Fi or other wireless communication networks. The server computer sends route planning information to each AGV and it can send their location, motion, and power information to the upper system, respectively. However, the communication between the AGVs is not allowed.

B. DESCRIPTION OF THE WORK ENVIRONMENT

The topological method (i.e. the grid method) is used to describe the environment map as shown in Fig. 2. The map is divided into five parts: Manual Sorting Space, Operating Centre, Charge Space, AGV Driving Space, and Parking Space.

(a) Manual Sorting Space: The staff will complete the pickup and delivery in this area when AGV delivers the goods

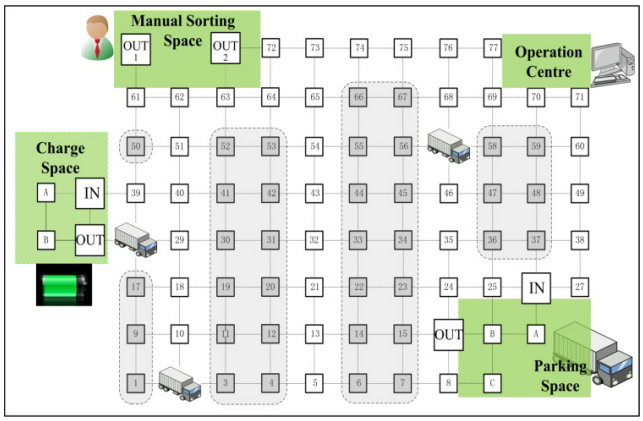

FIGURE 2. The environment map.

to the manual sorting space. The manual sorting space consists of several operation platforms and two exits (i.e. OUT_1 and OUT_2);

(b) Operating Centre: The server computers and the central control room are located in this area;

(c) Charge Space: When the power of the AGV is less than 20%, the AGV need to charge at the "Charge Space." The charge space consists of two charging platforms (i.e. A, B), one entrance (i.e. IN) and one exit (i.e. OUT);

(d) AGV Driving Space: The AGV completes picking up, putting down, and delivery. It is the main area for route planning, which is the main topic of this study. It is consisted of several workstations numbered as 1.2.3...;

(e) Parking Space: All vehicles are parked here before the system is turned on. Besides, store the breakdown vehicles. The parking space consists of three parking platforms (i.e. A, B, C), one entrance (i.e. IN) and one exit (i.e. OUT). The entrance can be regarded as workstation 26, and the exit can be regarded as workstation 16.

In the concerned automated warehouse, the goods are loaded on the shelves, and the shelves are arranged in the mesh-like and rectangle form. The workstation is the place where the AGVs pick up, pick down goods, and charge. AGVs will travel between workstations with shelves loaded upon them. The AGVs can move either forward or backward along the several regular paths connecting two adjacent workstations, and can only turn when it arrives at a workstation.

In order to ensure the vehicles travel along the routes fluently, and avoid abnormal collisions between two AGVs, it is essential to leave appropriate channels for traveling to workstations on the map for multiple AGVs. The white areas represent the areas that the goods cannot be stored, which are channels. The gray blocks represent the storage areas for shelves. There are no designed paths between each two adjacent workstations in the actual environment but AGVs can pass through them. One path between two adjacent workstations only can pass one AGV at one time. The deviation rectification algorithm and motor controller are used to ensure the AGV runs straight between the workstations. Nodes and arcs are stored in the grid data format.

Each workstation is marked and stored by using the two-dimensional Quick Response (QR) code label, which stores the ID (identity) and coordinate of the workstation. The QR code series are also used to verify the current location and direction of the AGV. AGVs respond after identification. The upper system can check whether the AGV has completed the task, and then guide the AGV to travel to the goal workstation. The two-dimensional code labels are posted on the surface of the environment, which include straight, turning, destination, and other corresponding coordinate's information. AGVs respond as quickly as after identification, and they will travel along the determined routes. The two-dimensional QR code labels are shown in Fig. 3.

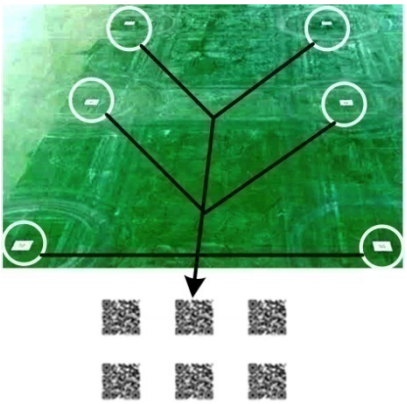

FIGURE 3. Two-dimensional code labels.

The workstations and the mesh-like routes connecting two adjacent workstations are considered as nodes and arcs respectively and are stored in the grid data format. The white square represents the two-dimensional QR code labels; the number in the box represents the ID of the workstations.

III. COLLISION-FREE ROUTE PLANNING ALGORITHM POLICY

In this paper, the shelves are placed above the nodes in the concerned warehouse. As shown in the Fig. 4, it is 75 cm between the bottom layer and the ground. The height of AGV is 50 cm. If the AGV needs to deliver the goods, the tray on the AGV will be lift. Since the height of the shelf bottom is greater than the height of the AGV, therefore, a non-load AGV can pass through under the shelf, whereas an on-load AGV cannot. Herein, the non-load AGV means that AGV is free of goods (i.e. no shelf is carried on); the on-load AGV means that AGV is loaded by goods (i.e. one shelf is carried on). The task types can then be classified into two categories: (1) On-load task: AGV picks up goods (i.e. shelves) at the starting workstation and then transport them to the destination workstation to pick down; (2) Non-load task: the empty AGV travels from the starting workstation to the destination station, and then pick up goods there. Based on the classification, an on-load vehicle should avoid the workstation which was occupied by a shelf. However, the non-load vehicles do not. So, different routing method is employed for the on-load and non-load AGVs.

FIGURE 4. The shelf and the AGV.

The route planning of the multi-AGVs control system consists of two stages: the off-line prediction and the on-line adjustment. The procedure of the off-line prediction stage is described as follows:

A. TASK DETERMINING

The server schedules the tasks and multiple AGVs according to the priority of the task. And then pair tasks with idle vehicles, which are selected AGV.

B. ROUTE PLANNING FOR EACH TASK (I.E. SINGLE AGV ROUTE PLANNING)

The server plans the routes for every selected AGV. The routes are determined by using improved Dijkstra's algorithm [25] mentioned in section B.

C. COLLISION DETECTION AND SOLUTION

Potential collisions are detected by comparing the workstation's ID, its occupancy time among all the tasks, and the moment when an AGV arrives. The collision detection was implemented iteratively. If a collision was detected, the schedule of the later arrival AGV should be modified using the collision-free strategy proposed in later.

D. DISPATCH THE DETERMINED ROUTES TO THE SELECTED AGVS

Order the idle vehicles to execute the task through wireless communication.

The collision detection method together with the solution is the main focus of this paper and is discussed in detail in the following sections. This procedure is an open loop control policy, which is very sensitive to the system randomness and is non-robust to any contingencies arising in practice [18]. For example, the starting time of an AGV may be delayed due to a longer pick up or pick down time. In that case, the AGV will not arrive at the workstations as determined by the off-line prediction. If the schedule remains unchanged, collisions may happen. Therefore, the on-line adjustment is of importance in the real-time situations.

When AGV passes a two-dimensional code label, both the workstation's ID and the moment are sent to the upper system layer. The upper system then checks whether the information

is in accordance with the predetermined schedule. If the workstation's serial number is different or the time deviation is greater than the threshold value, the on-line adjustment strategy will be implemented. In order to check frequently whether there is any obstacle on the route, the infrared distance sensor was installed in front of the AGV. However, the on-line adjustment approach is not the topic of this paper, which is not addressed in this work.

IV. SINGLE AGV ROUTING ALGORITHM

The candidate route of each task (i.e. the route of each single AGV) is predetermined by using the improved Dijkstra's algorithm [25]. As shown in Fig. 2, there is more than one shortest route with the same distance in the warehouse environment. Whereas, the traditional Dijkstra's algorithm stores only one intermediate node so that only one shortest route can be found through one searching. For example, Fig. 5 (a)-(d) illustrate that there are 4 shortest routes between workstation 11 and 22. But, only one shortest route can be found through traditional algorithm, i.e. (a).

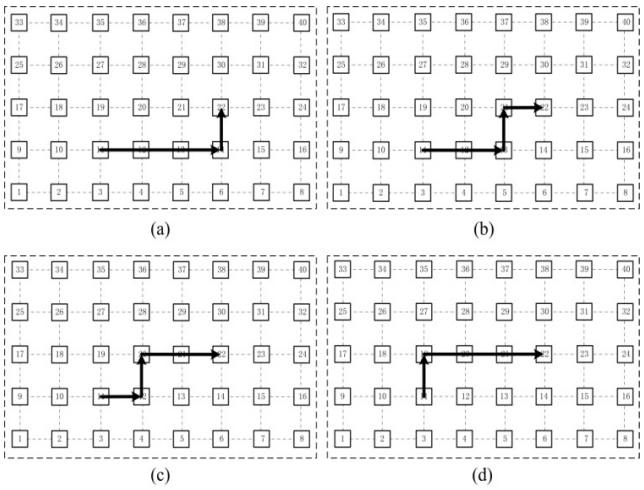

FIGURE 5. The shortest routes between node 11 and 22 (a) the first route, (b) the second route, (c) the third route, (d) the fourth route.

We improve the Dijkstra's algorithm by reserving all nodes with the same distance from the source node as the intermediate nodes, and search again from all the intermediate nodes until traversing to the terminal node. With multiple iterations, all the shortest routes with the same distance are found [25].

Consider a directed graph $D = (V, E)$ with n nodes and e arcs, where V is the set of nodes and E is the set of arcs. $W(A, B)$ denotes the weight of the arc (A, B). The improved Dijkstra's algorithm is described as follows:

(1) Initialization. Mark the source node v and put it into S.

(2) Traverse the nodes in V-S and select all the adjacent nodes as the candidate intermediate nodes.

(3) Select the node i with the smallest number among the candidate intermediate nodes and put it into S set.

(4) Regard i as the new intermediate node. Repeat (2) and (3) and choose the node j with the smallest number among the adjacent nodes. Update the distance between the stating node v and the node j. If $DIS(j) > DIS(i) + W(i, j)$, i.e. the distance passing through the node i is shorter than that not through it, altered $DIS(j)$ to $DIS(j) = DIS(i) + W(i, j)$, and put node j into S set.

(5) Repeat (2)-(5) n-1 times. All the shortest routes from the staring node to the goal node are stored in $DIS(X)$ when the search iteration traverses to the goal node.

Besides, taking into account of the turning time at the intermediate nodes, the number of turning nodes is determined for each of the shortest routes.

(6) One or two optimal routes, which have shortest distance and the shortest traveling time, are found. The route found first is regarded as the selected route, and the other route as the candidate route.

The route is stored in the format as follows: $T(t)_i = (V_i(t), E_i(t), S_k^i(t), rt_i, ST)$ (i = 1, 2, 3 . . . n). Where i represents the number of AGV andrt represents the AGV. k the series number of the workstation in the route, t the time when the AGV passes the node, V the ID of the source node, E the ID of the destination node; ST is a flag bit. $ST = 1$denotes on-load task, and $ST = 0$ a non-load task. Mentioned in section II, a non-load vehicle can pass through under a shelf, and the node occupied by the shelf is regarded as available. The node occupied by the shelf is regarded as an obstacle, which should be deleted from the environment map. So the upper system has to know in advance whether or not the node in the route of AGV is occupied by the shelf. If the node is not occupied by shelves, no matter whether the AGV is non-load ($ST = 0$) or on-load ($ST = 1$), it can pass through the node smoothly. However, if the node is occupied by the goods, we need to determine the current status of the AGV.

Due to each workstation is marked by a QR code label, which stores the ID and location of the workstation. When AGV arrives at the goal of the task, the information of this QR code label is transmitted to the upper system through wireless communication. Therefore, the upper system can easily obtain the terminal position of the AGV. In this way, for the on-load AGV (i.e. $ST = 1$), when the upper system implements route planning, it should delete the node occupied with the shelf, along with the adjacent path. It indicates that this AGV cannot pass through this node; for the non-load AGV (i.e. $ST = 0$), it should not delete the node along with the adjacent path. Therefore, it indicates the on-load AGV can pass through this node.

V. COLLISION DETECTION AND SOLUTION

The collision detection between every two AGVs is implemented iteratively among all the AGVs based on the priorities. There are four types of potential collisions in the automated warehouse system as shown in Fig. 6: (a) Head-on collision, (b) Cross collision, (c) Node-occupancy collision, [18] and (d) Shelf-occupancy collision. In the picture, the solid box represents AGV and the hollow box denotes the shelf.

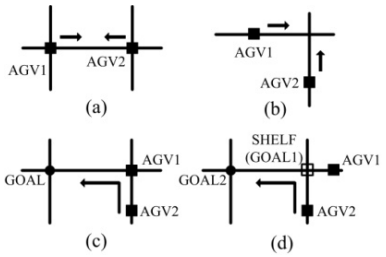

FIGURE 6. Collision types (a) Head-on collision, (b) Cross collision, (c) Node-occupancy collision, (d) Shelf-occupancy collision.

A. COLLISION CLASSIFICATIONS

a) Type 1: Head-on collision: As the examples shown in Fig. 7, the head-on collision happens if two AGVs travel on the same path at the same time or two vehicles pass the same workstation at the same time, but the directions are opposite. The time window model is shown in Fig. 8, where m, n and p represent the workstations in AGV's routes. Time window consists of several colored blocks, each block represents a workstation. Each AGV reserves some workstations on its route. The free time windows between the reserved ones are available for scheduling other vehicle occupancies.

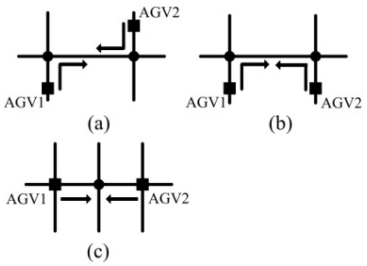

FIGURE 7. The head-on collision (a) and (b) the head-on collision happens on a path, (c) the head-on collision happens on a node.

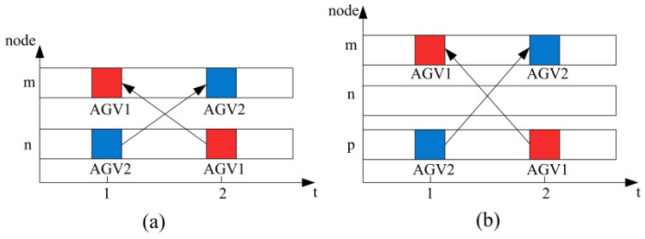

FIGURE 8. The time window of head-on collision (a) the time window of the Fig. 7(a) and 7(b), (b) the time window of the Fig. 7(c).

b) Type 2: The cross collision happens when two AGVs compete for the workstations at the intersections. The examples and the time window are shown in Fig. 9 and Fig. 10.

Picture (a)-(f) in Fig.7 denote two AGVs will travel along different routes as soon as they pass the intersection, but picture (g) and (h) represent two vehicles will travel on same route when they pass the intersection.

c) Type 3: As shown in Fig. 6, the node-occupancy collision happens if AGV_2 is on the route heading to its destination $GOAL_2$, while AGV_1 stops at its goal workstation $GOAL_1$, and the workstation $GOAL_1$ is included in the route of AGV_2.

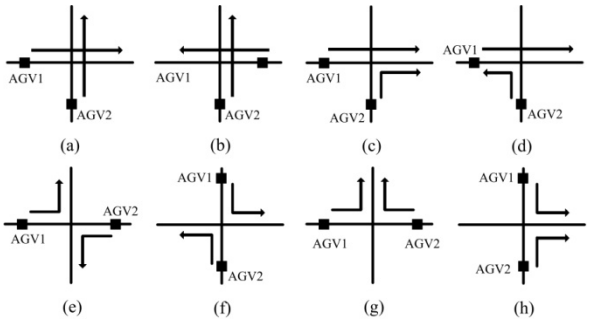

FIGURE 9. The cross collision (a)–(f) denote two AGVs will travel along different routes as soon as they pass the intersection, (g) and (h) represent two vehicles will travel on same route when they pass the intersection.

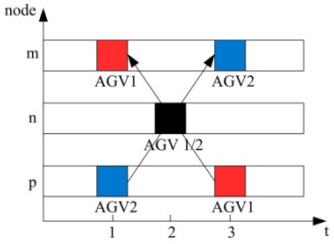

FIGURE 10. The time window of the cross collision.

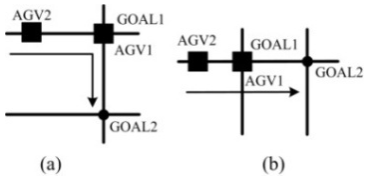

FIGURE 11. The node-occupancy collision (a) the first type of node-occupancy collision, (b) the second type of node-occupancy collision.

The examples are shown in Fig. 11, and the time window is shown in Fig. 12.

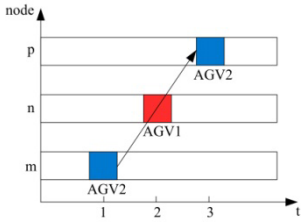

FIGURE 12. The time window of node-occupancy collision.

d) Type 4: As shown in Fig. 6, the shelf-occupancy collision happens if AGV_1 executes on-load task, and AGV_2 executes on-load or non-load task. When AGV_1 completes its current task, it will continue to perform next task. However, the shelf loaded upon the vehicle was parked on the destination (i.e. workstation SHELF). And the workstation SHELF is included in the route of AGV_2. The examples and the time window are shown in Fig. 13 and Fig. 14.

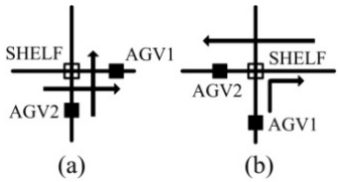

FIGURE 13. The shelf-occupancy collision (a) the first type of shelf-occupancy collision, (b) the second type of shelf-occupancy collision.

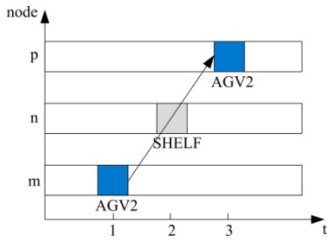

FIGURE 14. The time window of shelf-occupancy collision.

B. COLLISION DETECTIONS

The warehouse environment of the AGVs is regarded as a two-dimensional space, and each workstation has a unique ID.

Let us introduce some notations:

(*a*) S_m^n denotes the ID of a workstation m, which is included in the routes of AGV #n (i.e. AGV_n).

For example, in the collision type one (head-on collision): the ID of the goal workstation in AGV_1's route is S_α^1. This workstation is included in the route of AGV_2 with the ID described as S_β^2.

(*b*) S_α^1 denotes the ID of workstation α, which is included in the route of AGV_1. The ID of the workstation α's upstream workstation is $S_{\alpha-1}^1$, the ID of its downstream workstation is $S_{\alpha+1}^1$. And the route of the AGV_2 passes a workstation β with the ID S_β^2, the ID of the workstation β's upstream workstation is $S_{\beta-1}^2$, the ID of its downstream workstation is $S_{\beta+1}^2$.

(*c*) τ_m^n denotes the time that AGV_n arrives at workstation m.

For example, τ_α^1 and τ_β^2 denote the time that AGV_1 and AGV_2 arrive at the workstation α and β, respectively.

(*d*) t_m^n denotes the occupancy time period that AGV_n performs the task at the workstation m.

a) Type 1: For (a) and (b) in Fig. 7, if (1)-(3) are fulfilled, a head-on collision arises on the path $(\alpha(\beta), \alpha(\beta)+1)$.

$$\left|\tau_\alpha^1 - \tau_\beta^2\right| \le \delta \tag{1}$$

$$S_{\alpha+1}^1 = S_\beta^2 \tag{2}$$

$$S_\alpha^1 = S_{\beta+1}^2 \tag{3}$$

Where δ denotes the threshold for the collision detection. In order to ensure the traffic safety, it is set as a small constant based on the distance between the two adjacent workstations and the velocity and length of the AGV.

Note: If (4) and (5) are fulfilled, a head-on collision also arises on the workstation $\alpha\ (\beta) + 1$.

$$\left|\tau_{\alpha+1}^1 - \tau_{\beta+1}^2\right| \le \delta' \tag{4}$$

$$S_{\alpha+1}^1 = S_{\beta+1}^2 \tag{5}$$

Where δ' denotes the threshold, which is greater than δ.

b) Type 2: The cross collision satisfies (6) and (7).

$$S_\alpha^1 = S_\beta^2 \tag{6}$$

$$\left|\tau_\alpha^1 - \tau_\beta^2\right| \le \delta \tag{7}$$

c) Type 3: The node-occupancy collision satisfies (8) and (9).

$$S_\alpha^1 = S_\alpha^2 \tag{8}$$

$$0 < \tau_\alpha^2 - \tau_\alpha^1 < t_\alpha^1 + \delta \tag{9}$$

Note: α represents the ID of the goal node of the AGV_1, and this node is included in the route of AGV_2.

d) Type 4: The shelf-occupancy collision satisfies (10) and (11).

$$S_\alpha^1 = S_\alpha^2 \tag{10}$$

$$t_\alpha^1 + \delta < \tau_\alpha^2 - \tau_\alpha^1 \tag{11}$$

Note: α represents the ID of the workstation occupied by the shelf.

C. COLLISION SOLUTION STRATEGIES

The collision solution strategy plays an important role in the safety and efficiency of the automated warehouse system. The traditional solution to collision is waiting. In other words, the later arrived AGV waits at the upstream lane of the collision workstation (about L_s), until the former arrived AGV leaves. Where, L_s denotes the length of each AGV. For some collision types, e.g. head-on collision, the traditional waiting strategy cannot solve the collision between the vehicles, which can cause deadlock and other system failure.

Four collision-free solutions are proposed: (A) Selecting the candidate route; (B) Waiting for a short period of time before starting; (C) Modifying the route; (D) Re-dispatching tasks. The solution (D) is generally executed by a schedule algorithm. Based on the collision types mentioned above, the solution (A), solution (B), or solution (C) are used to solve the collision. Regardless of collision types, the schedule and/or the route of the former AGV remain unchanged, whereas the schedule and/or the route of the later AGV are changed.

As described in section III-B, one or two shortest routes can be found by the improved Dijkstra's algorithm. If the selected route for one AGV is in conflict with other AGV's, — the solution (A) will be employed — another candidate route will be chose as the selected one.

For the solution (A), only the schedule of the current route will be changed, the total travel time is as same as predetermined one.

The solution (B) is derived from traditional solution. Waiting before the collision section can reduce the system

dynamic performance. At the same time, it causes the waste of energy for waiting before the collision section — AGV can consume more electricity during the period of stop and restart. Therefore, we propose AGV_2 should put off starting at the start workstation for a predetermined period of time until AGV_1 leaves. Thus the route or the order of the workstations of the task remains unchanged, whereas the schedule or the time when the AGV_2 passes the workstations is modified.

The solution (C) is to modify the route and schedule of the later AGV. By using the improved Dijkstra's algorithm, if the route of AGV_1 collision with the route of AGV_2, the route of AGV_2 will be re-planned based on the new environment map excluding the congested sections.

The waiting time before staring is set as follows:

It is assumed that the designed speed of each vehicle is V_s; the vehicle length of each vehicle along the longitudinal direction is L_s; the safe distance between two AGVs is D_s. The waiting time in solution (A) is:

$$T_s = \frac{L_s + D_s}{V_s} \tag{12}$$

Based on the four collisions mentioned in section IV, the following conclusions are obtained through analyses and simulations. Following assumptions are used:

- The distance between two adjacent workstations is fixed as 1.5 m;
- The movement of an AGV is bi-directional when it moves through any path;
- There are two AGVs in the warehouse (i.e. AGV_1 and AGV_2);
- The AGV is regarded as a particle and can only execute one task at a time;
- The AGV moves in a two-dimensional plane and its speed is fixed as 0.3 m/s. So, the AGV spends 5.0 s between two workstations;
- The AGV spends 4.0 s in turning at a workstation.

Regardless of collision types, solution (A) is recommended to employ first. If there are no other candidate routes existing for AGV_2, or the collision cannot be resolved with solution (A), the solution (B) or solution (C) should be employed. Since the end workstation of each task is uncertain, so each AGV starts from a random node in this paper.

In warehouse system, throughput is often used to denote the productivity of a machine or a kind of process, which expresses in a figure-of-merit or a term meaningful in the given context, such as material output per hour/minute, cash turnover, and the number of orders shipped. Due to the merit of easy to measure, the warehouse system typically turns the throughput into the total time of a set of tasks. The reciprocal of the total time of a set of tasks is the freight volume per unit time, i.e. the throughput. For example, Warehouse has two idle AGVs, for example, the warehouse has two idle AGVs. Any AGV will not park and wait for another AGV to complete the task after it finishes its own. It will be regarded as a new idle AGV, which can execute the next task. Therefore, the total time of two AGVs refers to the evaluation index of the warehouse. The following analysis discusses all possible solutions based on different types of collision so as to determine the most appropriate solution for each type of collision.

Case 1, Head-on Collision: A piece of goods on the shelf should be transported from the "Manual Sorting Space" to the "Store Space" after the human finishes sorting. AGV_1 implements on-load task T_{11}: the start is "Manual Sorting Space (i.e. MMS)," and the goal workstation is 65; the shortest route (and arrival time of each workstation) is MMS(0s)→61(5s)→62(14s)→63(19s)→64(24s)→65(29s); The goods left on the workstation 57 should be stored in the warehouse. The AGV_2 execute non-load task T_{12}: the starting workstation is 57, and the destination workstation is 64; the shortest route (and arrival time of each workstation) is 57(0s)→68(5s)→67(14s)→66(19s)→65(24s)→64(29s). The server has planned the routes of the two AGVs, and it detects that there will be a head-on collision in the path (64, 65). If AGV_2's another candidate route exists, which has no conflict with the route of AGV_1, this alternative route is selected as the shortest route of the AGV_2.

Based on improved Dijkstra's algorithm [26], the alternative route of AGV_2 is 57(0s)→56(5s)→55(10s)→54(15s)→53(20s)→64(29s). The travel time is 29.0s. So, this is selected as the shortest route.

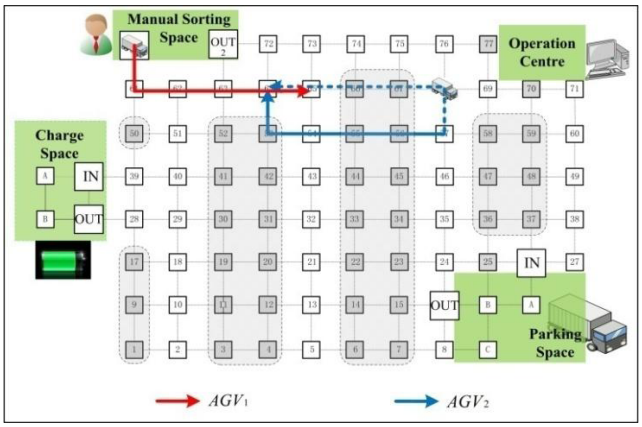

FIGURE 15. The map of head-on collision.

The map of the head-on collision is shown in Fig. 15, where the dotted line indicates the original route and the solid the altered one. The comparison results of the task time are shown in Fig. 16, where panning denotes the original schedule. The task time of solution (A) and solution (C) are both 58.0s, the task time of solution (B) is 88.0s. Therefore the solution (A) or the solution (C) is better. The time windows are shown in Fig. 17, which denotes the routes of AGV_1 and AGV_2.

Therefore, as for head-on collision, it is no mater that whether the collision occurs on the route or the workstation, the solution (A) or the solution (C) is the best approach. In other words, the collision workstations and its adjacent lanes are deleted, and then re-plan the route of AGV_2.

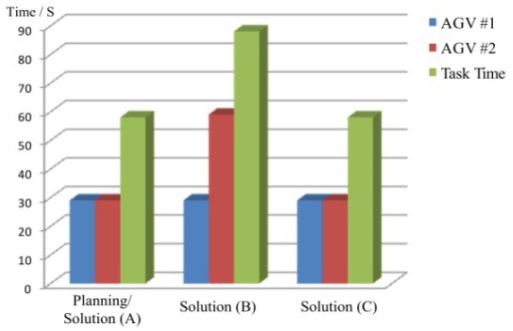

FIGURE 16. Comparison of the task time (head-on collision).

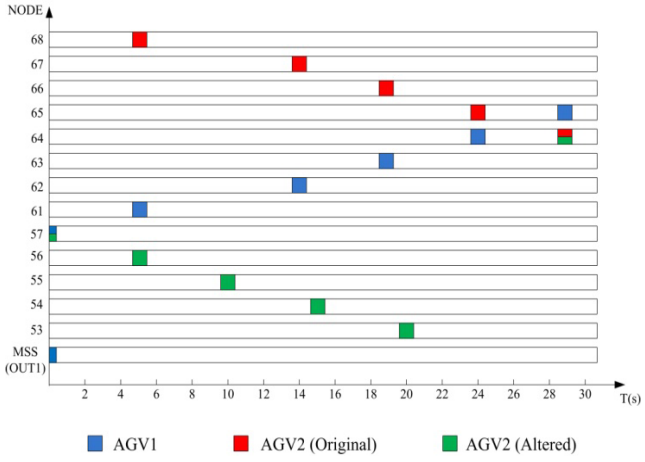

FIGURE 17. The time windows of two AGVs (head-on collision).

Case 2, Cross Collision: AGV_1 will pick up the shelf on workstation 31, and it is charging in "Charge Space." AGV_1 executes non-load task T_{21}: the start is "Charge Space (i.e. CS)," and the goal workstation is 45. The shortest route (and arrival time of each workstation) is CS(0s)→28(5s)→29(10s)→30(15s)→31(20s); AGV_2 will transport the goods on the shelf from workstation 3 to workstation 52, so AGV_2 implements non-load task T_{22}: the starting workstation is 3, and the goal workstation is 52. The shortest route (and arrival time of each workstation) is 3(0s)→11(5s)→19(10s)→30(15s)→41(20s)→52(25s). There is a cross collision at workstation 30. When the two vehicles meet, AGV_1 is on-load, AGV_2 is non-load.

The other candidate shortest route of AGV_2 does not exist. If the solution (B) is employed, AGV_2 should wait for about 3.0 s before stating. Based on the solution (C), the route of AGV_2 in solution (C) is determined at the modified environment map excluding the collision workstation.

The map of cross collision is shown in Fig. 18. The comparison results of the task time are shown in Fig. 19, where panning denotes the original schedule. Obviously, the task time by using solution (B) is 48.0s; the task time of solution (C) is 63.0s, which is longer than that of solution (B). Therefore the solution (B) is better and employed as the resolution method. The time windows of two AGVs are shown in Fig. 20.

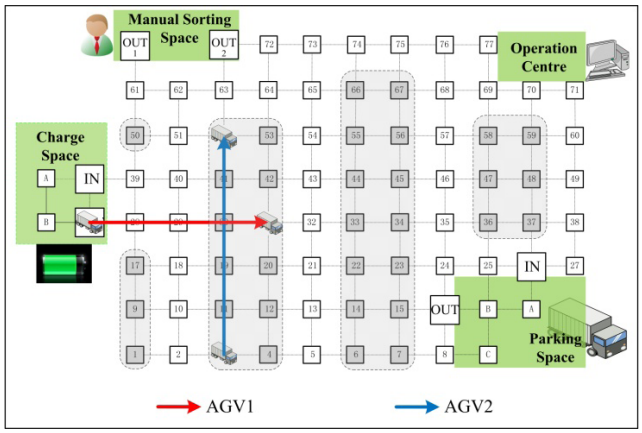

FIGURE 18. The map of cross collision.

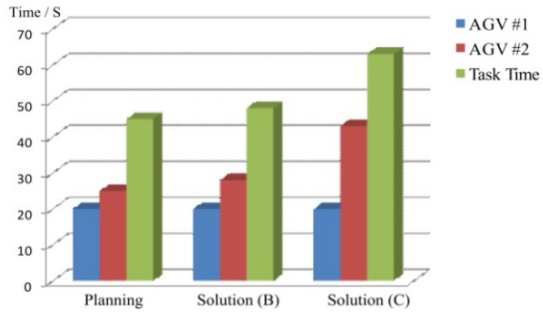

FIGURE 19. Comparison of the task time (cross collision).

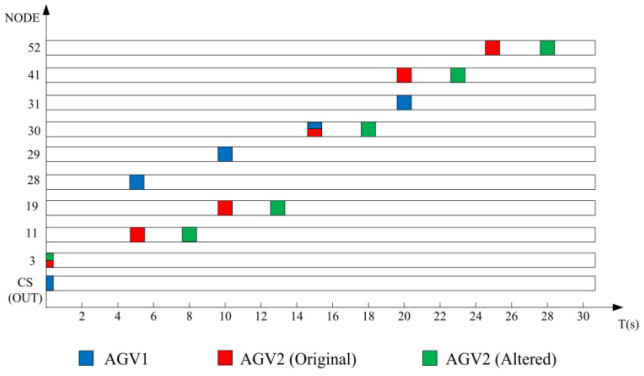

FIGURE 20. The time windows of two AGVs (cross collision).

Therefore, as for the cross collision, the solution (B) is the best approach. In other words, AGV_2 should wait for a period time before starting.

Case 3, Node-Occupancy Collision: The AGV_1 is charging at "Charge Space," and it will pick up the shelf on workstation 29. The AGV_1 executes non-load task T_{31}: the start is "Charge Space (i.e. CS)," the goal workstation is 29. Therefore, the route is CS(0s)→28(5s)→29(10s). The AGV_2 implements on-load task T_{32} : 2(0s)→10(5s)→18(10s)→29(15s)→40(20s)→54(25s)→62(30s)→63(39s).

The node-occupancy collision happens at workstation 29. Based on the improved Dijkstra's algorithm, the other

shortest route of AGV_2 exists, i.e. 2(0s)→3(5s)→11(14s)→19(19s)→30(24s)→41(29s)→52(34s)→63(39s). Therefore, this route is selected as the shortest route for AGV_2.

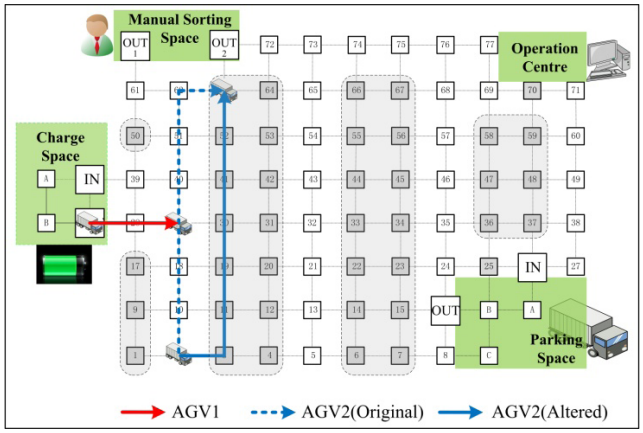

FIGURE 21. The map of node-occupancy collision.

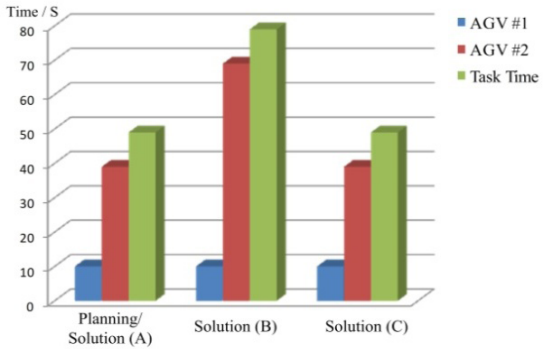

FIGURE 22. Comparison of The task time (node-occupancy collision).

The map of node-occupancy collision is shown in Fig. 21, where the dotted line indicates the original route and the solid line the altered one. The comparison results of the task time are shown in Fig. 22, where panning denotes the original schedule. The task time of the solution (B) is 79.0s; the task time of the solution (A) and the solution (C) are both 49.0s, which is shorter than the solution (B). Therefore the solution (A) or the solution (C) is better and employed as the resolution method. The time windows of two AGVs are shown in Fig. 23.

Therefore, as for node-occupancy collision, the solution (A) or the solution (C) is the best approach. In other words, the upper system deletes the nodes and its adjacent lanes in the map, and then re-plans the route of AGV_2.

Case 4, Shelf-Occupancy Collision: For the forth types of collision, the situation is more complex. All the same, it is supposed that there are two AGVs. It has been discussed in section II, an on-load AGV cannot pass through under the shelf, whereas a non-load AGV can.

If the AGV_2 is loaded with the shelf, the AGV is in on-load status. Because an on-load AGV cannot pass through under

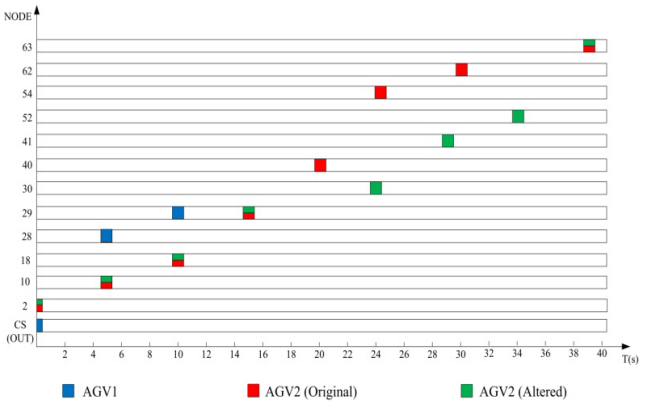

FIGURE 23. The time windows of two AGVs (node-occupancy collision).

the shelf, the upper system considers the workstation which a shelf as the obstacle. Only solution (A) or solution (C) can be used. The collision workstations and its adjacent lanes are deleted from the map, and then re-plan the route of the AGV_2.

The AGV_1 executes on-load task T_{41}: 44(0s)→45(5s)→46(10s)→47(15s). AGV_2 also executes on-load task T_{42}: MSS(0s)→72(5s)→73(10s)→74(15s)→75(20s)→76(25s)→77(30s)→69(39s)→58(44s)→47(49s)→36(54s). There is a shelf-occupancy collision in workstation 47 between AGV_1 and AGV_2. Since the AGV_1 and the AGV_2 are on-load when they arrive at workstation 47, the AGV_2 cannot pass though this workstation.

The other alternative route of AGV_2 exists, which is determined by improved Dijkstra's algorithm, i.e. MSS(0s)→63(5s)→52(10s)→41(15s)→30(20s)→31(25s)→32(30s)→33(39s)→34(44s)→35(49s)→36(54s). Therefore, this is selected as the shortest route of AGV_2. The map of shelf-occupancy collision is shown in Fig. 26, where the dotted line indicates the original route and the solid the altered. The comparison results of task time are shown in Fig. 27, where panning denotes the original schedule. The task time of the solution (A) and the solution (C) are both 69.0s; the task time of the solution (B) is 99.0s, which is longer than the solution (A) or the solution (C). Therefore the solution (A) or the solution (C) is better and employed as the resolution method. The time windows of two AGVs are shown in Fig. 28.

If the AGV_2 is not loaded with the shelf, the AGV is on non-load status. The server considers the workstation with the shelf as the available workstation, so that the AGV_2 can pass though.

Ditto, AGV_1 executes on-load task T_{41}, the starting node is 44, and the destination node is 47. The shortest route (and arrival time of each node) is: 44(0s)→45(5s)→46(10s)→47(15s). Upon completion of the on-load task, it will perform the next task immediately. It leaves immediately after put down the shelf at the workstation 47. Because a piece of goods on the shelf should be transported from the "Manual Sorting Space" to the "Store Space" after the human finishes sorting. The AGV_2 executes non-load task T_{42}, the start is "Manual Sorting

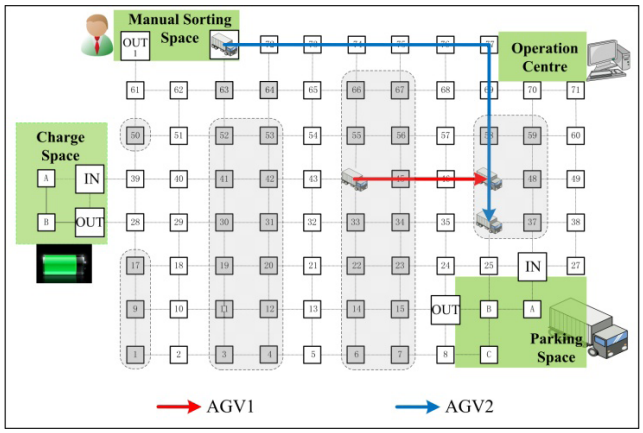

FIGURE 24. The map of shelf-occupancy collision (AGV cannot pass).

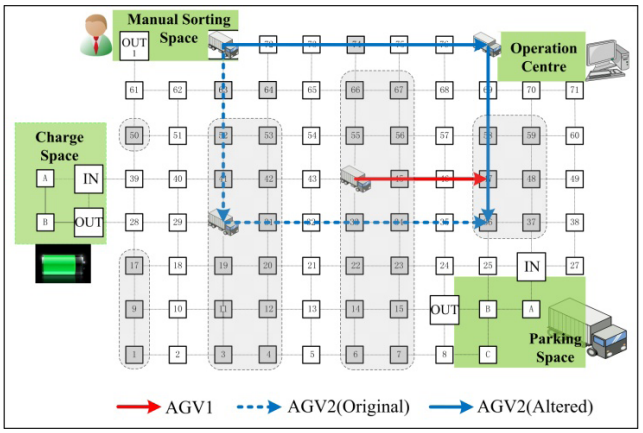

FIGURE 26. The time windows of two AGVs (shelf-occupancy collision).

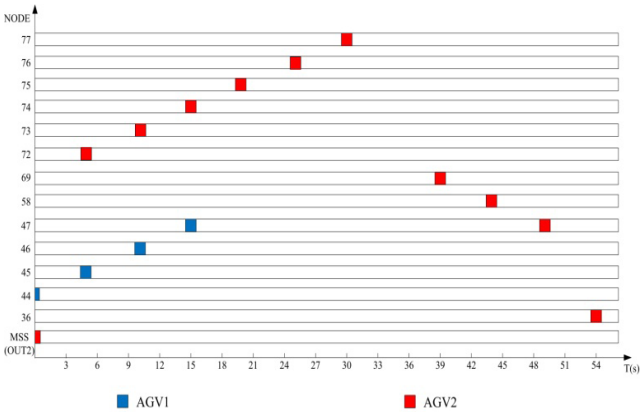

FIGURE 25. Comparison of the task time (shelf-occupancy collision).

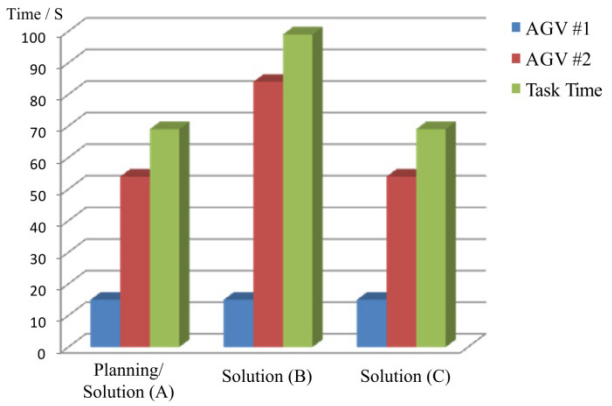

FIGURE 27. The map of shelf-occupancy collision (AGV can pass).

Space (i.e. MMS)," and the destination workstation is 36. The shortest route (and arrival time of each workstation) is: MSS(0s)→72(5s)→73(10s)→74(15s)→75(20s)→76(25s)→77(30s)→69(39s)→58(44s)→47(49s)→36(54s).

The server detected that, there is a shelf-occupancy collision on the workstation 47 between AGV_1 and AGV $_2$according to the alternative routes. AGV_2 executes non-load task, so the AGV_2 is non-load when it arrives at workstation 47. The AGV_2 can pass though this workstation. In this case, So, in this case, the solution (A) is the best approach. The other alternative route of AGV_2 is selected as the shortest route. The map of shelf-occupancy collision is shown in Fig. 24. The time windows of two AGVs are shown in Fig. 25.

To sum up, the solution (A) or (C) is suitable for solving head-on collision. The other alternative route should be selected, or the route of AGV_2 should be re-planned. The solution (B) is suitable for solving cross collision, the AGV_2 delay starting time of 3.0s. The solution (A) or (C) is suitable for solving node-occupancy collision. The other alternative route should be selected, or the route of AGV_2 should be re-planned. As for shelf-occupancy collision, the solution depends on AGV's status before collision section: if AGV is non-loaded, it can pass directly; if AGV is on-load, Solution (A) or Solution (C) should be adopted.

VI. CASE STUDY

In this section, we focus on a real automated warehouse depicted in Fig. 2. There are several available AGVs, and the arrowed lines represent the routes of AGVs. The case study is developed by using *Visual C++*, and the OS is *Windows10*. The environment map structure includes the two-dimensional spatial coordinates of the workstations and the adjacency relation between the workstations.

On this case, four AGVs are used. At the first place, the sever plans the initial routes for four AGVs respectively by using single AGV routing algorithm. AGV_1 picks up goods on workstation 33 after charging. AGV_1 executes non-load task T_1: CS(0s)→28(5s)→29(10s)→30(15s)→31(20s)→32(25s)→33(30s); AGV_2 transports the goods from workstation 60 to workstation 28, AGV $_2$implements on-load task T_2: 60(0s)→40(5s)→29(10s)→28(15s); AGV_3 travels from workstation 3 to workstation 63 to pick down the goods, AGV_3 implements on-load task T_3: 3(0s)→11(5s)→19(10s)→30(15s) →41(20s)→52(25s)→63(30s); AGV_4 starts from "Parking Space (i.e. PS),"

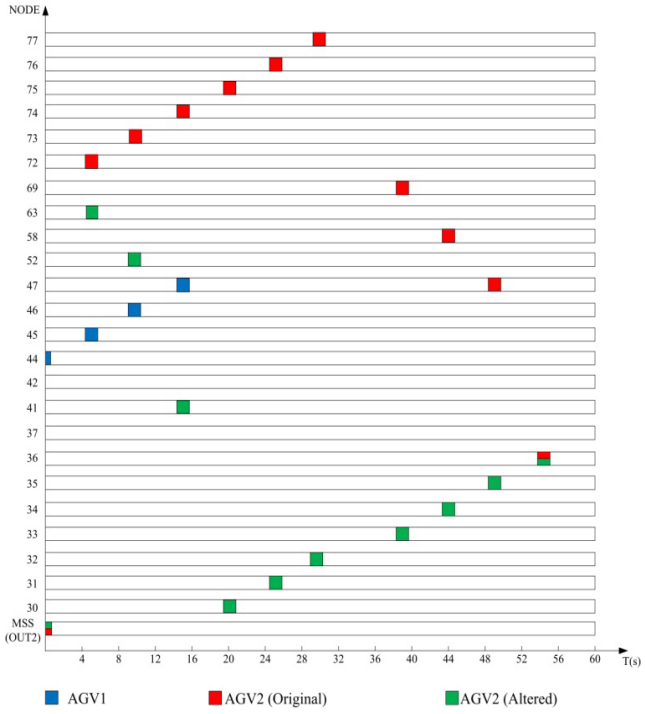

FIGURE 28. The time windows of the shelf-occupancy collision (AGV can pass).

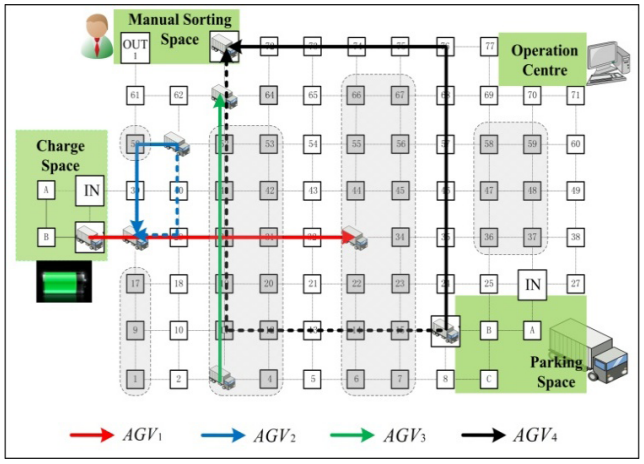

FIGURE 29. The routes of multiple AGVs.

and then it picks up goods from "Manual Sorting Space (i.e. MMS)," therefore AGV_4 implements non-load task T_4: PS(0s)→15(5s)→14(10s)→13(15s)→12(20s)→11(25s)→19(34s)→30(39s)→41(44s)→ 52(49s)→ 63(54s)→MSS(59s).

At the second place, the server implements the collision detection iteratively.

- Firstly, it compares the coordinates and the time window of the workstations in the routes of AGV_1 and AGV_2. There is a head-on collision on the path (28, 29) between AGV_1 and AGV_2, which can be solved by using the solution (A). The other shortest routes of AGV_2 is determined by the improved Dijkstra's algorithm, which is 60(0s)→59(5s)→39(14s)→28(19s).

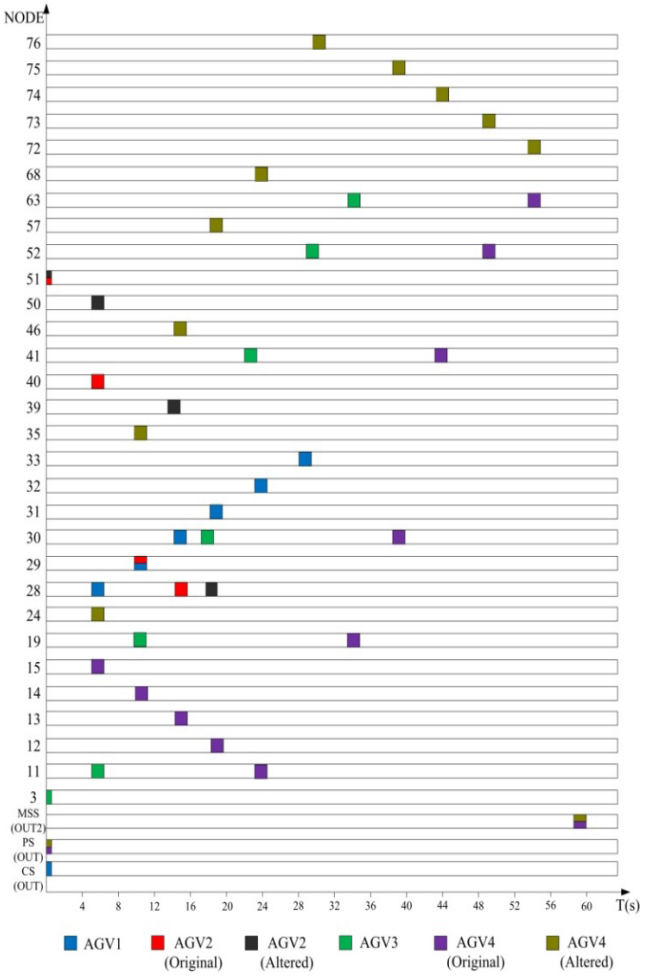

FIGURE 30. The time windows of multiple AGVs.

- Secondly, the route of AGV_3 is compared with the routes of AGV_1 and AGV_2orderly. There is a cross collision on workstation 30 between AGV_3 and AGV_1. The solution (B) is employed for AGV_3, which should wait for about 3.0s before stating.
- Thirdly, the server compares the route of AGV_4 with the routes of AGV_1, AGV_2 and AGV_3 accordingly. Because AGV_3 executes on-load task, it puts down the shelf on workstation 63. There is shelf-occupancy collision on workstation 63 between AGV_3 and AGV_4, which can be solved by using the solution (A). The other shortest routes of AGV_4 in solution (A) are determined by using the improved Dijkstra's algorithm, i.e. PS(0s) →24(5s) → 35(10s) →46(15s) → 57(20s) →68(25s)→76(30s) →75(39s)→74(44s)→73(49s)→72(54s)→ MSS(59s). Then the server compares this route again with the ones of AGV_1, AGV_2 and AGV_3 accordingly until no conflicts exist anymore.

The above progresses are completely offline in the server. The server compares the routes between two AGVs. After there are no conflicts among the possible routes, the server dispatches these routes to AGVs through

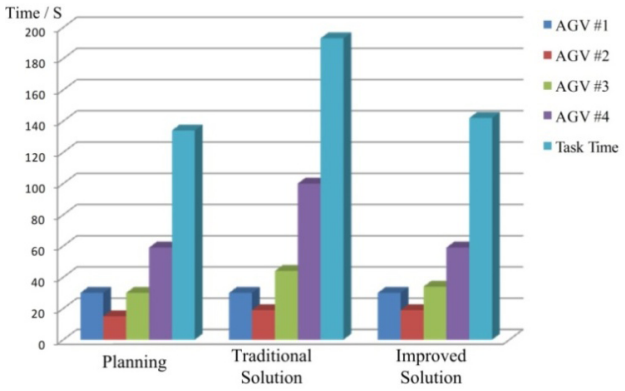

FIGURE 31. Comparison of the task time.

wireless communication. The routes of multiple AGVs are shown in Fig. 29, where the dotted line indicates the original route and the solid the altered. The time windows of multiple AGVs are shown in Fig. 30. The comparison results of the task time are shown in Fig. 31, where panning denotes the original schedule. The task time of the improved solution is142.0s, the task time of the traditional solution is more than 193.0s, which is longer than the improved solution. Therefore the improved solution is better and employed as the resolution method.

As shown in the Figures, obviously, taking into account of the task time, the solution (A) is employed for AGV_2 and AGV_4, but the solution (B) is employed for AGV_3. The task time is shortened compared to the traditional solution strategy.

VII. CONCLUSION

What motivated our work is the fact that the increasing use of AGV in the automated warehouse. This paper studied the collision-free route planning problem for multi-AGVs system in warehouse.

The environment of the warehouse is divided into five areas. The map was modeled by using grid method. Firstly, the predetermined route of each task is planned by using improved Dijkstra's algorithm. Secondly, the collision detection and classification are implemented by the server according to the ID and the time window of workstations in the routes. The potential collisions are classified into four types. Finally, three solutions are proposed, i.e. solution (A): Select the candidate route; solution (B): The later AGV waits before staring; solution (C): Modify the routes of later AGV. And each collision classification followed by one or two cases. Based on these cases, we select the corresponding strategy for different collision classification.

We improve the traditional collision waiting solution. Besides, the status of the AGV is included in the paper. Based on the status of AGV, a new collision type is proposed, and different solutions be used. The case study demonstrates that the proposed route planning approach increases the efficiency of the automated warehouse system. It provides a technical guidance for routing for multi-AGVs system in warehouse.

REFERENCES

[1] N. Wu and M. Zhou, "Shortest routing of bidirectional automated guided vehicles avoiding deadlock and blocking," *IEEE/ASME Trans. Mechatron.*, vol. 12, no. 1, pp. 63–72, Feb. 2007.

[2] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 6, pp. 898–912, Dec. 2003.

[3] M. Saska, M. Macas, L. Preucil, and L. Lhotska, "Robot path planning using particle swarm optimization of Ferguson splines," in *Proc. IEEE Conf. Emerg. Technol. Factory Autom.*, Prague, Czech Republic, Sep. 2006, pp. 833–839.

[4] R. Kristiansen, "Motion planning for robotic manipulators," Ph.D. dissertation, Fac. Eng. Sci., King's College London, London, U.K., 2014.

[5] B. Yang and W.-J. Liu, "A improved method of robot's path planning based visibilty graph," *Comput. Knowl. Technol.*, vol. 5, pp. 434–435, Feb. 2009.

[6] X. Fan, X. Luo, S. Yi, S. Yang, and H. Zhang, "Optimal path planning for mobile robots based on intensified ant colony optimization algorithm," in *Proc. IEEE Int. Conf. Robot., Intell. Syst. Signal Process.*, vol. 1. Oct. 2003 pp. 131–136.

[7] K. Zheng, D. Tang, W. Gu, and M. Dai, "Distributed control of multi-AGV system based on regional control model," *Prod. Eng.*, vol. 7, no. 4, pp. 433–441, Jul. 2013.

[8] M. S. Sedehi and R. Z. Farahani, "An integrated approach to determine the block layout, AGV flow path and the location of pick-up/delivery points in single-loop systems," *Int. J. Prod. Res.*, vol. 47, no. 11, pp. 3041–3061, 2009.

[9] G. Li, Y. Tamura, A. Yamashita, and H. Asama, "Effective improved artificial potential field-based regression search method for autonomous mobile robot path planning," *Int. J. Mechatron. Autom.*, vol. 3, pp. 141–170, Jan. 2013.

[10] M.-C. Chen, Y.-H. Hsiao, H. Reddy, and M. K. Tiwari, "A particle swarm optimization approach for route planning with cross-docking," in *Proc. 7th Int. Conf. Emerg. Trends Eng. Technol. (ICETET)*, Kobe, Japan, Nov. 2015, pp. 1–6.

[11] S. Ravizza. "Control of automated guided vehicles (AGVs)," Ph.D. dissertation, ETH Zurich, Zürich, Switzerland, 2009.

[12] N. Wu and M. Zhou, "Modeling and deadlock control of automated guided vehicle systems," *IEEE/ASME Trans. Mechatronics*, vol. 9, no. 1, pp. 50–57, Mar. 2004.

[13] N. Wu and M. Zhou, "Modeling and deadlock avoidance of automated manufacturing systems with multiple automated guided vehicles," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1193–1202, Dec. 2005.

[14] A. W. ten Mors, "The world according to MARP," Ph.D. dissertation, EWI, Delft Univ. Technol., Delft, The Netherlands, Mar. 2010.

[15] S. Hao *et al.*, "An optimal task decision method for a warehouse robot with multiple tasks based on linear temporal logic," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Banff, AB, Canada, Oct. 2017, pp. 1453–1458.

[16] Q. Sun, H. Liu, Q. Yang, and W. Yan, "On the design for AGVs: Modeling, path planning and localization," in *Proc. Int. Conf. Mechatron. Autom. (ICMA)*, Beijing, China, Aug. 2011, pp. 1515–1520.

[17] M. Karova, D. Zhelyazkov, M. Todorova, I. Penev, V. Nikolov, and V. Petkov, "Path planning algorithm for mobile robot," in *Proc. Int. Conf. Appl. Comput. Sci.*, 2015, pp. 283–287.

[18] A. Vatavu, A. D. Costea, and S. Nedevschi, "Modeling and tracking of dynamic obstacles for logistic plants using omnidirectional stereo vision," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Hamburg, Germany, Sep./Oct. 2015, pp. 3552–3558.

[19] J. Rong, "Research on collision avoidance strategy of AGV," in *Proc. Int. Conf. Manuf. Construct. Energy Eng. (MCEE)*, 2017.

[20] C. Liu, J. Tan, H. Zhao, Y. Li, and X. Bai, "Path planning and intelligent scheduling of multi-AGV systems in workshop," in *Proc. 36th Chin. Control Conf.*, Dalian, China, Jul. 2017, pp. 2735–2739.

[21] R. Yuan, T. Dong, and J. Li, "Research on the collision-free path planning of multi-AGVs system based on improved A* algorithm," *Amer. J. Oper. Res.*, vol. 6, pp. 442–449, Oct. 2016.

[22] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Hierarchical traffic control for partially decentralized coordination of multi AGV systems in industrial environments," in *Proc. IEEE Int. Conf. Robot., Autom. (ICRA)*, Hong Kong, May/Jun. 2014, pp. 6144–6149.

[23] H.-S. Ahn, Y. Chenz, and K. L. Moore, "Multi-agent coordination by iterative learning control: Centralized and decentralized strategies," in *Proc. IEEE Int. Symp. Intell. Control (ISIC)*, Denver, CO, USA, Sep. 2011, pp. 28–30.
[24] A. W. T. Mors, "Conflict-free route planning in dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot., Syst.*, San Francisco, CA, USA, Sep. 2011, pp. 2166–2171.
[25] Q. Guo, Z. Zheng, and X. Yue, "Path-planning of automated guided vehicle based on improved Dijkstra algorithm," in *Proc. 29th Chin. Control Decision Conf.*, Chongqing, China, May 2017, pp. 7280–7285.
[26] Z. Zhang, Q. Guo, and P. Yuan, "Conflict-free route planning of automated guided vehicles based on conflict classification," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Banff, AB, Canada, Oct. 2017, pp. 1459–1464.

ZHENG ZHANG (S'17) was born in Hohhot, China, in 1990. He received the B.S. degree in automation from the Beijing University of Chemical Technology in 2014, where he is currently pursuing the M.S. degree majoring in control science and engineering.

His research interest includes intelligent control, robot path planning, and trajectory tracking control. He has published two articles in international conference: Path-planning of automated guided vehicle based on improved Dijkstra's algorithm at the IEEE CCDC 2017 and Conflict-free Route Planning of AGVs Based on Conflict Classification at the IEEE SMC 2017.

Mr. Zhang had participated the project Robot Path Planning Algorithm and Control System Development, which in cooperation with Beihang University, Beijing, China.

QING GUO was born in Weifang, Shandong, China. She received the B.S. degree in automation and the M.S. degree in control science and engineering from the Beijing University of Chemical Technology in 1992 and 1998, respectively; and the Ph.D. degree in engineering from the Nara Institute of Science and Technology in 2009.

From 1998 to 2014, she was a Lecturer with the School of Information Science, Beijing University of Chemical Technology. Since 2015, she has been a Vice Professor with the School of Information Science, Beijing University of Chemical Technology. Her research interests include polymer quality controlling, optimization and control of polymerization process, and robot trajectory optimization and control.

She has published over 10 articles and two monographs. The main projects she undertaken include The Development of the Board Counting System of the Wood Industry, from 2014 to 2015, Study on Extraction Mechanism and Process Optimization of Broken-Mass-Transfer of Traditional Chinese Medicine with Ultrasonic Frequency Conversion Turbulence, from 2012 to 2015, and Robot Path Planning Algorithm and Control System Development, from 2016 to 2017.

JUAN CHEN was born in Harbin, Heilongjiang, China, in 1961. She received the B.S. degree in automation from the College of Northeastern Heavy Machinery, Yanshan University, Qinhuangdao, China, in 1983, the M.S. degree in control science and engineering from the Harbin Institute of Technology, Harbin, in 1999, and the Ph.D. degree in control science and engineering from the Beijing University of Chemical Technology, Beijing, China, in 2006.

She is currently a Professor and a Doctoral Tutor of the School of Information Science, Beijing University of Chemical Technology. Her research interests include intelligent detection and intelligent control, system modeling and control, motion control, and trajectory tracking control.

She has authored over 80 articles and holds three invention patents. Her awards and honors include the Henan Province Science and Technology Progress Award in 1987 and the National Fund for Natural Science Fund Project Award in 2014 and 2018.

PEIJIANG YUAN received the B.S. degree in automation from the Department of Automation, Tsinghua University, Beijing, China, in 1997, and the Ph.D. degree in electronic and computer engineering from the University of Western Ontario, London, ON, Canada, in 2005.

He joined Beihang University, Beijing, in 2009, where he is currently an Associate Professor with the Intelligent Technology and Robotics Research Center. He has authored over 50 papers published in international journals and conference proceedings. His current research interests include aviation manufacturing robots, robot localization and perception, intelligent control, and machine vision.

Dr. Yuan has served several international conferences as a program committee member. He was the Program Committee Co-Chair of the 2010 IEEE World Congress on Computational Intelligence and a Committee Member of the 2011 IEEE Robotics and Automation and Intelligent Robots and Systems.