Received April 4, 2020, accepted April 24, 2020, date of publication April 29, 2020, date of current version May 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2991190

Dynamic Resource Reservation Based Collision and Deadlock Prevention for Multi-AGVs

YUNLONG ZHAO, XIAOPING LIU, GANG WANG, SHAOBO WU, AND SONG HAN
Automation School, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Xiaoping Liu (liuxp@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61902038 and Grant 61771069.

ABSTRACT Automated guided vehicles (AGVs) are widely used for material handling in warehouses and automated production lines due to their high efficiency and low cost. However, AGVs usually interact with each other because of the restricted capacity of the layout. Although many algorithms have been proposed to address the problem, most of them are inefficient for collision and deadlock avoidance in dynamic environments. This paper proposes a dynamic resource reservation (DRR) based method supporting time-efficient scheduling and collision avoidance of multiple AGVs. In this method, the layout is divided into square blocks with the same size that are abstracted as points in the undirected graph. In order to solve the collision and deadlock problem dynamically, the shared resource points of each vehicle are extracted from their guide paths in real time. Unlike the traditional approaches most of which adopt a static point occupation policy, DRR exploits dynamical reservations of shared resource points to change AGV movement states for avoiding collisions and deadlocks, resulting in better time efficiency. We jointly implement the algorithm on both central and local controllers. Extensive simulation results demonstrate the feasibility and efficiency of the proposed collision and deadlock prevention method.

INDEX TERMS Automated guided vehicles, deadlock and collision prevention, resource reservation, shared resource points.

I. INTRODUCTION

In recent years, automated guided vehicles (AGVs) gradually play a more and more important role in warehouses and industries [1]. Introduction of automated guided vehicles (AGVs) for material handling and visual grab in warehouses reduces operation cost and improves the throughput and efficiency of goods transportation [2], [3]. However, AGVs very often interact with each other during movements. It is necessary to use an effective on-line supervisory control strategy to solve collision and deadlock problems in the system layout.

Algorithms for dispatching, routing, scheduling, collision and deadlock avoidance can be centralized or decentralized. [4]. In the decentralized approaches, agents can obtain information generally from their neighbors. The main advantages of such a solution are its reliability, flexibility and scalability, but its security and efficiency are relatively low [5]. Digani *et al.* [6] propose a hierarchical traffic control algorithm, which implements path planning on a two-layer architecture. The first layer is a topological graph of the plant where each node is a macro-area of the environment. The second layer is the real route map on which the AGVs move. The coordination among the AGVs is obtained by exploiting local negotiation (i.e. decentralized coordination). Zaremba *et al.* [7] present a new approach for distributed control of automatic guided vehicle systems which uses max-algebra formalism to model system operations. Its novelty is that it replaces the classical problem of vehicle real-time scheduling with the problem of distributed time setting. Draganjac *et al.* [8] propose a strategy for coordinating vehicles in multi-AGV systems, which integrates autonomous motion planning and decentralized decision making. The coordination algorithm ensures reliable resolution of different conflict situations among the vehicles based on a removal strategy and a private zone mechanism. Cancemi *et al.* [9] propose a distributed method to coordinate heterogeneous vehicles in a large-scale industrial environment. The method consists of a resource-sharing protocol and a re-planning strategy. Based on the distributed resource-sharing protocol and the re-planning strategy, vehicles use the map information to calculate the paths and coordinate the motions. Roszkowska [10] presents a distributed

The associate editor coordinating the review of this manuscript and approving it for publication was Shaohua Wan.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

control mechanism for coordinating heterogeneous mobile robots. The method prevents collisions and deadlocks by reducing the speed of some robots. In [11], the authors propose a shared resource based decentralized coordination algorithm for safe and efficient managements of a group of mobile robots. Pérez *et al.* [12] propose a decentralized method based on hierarchical path planning and mutually exclusive resource for coordination. When a vehicle dynamically plans its own trajectory, the other vehicles are treated as static obstacles. In [13], the authors propose a distributed predictive path following controller with arrival time awareness for multiple waterborne automated guided vessels. The work presented in [14] describes a distributed control system for coordinating vehicles moving along a network of interconnected predefined path areas. The efficiency of the system has been demonstrated with laboratory robots. However, since the vehicle is not autonomous but controlled by area controllers, this approach requires additional infrastructures to be installed in the work environment.

In the centralized control approaches, all information related to AGVs and the states of the transportation systems are stored and computed in central controllers. The centralized systems generally have high efficiency, but incur high time complexity. The methods based on time windows to avoid collisions and deadlocks are currently in use. In [15], the authors use time windows to solve the shortest path problem dynamically. Since the number of active vehicles and the corresponding missions change with time, the proposed routing method makes the determined shortest path feasible by time window elongation, resulting in collision-and-deadlock-free travelling for all active vehicles. However, this method is usually used in topologies with unidirectional lanes. Zhang *et al.* [16] propose a collision-free routing method. The route is predetermined by the improved Dijkstra algorithm. Four alternative methods are proposed to avoid each type of collision. Fan *et al.* [17] propose a heuristic algorithm to search for free time windows of the blocks in the selected paths for each AGV, and select the path with the earliest arrival time window as its scheduled path. In dynamic environments, the calculation of time windows subjects to many factors, such as acceleration and deceleration time of AGVs, external obstacles (e.g. people), which makes it difficult to calculate the time windows accurately. Imprecise time windows and external obstacles can lead to unpredictable collisions. Xin *et al.* [18] propose a time-space model and use a dedicated algorithm to minimize the cycle time of operation tasks for each AGV while avoiding collisions. Zone control is a simple and effective method to avoid collisions and deadlocks, which is widely used in AGV systems [19]. The guide path is composed of a series of zones that represent workstations, intersections of several lanes, or simple parts of a straight lane. Vehicles access to any zone must be authorized in advance by the central controller to avoid collisions and deadlocks [20], [21]. In [4], a chain of reservation (COR) based coordination method is proposed. In the method, the environment is divided into squares. The path points of AGVs are reserved at the beginning of the task to avoid deadlocks and collisions. Ho *et al.* [22] propose a dynamic zone strategy which based on two procedures to prevent collisions and maintain the load balance between the vehicles of different zones. Ho *et al.* [23] propose a new zone control strategy which uses zone partition design and dynamic zone control method to prevent collisions. However, the strategy can only be used in a system with single-loop guide paths. Li *et al.* [24] adopt a strategy to find k-th shortest routes, from which a collision-and-deadlock-free route can be selected to improve the system efficiency. In [25], two effective traffic-control policies with polynomial-time complexity are proposed to avoid collisions and deadlocks. The authors exploit the authorization-based point control policy to avoid collisions. And the deadlock recovery policy is to eliminate the cycle in the full graph by re-planning one of the first AGVs that causes a deadlock. The design of safety zones is the key of the zone control. Too large or too small security zone division can affect the efficiency of collision and deadlock resolutions. Petri net, as an effective tool for modeling and analyzing deadlock problems, are often used in AGVs control systems [26]–[28]. In [29], the authors use Petri nets to decouple the upper level planning from the lower level logical control in an AGV system. In [30], the authors use time Petri net decomposition method to conduct AGV scheduling and conflict-free path planning. Uzam [31] propose an optimal deadlock-prevention policy which is based on the division of reachable states. However, the obtained policy is not structurally optimal. Wu *et al.* [32] present a deadlock-prevention controller based on resource-oriented Petri net (ROPN) model and zone control strategy. By taking advantage of ROPN, the relationship between bad markings and structural properties is revealed. In [33], an ordinary Petri-net (PN) based approach is proposed to design a programmable logical controller (PLC) for collision prevention. The authors use an ordinary PN to model the AGV system and its controllers. Then, the method to automatically translate a closed-loop PN into a ladder diagram program is proposed. Petri net, as a powerful tool for discrete-event systems (DESs) analysis and synthesis, is often used in investigations of AGV control system properties [15]. However, Petri net-based methods can cause state explosions since at each time a robot needs to check the whole state space to determine whether it is safe to move back to its current state. As mentioned above, many different control policies are proposed to realize collision-and-deadlock-free travelling in transport systems with multi-AGVs. However, most of the existing approaches are less efficient for collision and deadlock avoidance in dynamic environments.

The aim of this paper is to improve the efficiency of the collision and deadlock resolution in dynamic environments. In this work, a dynamic resource reservation based (DRR) method to prevent collisions and deadlocks is proposed. The shared resource points of each AGV are calculated in real time. We exploit two conditions to make decisions for dynamical resource reservations. The main novelty of this paper lies in dynamic reservations of resource points, which can ensure AGVs' collision-and-deadlock-free travelling while improving the time efficiency. The control algorithm is implemented on both central and local controllers.

The rest of the paper is structured as follows. Section 2 presents the system description. Section 3 introduces the main design of our proposed method. In section 4, the simulation results are presented. Section 5 draws the conclusions.

II. THE ENVIRONMENT DESCRIPTION

An industrial application often involves a number of AGVs delivering goods and materials among workstations and storage pipes. We consider R AGVs that are moving in the same environment, which can be denoted as $R_r = \{r_r : 1 \leq r \leq R\}$. The environment is completely known and structured.

For ease of modeling, the layout is divided into square blocks with the same size, as shown in Fig. 1. Yellow blocks represent the parking places. Green blocks represent the passable guide path blocks. Each passable path block is assigned to specific traveling directions represented as arrows in the block. Vehicles can stay or rotate at the defined areas. They are positioned at the center of these fields. From this point of view, the traveling of AGV can be described as moving from one block to the next block. Note that AGVs cannot move to the next blocks during the deceleration time, which can be realized by designing the AGV speed and the block size reasonably in reality.

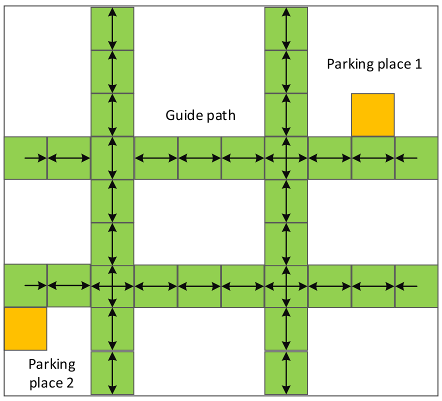

FIGURE 1. The layout divided into square blocks.

Moreover, considering the mathematical analysis and supervisory control algorithm design, the layout of a multi-AGVs transport system can be further abstracted into an undirected graph $G = (N, A)$. The blocks are abstracted as points and the adjacent points are connected by virtual directed arcs, as shown in Fig. 2. The green points represent the center of passable guide path blocks. The yellow points represent the center of parking places.

The set of all the points in the graph is denoted as $N = \{n_x : 1 \leq x \leq m\}$. Each vehicle has its own parking place.

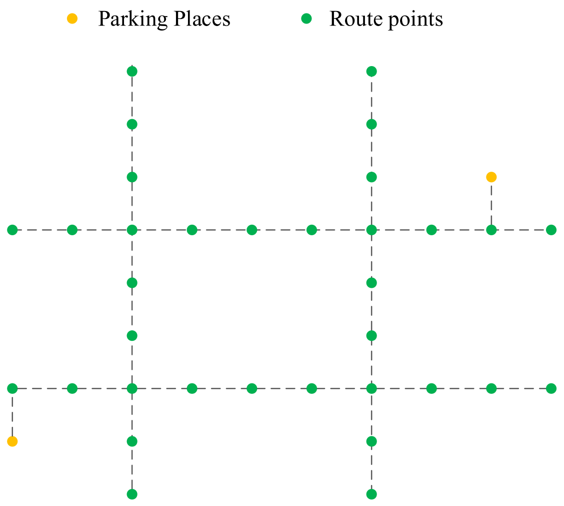

FIGURE 2. The topology of the layout.

The parking places are the origins of motions and all AGVs wait for a transport task at their parking places. The set of parking places is described as $N^o = \{n_x : n_x \in N, R \leq x < card(N)\}$. It means that the number of AGVs can be equal or less than the number of parking places. And there exists at least one point that is not a parking place. The resource points represent passable guide path points, which can be denoted as $N^s = \{n_x : n_x \in N, n_x \notin N^o\}$. Note that the parking place of each AGV cannot be a resource point. Each AGV can move from parking places to resource points and vice versa. The set of active tasks is represented as $M_a = \{m_i : m_i \in M\}$, where M is a set of all tasks generated by the transport system.

Assumption 1: Each parking place $n_x \in N^o$ is adjacent to at least one resource point $n_y \in N^s$.

Assumption 2: All AGVs in the system cannot change their transport routes.

In most of the coordinate algorithms, the transport routes of AGVs are assumed unchangeable [2] [12]. Because the path re-planning of AGVs may lead to new collisions and deadlocks. In a sense, re-planning does not resolve the collisions and deadlocks completely. And it does increase the computing complexity of the system.

III. DRR-BASED COLLISION AND DEADLOCK PREVENTION

In this section, a collision and deadlock prevention method is proposed to eliminate collisions and deadlocks in the transport system. First, the calculation of shared resource points is introduced. Then, we elaborate the AGVs how to reserve the resource points according to two conditions. Lastly, the implementation of two control policies on both central and local controllers is described in detail.

A. SHARED RESOURCE CALCULATION

AGVs move along the guiding paths and occupy spaces. During the movement, AGVs often collide with each other because of the limited capacity of the area. When a task is assigned to an AGV, it reserves, occupies, and releases the resource points. Each AGV can occupy only one resource point at a time.

Definition 1: The state of AGV r_a in the system can be idle ($s_a = 0$), moving ($s_a = 1$) and waiting ($s_a = 2$).

$s_a = 0$ indicates that r_a have no task to execute. The moving state $s_a = 1$ indicates that r_a is on the move. $s_a = 2$ indicates that r_a is waiting for the desired resource points that have been reserved by another AGV.

Definition 2: The state of the resource point n_x in the system could be idle ($s(n_x) = 0$), occupied by AGV r_a ($s(n_x) = r_a$).

$s(n_x) = 0$ indicates that there is no AGV reserving n_x. $s(n_x) = r_a$ indicates that n_x has been reserved by AGV r_a and the others cannot pass by n_x.

The tasks that AGVs have to accomplish correspond to transporting an item from a pick-up point to a target point. Thus, once a task is assigned to AGV r_a, the path p_a that the vehicle has to follow is computed. A path is simply a sequence of adjacent points, which can be denoted as $p_a = \{n_x, n_{x+1}, \cdots\}$.

In the transport system, AGVs can send their real-time information, including their locations, to the central controller. From this information, we can determine the state with respect to the motion progress of AGVs, i.e., which points are occupied by the AGVs currently, and which ones are the next desired points to be occupied in the next step.

Definition 3: The travelling information of an active AGV r_a is described as $I_a = \{n_x, n_{x+1}\}$, where n_x is the current point, and n_{x+1} is the next desired point.

Definition 4: The residual transport route Γ_a of AGV r_a represents the residual sequence of points that remain to be visited by r_a before finishing the task.

Definition 5: For an active AGV r_a, $\Omega_a$consists of an ordered sequence of resource points shared with other AGVs, which can be denoted as $\Omega_a = \{n_x : n_x \in \Gamma_a, n_x \in \Gamma_b, b \neq a\}$, where b is the number of another AGV.

When AGVs share resource points with some other AGVs, collisions and deadlocks may occur. In order to solve this problem dynamically, it is necessary to record Ω of each AGV in real time. After AGVs reach the desired points, Γ is updated. At the same time, $\Omega$of each AGV is updated according to Γ. The procedure is described in detail in Example 1.

Example 1: Shared resource points calculation as shown in Fig. 3.

In this example, the residual transport routes of r_1 and r_2 can be represented as $\Gamma_1 = \{7, 6, 5, 4, 3, 10, 11\}$ and $\Gamma_2 = \{3, 4, 5, 6, 7, 8\}$, respectively. Therefore, the shared resource points can be described as $\Omega_1 = \{8, 7, 6, 5, 4, 3\}$, $\Omega_2 = \{3, 4, 5, 6, 7, 8\}$, respectively.

Note that there might be more parking places than AGVs in the transportation system. Then, each AGV may be directed to the nearest parking place after delivering its loads to the delivery point.

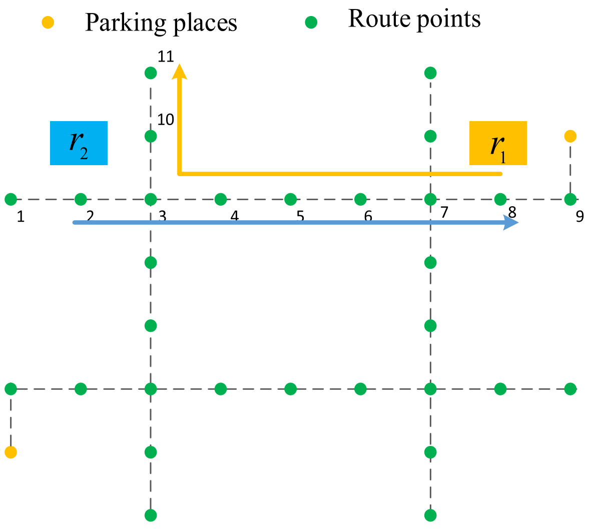

FIGURE 3. Example of shared resource calculation.

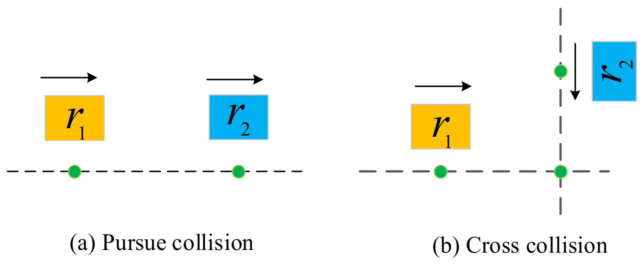

FIGURE 4. Two types of collisions.

B. DYNAMIC RESOURCE RESERVATION

Obviously, there are two types of collisions in the system, as shown in Fig. 4.

Definition 6: Collision: An AGV reserves or occupies a resource point that belongs to another AGV, and still moves to that point.

Pursue collision occurs when an AGV strike another one on the rear. Cross collision occurs when two AGVs are occupying the same resource point at the same time.

There are two types of deadlocks as shown in Fig. 5.

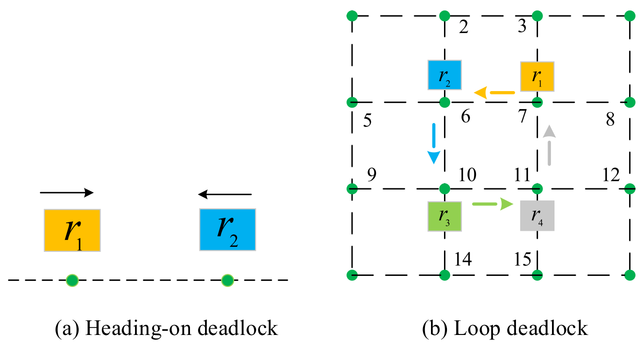

FIGURE 5. Two types of deadlocks.

Definition 7: Deadlock: The related AGVs cannot move on because each of them has to occupy the points occupied by another AGV. A deadlock occurs when two AGVs reserve the shared resource at the same time.

It is worth noting that the deadlock is actually an unresolvable collision.

Heading-on deadlock (Fig. 5(a)) occurs when two AGVs travelling at adjacent points at the same time, but the directions are opposite [13]. Loop deadlock (Fig. 5(b)) is a situation where the travelling points of the related AGVs form a closed loop.

The method proposed in this paper for collisions and deadlocks prevention is realized by changing the motion states of AGVs according to the states of the resource points. Two conditions are set to judge whether the point can be reserved by the AGV r_a. Only if the AGV satisfies all the required conditions can it go to the next desired point. Otherwise, the AGV would stop at the current resource point.

Condition 1: $n_{x+1} \notin \Omega_a$ and $s(n_{x+1}) = 0$.

It means that the next resource point of r_a does not belong to the sets of guide paths of any other AGVs.

Condition 2: $n_{x+1} \in \Omega_a$, but $\forall n_y \in \Omega_a$, $s(n_y) \neq r_b$, where r_b is another active AGV.

It means that the next resource point of r_a belongs to the guide paths of other AGVs, but all the shared resource points in Ω_a are not reserved by the other AGVs.

Summing up, when the next desired point of AGV r_a satisfies condition1 or 2, the next desired point n_{x+1} can be reserved by it. It means that the state of point n_{x+1} is $s(n_{x+1}) = r_a$. Otherwise r_a stops. After departing from the current point n_x, the state of point n_x changes from $s(n_x) = r_a$ to $s(n_x) = 0$. The AGV repeats this operation along the whole route.

Theorem 1: $\forall r_a \in R_r$: the movement of r_a based on DRR is collision-free in the system.

Proof: As mentioned above, r_a can start moving to the next point if and only if the next point can satisfy condition 1 or condition 2, which means the next point is only reserved by r_a. After movement, r_a removes the occupation of the current point. For condition 1, the next point does not belong to the shared resource points of r_a. Thus, there are obviously no collisions. For condition 2, the shared resource points can only be reserved by one AGV at a time, i.e. two AGVs cannot move to the same point at the same time. Therefore, collisions between AGVs are impossible.

Theorem 2: $\forall r_a \in R_r$: the movement of r_a based on DRR is deadlock-free in the system.

Proof: According to theorem 1, collisions between AGVs are impossible, which means the shared resource points cannot be reserved by two AGVs. Based on the definition of the deadlock, deadlocks between AGVs are impossible.

Example 2: Heading-on deadlock prevention as shown in Fig. 6.

There are three unfinished tasks that performed by r_1, r_2 and r_3. The corresponding residual routes can be represented as Γ_1 ={7,6,5,4,3,10,11}, Γ_2 ={3,4,5,6,7,8}, Γ_3 ={13,7,14,15}. The sets of shared resource points are

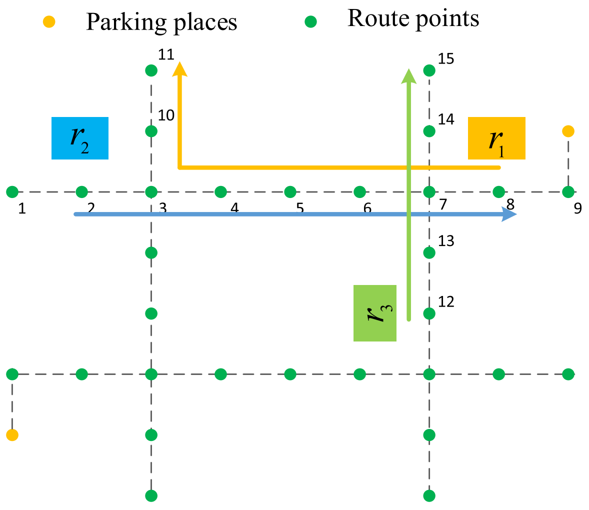

FIGURE 6. Example of heading-on deadlock prevention.

Ω_1 ={8,7,6,5,4,3}, Ω_2 ={3,4,5,6,7,8}, Ω_3 ={7}. The travelling information is I_1= {8, 7}, I_2= {2, 3},I_3= {12, 13}. Because the next point 7 of r_1 belongs to Ω_1, according to condition 2, there are no points in Ω_1 occupied by other AGVs. The point 7 can be reserved by r_1.r_1 removes the occupation of point 8 and moves on. As for r_2, the next point 3 belongs to Ω_2, but the shared resource points have been occupied by r_1, so r_2 will change its state from moving to waiting and cannot remove the occupation of point 2. The next desired point 13 of AGV r_3 is not a shared resource point, and thus r_3 can move on. Therefore, deadlocks between AGVs are impossible.

Example 3: Loop deadlock prevention as shown in Fig. 7.

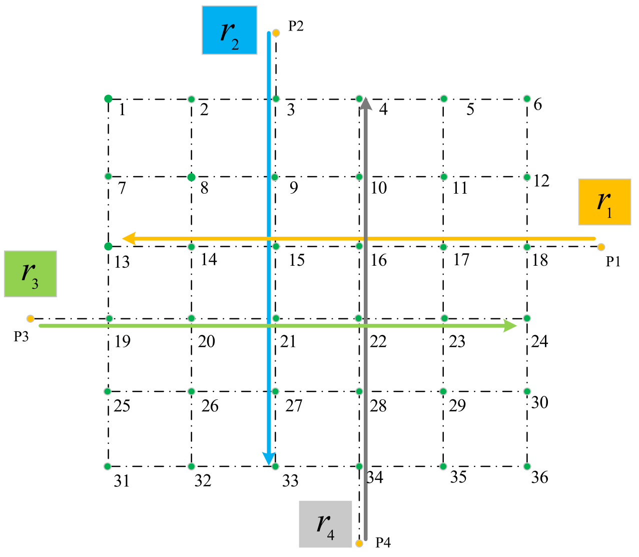

FIGURE 7. Example of loop deadlock prevention.

There are four AGVs in the system that can be represented as $R_r = \{r_1, r_2, r_3, r_4\}$. The corresponding residual routes can be represented as Γ_1 ={18,17,16,15,14,13},

Γ_2 ={3,9,15,21,27,33}, Γ_3 ={19,20,21,22,23,24} and Γ_4 ={34,28,22,16,10,4}. The sets of shared resource points are Ω_1 ={16,15}, Ω_2 ={15,21}, Ω_3 ={21,22} and Ω_4 ={22,16}. We assume that AGVs receive the task according to the AGV number. When r_1 arrives at point 17, r_2, r_3, r_4 arrive at 9, 20, 28 in turn. The travelling information is I_1= {17, 16}, I_2= {9, 15}, I_3= {20, 21} and I_4= {28, 22}. Because the next point 16 of r_1 belongs to Ω_1, according to condition 2, there are no points in Ω_1 occupied by other AGVs. The point 7 can be reserved by r_1. In the same way, r_2 and r_3 can reserve the points 15 and 16, respectively. As for r_4, the next point 22 belongs to Ω_4 and Ω_3. However the shared resource point 16 has been occupied by r_1, so r_4 will change its state from moving to waiting and cannot remove the occupation of the point 28. Therefore, deadlocks between AGVs are impossible.

C. ALGORITHM IMPLEMENTATION

The above-mentioned AGV collision and deadlock prevention method can be applied to the traffic system control. In this section, we elaborate the control algorithm implemented on both central and local controllers.

Algorithm 1 Control Policy of the Central Control Level

```
1  while system in operating statedo
2          read input data and create task list $M$
3          if $\exists m_i \in M$ then
4            if $\exists r_i \in R_a$ then
5              find the shortest route $p_i$
6              create the residual route set $\Gamma_i$
7              create the shared resource points set $\Omega_i$
8              send $p_i$ to $r_i$
9              repeat
10               if s $(n_{x+1}) = r_i$ then
11                 send an order to start movement to $r_i$
12                 read feedback information
13                 if s $(n_{x+1}) \in N^0$ then
14                   remove task $m_i$ from $M$
15                   go to line 3
16                 else
17                   update $\Gamma_i$ and $\Omega_i$
18                 end
19               else
20                 send an order to stop movement
21               end
22           end
23         else
24           go to line 4
25         end
26       else
27         go to line 3
28 end
```

The control policy of the central controller is described in detail in Algorithm 1. In this algorithm, the set M is a task list with specified order. R_r is a set of idle AGVs. The term s $(n_{x+1}) = r_i$ means that the next desired point n_{x+1} is reserved or occupied by r_i according to the conditions mentioned above. At the beginning of the simulation, the central controller generates path information and sends the information to an idle AGV (**line2-8**). When the central controller receives the travelling information that indicates AGVs arrive at a point, the sets Γ_i and Ω_i are updated (**line 10-18**). The arrival of the AGV r_i at the parking point of the task indicates that task m_i has been completed. Then, the completed task m_i is removed from the task list M. Above procedures are repeated by the central controller until all the tasks in M are completed.

Algorithm 2 Control Policy of the Local Control Level

```
1  while system in operating statedo
2      read the path $p_i$ from central controller
3      update the travelling information $I_i = \{n_x, n_{x+1}\}$
4      read the state of the next point s $(n_{x+1})$
5      if s $(n_{x+1}) = r_i$ then
6        start movement to the next point $n_{x+1}$
7        after movement to the point $n_{x+1}$
         send the travelling information
8          if $n_{x+1}$is the pick-up station then
9            stop to pick up the load
10             go to  line 3
11         if $n_{x+1}$ is the delivery station then
12             stop to unload the load
13             go to line3
14         if  $n_{x+1}$ is the parking point then
15           stop to receive a new task
16           go to line 2
17         else
18           move on
19         end
20     else
21         go to line 5
22     end
23 end
```

The algorithm implemented on the local controller is described in Algorithm 2. When vehicle r_i receives the task, the travelling information $I_i = \{n_x, n_{x+1}\}$ is generated according to the current position and the route p_i. As mentioned above, r_i can move to the next point if and only if the point is reserved by it, i.e. s $(n_{x+1}) = r_i$. Once r_i starts moving, the central controller removes the occupation of n_x. According to the property of the point, AGV can pick up, unload, stop or move on to the next point. The AGV r_i repeats this operation until the task m_i is finished (**line 5-20**). After the AGV finishes the task, its state becomes idle immediately, and then it can receive a new task.

As mentioned above, the computational time of our algorithm mainly depends on the calculation of the shared resource points. For calculating the shared resource point set of an AGV, the central controller compares its residual path with those of the other AGVs. Thus, the computation time is positively related to the number of AGVs, i.e., the computational complexity $O(N)$.

IV. SIMULATION

To the best of our knowledge, COR-based collision and deadlock prevention method is recently updated and state-of-the-art method used in scenarios similar to this paper [4]. Therefore, to demonstrate the effectiveness of the proposed method, extensive simulations have been performed and the results are compared with COR. For this purpose, we first present a performance overview of DRR with different number of AGVs. Then the impact of the number of tasks and dynamic obstacles on the efficiency of collision and deadlock prevention is shown. Lastly, we provide an insight into the mechanism of collision and deadlock avoidance via recording the procedures of AGVs.

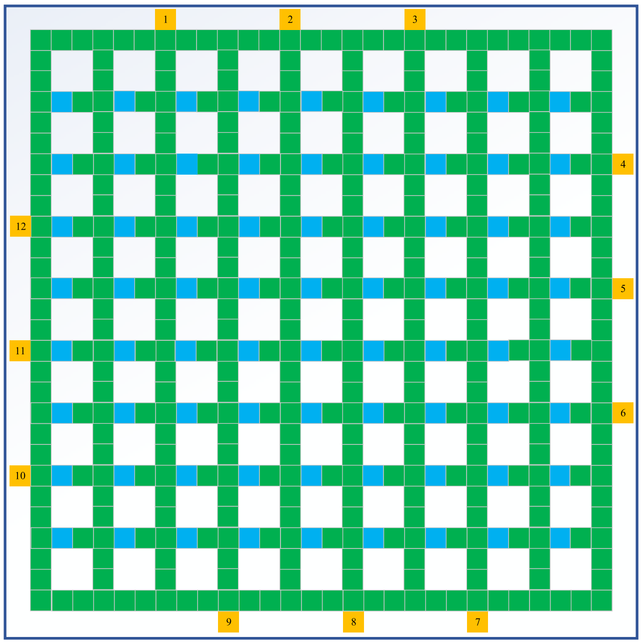

FIGURE 8. **The simulation layout.**

A. SIMULATION SETTINGS

The experiment layout with bidirectional lanes is shown in Fig. 8. The size of the square block is 1m $\times$1m and the size of the AGVs is 0.7m$\times$0.7m. In the layout, we have 72 stations which are colored in blue. Each station can be a pick-up or delivery point. There are 12 parking points marked in yellow. Guide paths are represented as green block squares. Unless otherwise specified, the velocity of the AGVs is set as 1m/s and the turning time is set as 1s.

In this paper, we focus on the collision and deadlock prevention in the transport system. In the experiment, the tasks that delivery goods and materials among workstations are generated in a task list. The central controller assigns the tasks to specified AGVs according to the order in the task list. The route of each AGV is initially generated by using the A∗ algorithm.

B. PERFORMANCE METRICS

During the simulation, we use three main metrics to evaluate the effectiveness of the collision and deadlock prevention method proposed in this paper.

- Total travelling time T: $T = T_e - T_s$, where T_s is the starting time of the first task and T_e is the time all the AGVs stop at the parking places.
- Waiting time of each AGV T_w: $T_w = \sum_{i=1}^{m} t_w^i$, where t_w^i is the time of AGV has to stop to avoid deadlocks and collisions for performing task m_i and m is the number of all the tasks generated in the transport system.
- Average waiting time T_a: the average of the waiting time of all AGVs.

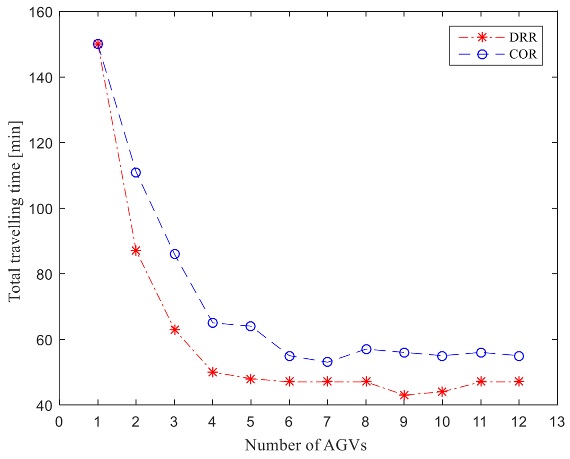

FIGURE 9. **Total travelling time with different number of AGVs.**

C. RESULTS

1) PERFORMANCE OVERVIEW

In this simulation, we compare the total travelling time of DRR and COR. The number of tasks is set as 120. The number of vehicles varies from1 to 12 and the rest of the parameters remain the same as the default values in Section IV-A. The relation between the number of AGVs and the total travelling time is shown in Fig. 9. The total travelling time of DRR is 11%∼28% less than that of COR with different number of AGVs while having almost equal travelling distance. This is because DRR reserves the resource points dynamically compared to COR in which the resource points are reserved at the beginning of the tasks. From Fig. 10, it can be seen that the average waiting time of each AGV of DRR is 25%∼50% less than that of COR.

As the number of AGVs increases gradually, the total travelling time of the two methods decreases. The best solution of DRR and COR is found with 9 and 7 vehicles, respectively. However, further increase of the number of vehicles does not bring a significant improvement, probably due to the rapidly increased congestions. Overall, with different number

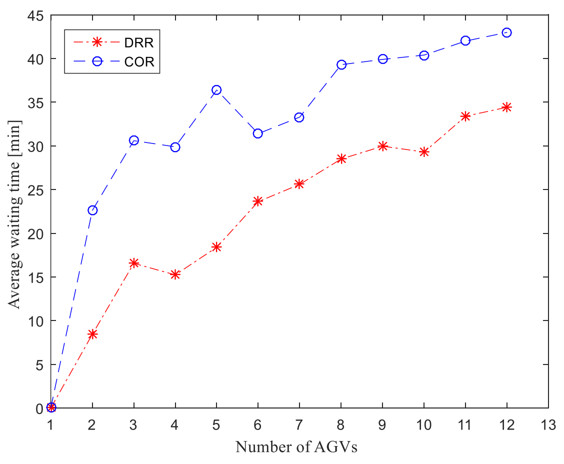

FIGURE 10. Average waiting time with different number of AGV.

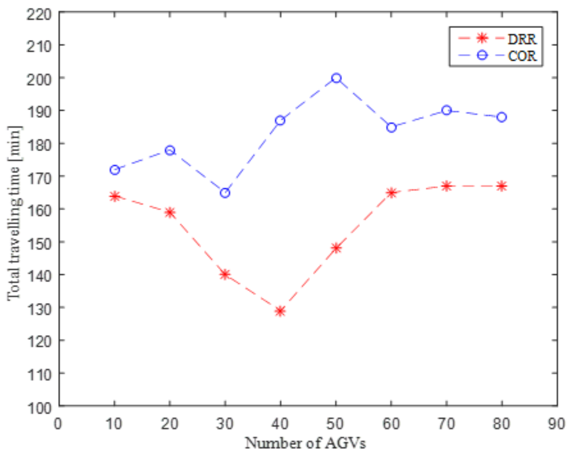

FIGURE 11. Total travelling time with larger number of AGVs.

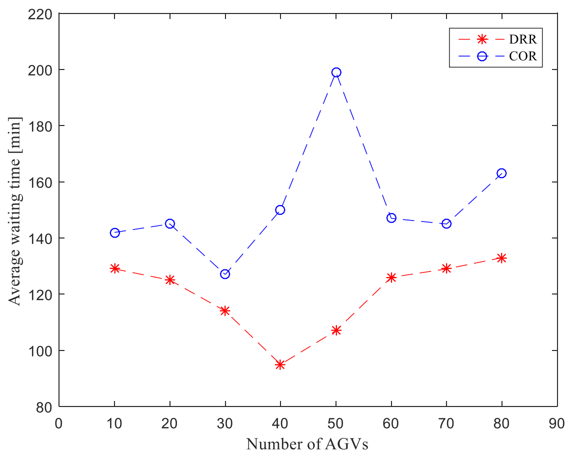

FIGURE 12. Average waiting time with larger number of AGVs.

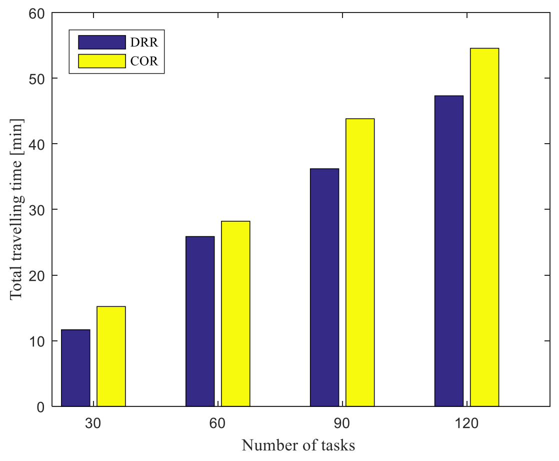

FIGURE 13. Total travelling time with different number of tasks.

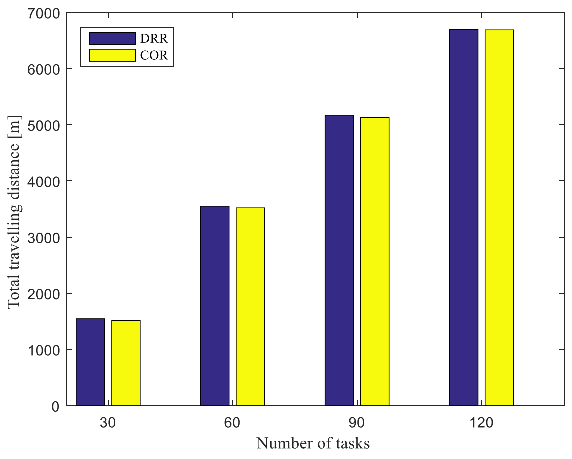

FIGURE 14. Total travelling distance with different number of tasks.

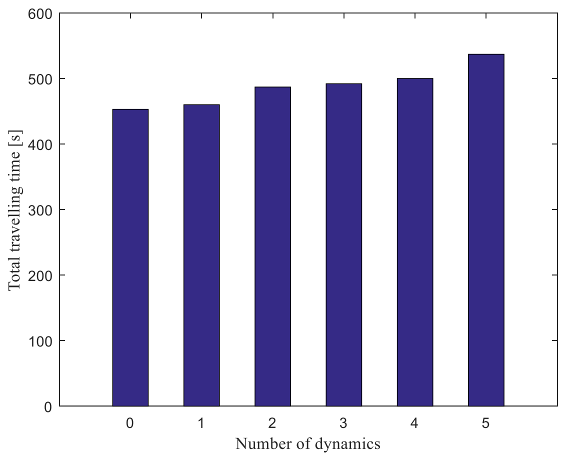

FIGURE 15. Total travelling time with different number of dynamic obstacles.

of AGVs, DRR proposed in this paper is more efficient than COR.

In practice, there are usually dozens of AGVs working in the environments. In order to verify the performance of DRR with large-scale AGV systems, we further implement DRR in a 400m^2 topological graph. The number of tasks is set as 120. The number of vehicles varies from10 to 80. The relation between the number of AGVs and the total travelling time is shown in Fig.11. The total travelling time of DRR is 4.6%∼28.8% less than that of COR with different number of AGVs. From Fig. 12, it can be seen that the average waiting time of DRR is 9.2%∼46.2% less than that of COR.

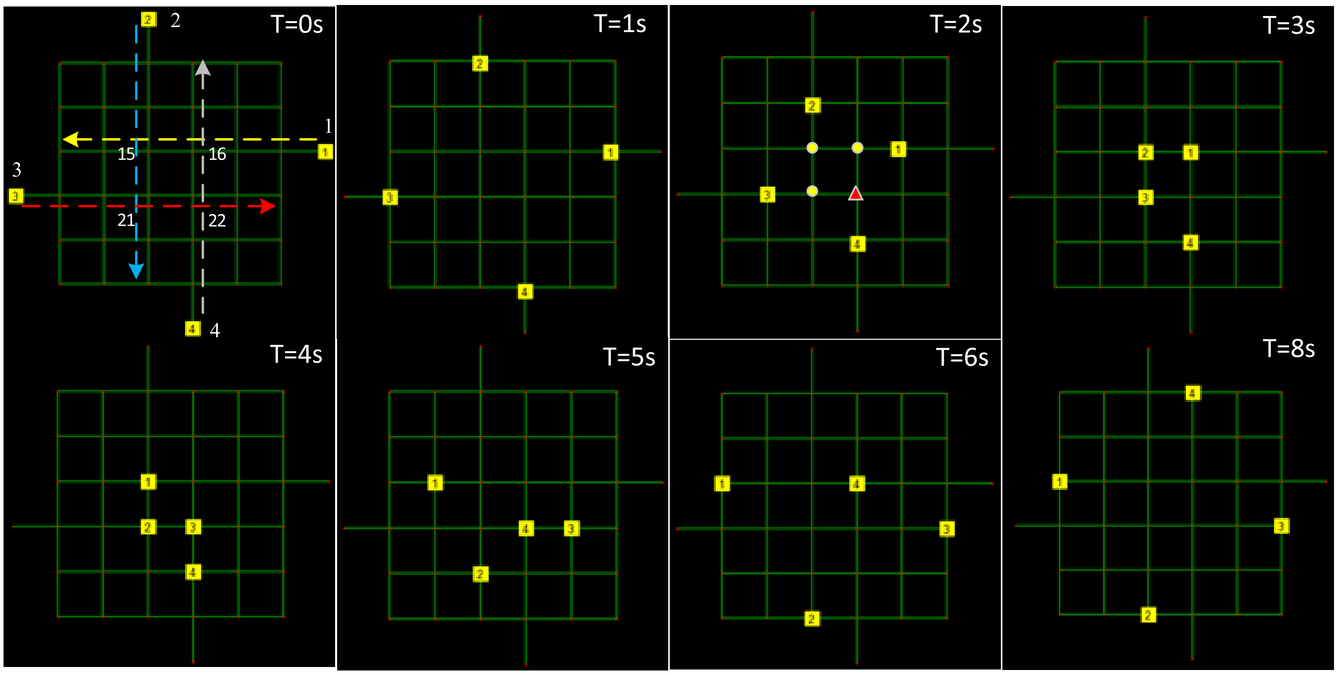

FIGURE 16. Snapshot of the deadlock prevention process.

2) IMPACT OF THE NUMBER OF TASKS

In this simulation, the impact of the number of tasks on the performance of those two methods is investigated. We set the number of vehicles as 6. From Fig. 13, we can see that the total travelling time with different number of tasks based on DRR is 22%∼40% less than that of COR. However, the travelling distance of both methods is almost equal, as shown in Fig. 14. As expected, from Fig. 13, we can also see that the total travelling time and the total travelling distance increase for both methods as the number of tasks increases. For different number of tasks, the DRR shows better performance with less total travelling time.

3) IMPACT OF DYNAMIC OBSTACLES

In practice, external obstacles (e.g. people, manual forklifts) often appear in the working environment of AGVs. In order to verify the performance of DRR with dynamic obstacles, we implement DRR in a 5*5 topological graph with 12 AGVs. The velocity of each AGV is set as 0.5m/s.

In the simulation, we add dynamic obstacles with a residence time of 10s to random position. From Fig. 15, it can be seen that there are no collisions and deadlocks in such a dynamic and congested situation (one AGV per 2 square meters). And the total travelling time increases as the number of the obstacles increase.

4) PROCESS OF LOOP DEADLOCK PREVENTION

To have an insight into the detailed mechanism of collision and deadlock prevention method, we implement DRR in a 5∗5 topological graph with 4 AGVs. As shown in Fig.16, we assume that four AGVs are assigned tasks according to the ID of each AGV. The routes of AGVs are represented as dotted lines in different colors. Obviously, 15,16,21,22 are the shared resource points. At $\mathbf{t} = \mathbf{2s}$, the points 16, 15 and 21 are reserved by $\mathbf{r_1}$, $\mathbf{r_2}$ and $\mathbf{r_3}$, respectively. However, point 22 is the shared point and 16, 21 have been reserved by $\mathbf{r_1}$ and $\mathbf{r_3}$, respectively. So $\mathbf{r_4}$ cannot reserve the desired point 22, and thus $\mathbf{r_4}$ has to stop, as shown in t=3s. At the same time, $\mathbf{r_3}$ can first reserve the point 22 and release the occupation of the point 21. $\mathbf{r_2}$ can reserve the point 21 and $\mathbf{r_1}$ can reserve 15, as shown in t=4s. Once the occupations of the shared points are removed, $\mathbf{r_4}$ can start movement, as shown in t=5s. At t=8s, all the tasks are completed and no deadlocks occur.

V. CONCLUSION

In this paper, a dynamic resource point reservation based collision and deadlock prevention method DRR is proposed, which can ensure collision-free AGVs travelling while achieving high time efficiency. DRR changes AGV motion states to prevent collisions and deadlocks dynamically, depending on whether the points have been reserved. Simulation results show that the total travelling time of DRR is 11% ∼ 28% less than that of COR with different number of AGVs while having almost equal travelling distance. The total travelling time and waiting time of each AGV of DRR are less than those of COR with different number of tasks. The future work should focus on optimizing the task dispatch sequence that can reduce the AGV task execution time while ensuring collision-and-deadlock-free travelling.

REFERENCES

[1] R. Yu, H. Zhao, S. Zhen, K. Huang, X. Chen, H. Sun, and K. Zhang, “A novel trajectory tracking control of AGV based on Udwadia-Kalaba approach,” *IEEE/CAA J. Automatica Sinica*, early access, Nov. 9, 2017, doi: 10.1109/JAS.2016.7510139.

[2] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, ''Efficient computation offloading for Internet of vehicles in edge computing-assisted 5G networks,'' *J. Supercomput.*, vol. 76, no. 4, pp. 2518–2547, Apr. 2020.

[3] S. Wan and S. Goudos, ''Faster R-CNN for multi-class fruit detection using a robotic vision system,'' *Comput. Netw.*, vol. 168, Feb. 2020, Art. no. 107036, doi: 10.1016/j.comnet.2019.107036.

[4] W. Malopolski, ''A sustainable and conflict-free operation of AGVs in a square topology,'' *Comput. Ind. Eng.*, vol. 126, pp. 472–481, Dec. 2018.

[5] T. W. Min, L. Zhe, H. Khee Yin, G. C. Hiang, and L. K. Yong, ''A rules and communication based multiple robots transportation system,'' in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat. (CIRA)*, Monterey, CA, USA, Nov. 1999, pp. 180–186.

[6] 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 2014, pp. 6144–6149.

[7] M. B. Z. Remba, A. Obuchowicz, Z. A. Banaszak, and K. J. Jed Rzejek, ''A max-algebra approach to the robust distributed control of repetitive AGV systems,'' *Int. J. Prod. Res.*, vol. 35, no. 10, pp. 2667–2688, Nov. 2010.

[8] I. Draganjac, D. Miklic, Z. Kovacic, G. Vasiljevic, and S. Bogdan, ''Decentralized control of multi-AGV systems in autonomous warehousing applications,'' *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1433–1447, Oct. 2016.

[9] L. Cancemi, A. Fagiolini, and L. Pallottino, ''Distributed multi-level motion planning for autonomous vehicles in large scale industrial environments,'' in *Proc. IEEE 18th Conf. Emerg. Technol. Factory Autom. (ETFA)*, Cagliari, Italy, Sep. 2013, pp. 1–8.

[10] E. Roszkowska, ''Decentralized motion-coordination policy for cooperative mobile robots,'' in *Proc. 9th Int. Workshop Discrete Event Syst.*, Goteborg, Sweden, May 2008, pp. 364–369.

[11] M. P. Fanti, A. M. Mangini, G. Pedroncelli, and W. Ukovich, ''A decentralized control strategy for the coordination of AGV systems,'' *Control Eng. Pract.*, vol. 70, pp. 86–97, Jan. 2018.

[12] D. H. Pérez and H. M. Barberá, ''Decentralized coordination of automated guided vehicles,'' in *Proc. 7th Int. Joint Conf. Auto. Agents Multiagent Syst. (AAMAS)*, Estoril, Portugal, 2008, pp. 1195–1198.

[13] H. Zheng, R. R. Negenborn, and G. Lodewijks, ''Fast ADMM for distributed model predictive control of cooperative waterborne AGVs,'' *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1406–1413, Jul. 2017.

[14] 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.

[15] N. Smolic-Rocak, S. Bogdan, Z. Kovacic, and T. Petrovic, ''Time windows based dynamic routing in multi-AGV systems,'' *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 1, pp. 151–155, Jan. 2010.

[16] Z. Zhang, Q. Guo, J. Chen, and P. Yuan, ''Collision-free route planning for multiple AGVs in an automated warehouse based on collision classification,'' *IEEE Access*, vol. 6, pp. 26022–26035, Mar. 2018.

[17] Z. Fan, C. Gu, X. Yin, C. Liu, and H. Huang, ''Time window based path planning of multi-AGVs in logistics center,'' in *Proc. 10th Int. Symp. Comput. Intell. Design (ISCID)*, Hangzhou, China, Dec. 2017, pp. 161–166.

[18] J. Xin, C. Meng, F. Schulte, J. Peng, Y. Liu, and R. Negenborn, ''A time-space network model for collision-free routing of planar motions in a multi-robot station,'' *IEEE Trans. Ind. Informat.*, early access, Jan. 22, 2020, doi: 10.1109/TII.2020.2968099.

[19] Q. Li, J. T. Udding, and A. Pogromsky, ''Zone-control-based traffic control of automated guided vehicles,'' *Lect. Notes Control Inf. Sci.*, vol. 456, pp. 53–60, Sep. 2015.

[20] M.-S. Yeh and W.-C. Yeh, ''Deadlock prediction and avoidance for zone-control AGVS,'' *Int. J. Prod. Res.*, vol. 36, no. 10, pp. 2879–2889, Oct. 1998.

[21] X. Yan, C. Zhang, and M. Qi, ''Multi-AGVs collision-avoidance and deadlock-control for item-to-human automated warehouse,'' in *Proc. Int. Conf. Ind. Eng., Manage. Sci. Appl. (ICIMSA)*, Seoul, Jun. 2017.

[22] Y.-C. Ho, ''A dynamic-zone strategy for vehicle-collision prevention and load balancing in an AGV system with a single-loop guide path,'' *Comput. Ind.*, vol. 42, nos. 2–3, pp. 159–176, Jun. 2000.

[23] Y.-C. Ho and T.-W. Liao, ''Zone design and control for vehicle collision prevention and load balancing in a zone control AGV system,'' *Comput. Ind. Eng.*, vol. 56, no. 1, pp. 417–432, Feb. 2009.

[24] X. Li, C. Zhang, W. Yang, and M. Qi, ''Multi-AGVs conflict-free routing and dynamic dispatching strategies for automated warehouses,'' in *Proc. Int. Conf. Mobile Wireless Technol.* Singapore: Springer, 2018, pp. 277–286.

[25] M. Qi, X. Li, X. Yan, and C. Zhang, ''On the evaluation of AGVS-based warehouse operation performance,'' *Simul. Model. Pract. Theory*, vol. 87, pp. 379–394, Sep. 2018.

[26] 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.

[27] 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.

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

[29] N. Wu and W. Zeng, ''Deadlock avoidance in an automated guidance vehicle system using a coloured Petri net model,'' *Int. J. Prod. Res.*, vol. 40, no. 1, pp. 223–238, Jan. 2002.

[30] T. Nishi and Y. Tanaka, ''Petri net decomposition approach for dispatching and conflict-free routing of bidirectional automated guided vehicle systems,'' *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 42, no. 5, pp. 1230–1243, Sep. 2012.

[31] M. Uzam, ''An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions,'' *Int. J. Adv. Manuf. Technol.*, vol. 19, no. 3, pp. 192–208, Feb. 2002.

[32] N. Wu, M. Zhou, and G. Hu, ''One-step look-ahead maximally permissive deadlock control of AMS by using Petri nets,'' *ACM Trans. Embedded Comput. Syst.*, vol. 12, no. 1, pp. 1–23, Jan. 2013.

[33] J. Luo, H. Ni, and M. Zhou, ''Control program design for automated guided vehicle systems via Petri nets,'' *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 1, pp. 44–55, Jan. 2015.

YUNLONG ZHAO received the M.S. degree from the China University of Ming & Technology, Beijing, China, in 2016. He is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. His research interests include multirobot system, and coordination and path planning.

XIAOPING LIU received the Ph.D. degree from Tianjin University, Tianjin, China, in 1994. He is currently a Professor with the Beijing University of Posts and Telecommunications. His research interests include logistics automation equipment and system parameter identification.

GANG WANG received the Ph.D. degree from Beihang University, Beijing, China, in 2016. He is currently a Lecturer with the Beijing University of Posts and Telecommunications. His research interests include logistics automation equipment and robot precision analysis and synthesis.

SHAOBO WU received the Ph.D. degree in mechatronics engineering from Beihang University, Beijing, China, in 2018. He is currently a Postdoctoral Researcher with the Beijing University of Posts and Telecommunications. His research interest includes industrial wireless.

SONG HAN received the B.S. degree from the University of Science and Technology of China, in 2014. He is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. His research interests include robotics and machine vision.