IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 4 November 2023, accepted 17 November 2023, date of publication 21 November 2023, date of current version 29 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3335843

RESEARCH ARTICLE

Geometric Zone-Control Algorithm for Collision and Deadlock Avoidance in AGV System

KWANWOO LEE AND SANGCHUL PARK
Department of Industrial Engineering, Ajou University, Suwon 16499, Republic of Korea
Corresponding author: Sangchul Park (scpark@ajou.ac.kr)

This work was supported by a National Research Foundation grant [NRF-2020R1A2C1004544] funded by the Korean Government (Ministry of Science and ICT); and an Institute for Information and Communications Technology Promotion grant [IITP-2021000292] funded by the Korean Government (Ministry of Science and ICT).

ABSTRACT Automated guided vehicle (AGV) system control presents several challenges, among which deadlock situations are particularly problematic, as they can significantly reduce the overall performance of the AGV system. Existing studies are based on the assumption that there is sufficient space between nodes and links in AGV guidepath topology. This study proposes a novel zone-control algorithm for AGV systems designed to prevent collisions and deadlocks. The proposed algorithm involves a zone-partitioning technique that considers both AGV geometry and guidepath topology. This method identifies all collision-prone areas and divides the AGV guidepath into zones. By effectively employing these zones, the zone-control algorithm successfully addresses and resolves deadlock problems in AGV systems. The effectiveness of the proposed algorithm was evaluated against state-of-the-art methods using irregular layouts. Experimental results demonstrated that the proposed method effectively handled delivery tasks, resulting in a 58–85% improved performance, thereby verifying its efficacy. The proposed algorithm offers a practical and effective solution for AGV systems with irregular guidepath topologies at real manufacturing sites.

INDEX TERMS Automated guided vehicle, automated material handling system, deadlock avoidance, zone-control.

I. INTRODUCTION

Automated guided vehicles (AGVs) are mobile robots that are widely used in various industrial settings, such as manufacturing sites, warehouses, and harbor docks [1], [2]. AGV systems offer greater flexibility than other automated material handling systems, such as overhead hoist transport (OHT) and conveyor systems. However, AGV system control presents several challenges, including path planning, dispatching, and deadlock resolution. Among these issues, deadlock situations, in which AGVs become stuck because of conflicts with other AGVs or obstacles [2], are particularly problematic, as they can significantly reduce the overall performance of the AGV system. Consequently, numerous research efforts have been devoted to developing effective deadlock-resolution methods.

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

Zone-control prevents collisions and deadlocks in AGV systems by limiting the number of AGVs that can enter specific areas, as shown in Fig. 1 [2]. This approach divides the AGV guidepath into zones, with each zone representing a logical segmentation that can be occupied by only one AGV at a time. Owing to its simplicity, this approach has been adopted by several studies to develop effective deadlock avoidance algorithms [2]. For instance, Fanti et al. [3] proposed a zone-control method that describes an AGV system using a colored timed Petri net model. Moorthy et al. [4] introduced a method for detecting cyclic deadlocks in zone-control. Yoo et al. [5] proposed the application of a graph-theoretical deadlock detection algorithm for AGV systems. However, these methods do not provide guidance for dividing the AGV guidepath into zones.

There is an inherent trade-off between deadlock-avoidance performance and space utilization. High space utilization is critical because it is directly linked to delivery time.

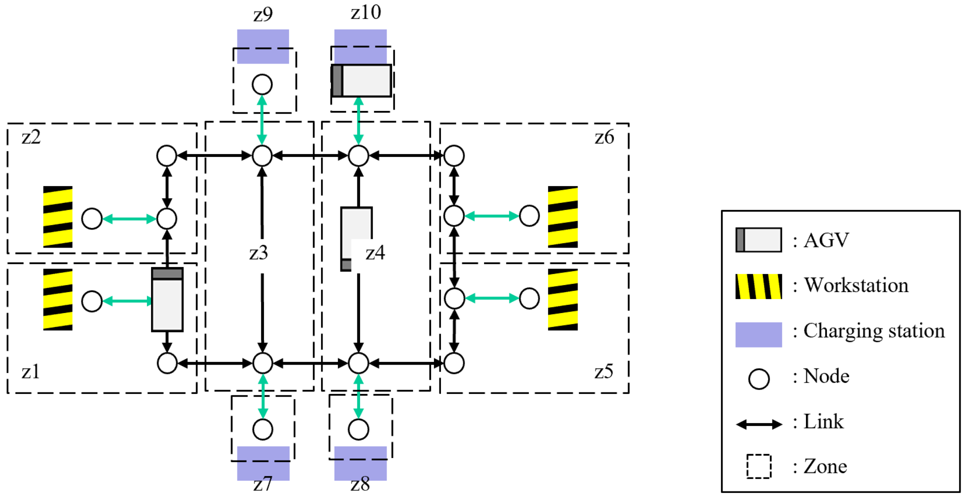


FIGURE 1. **Zone-control approach to avoid deadlocks.**

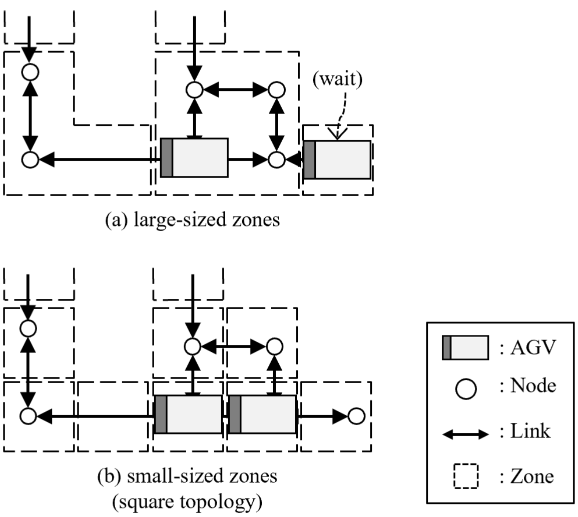


FIGURE 2. **Impact of zone-partitioning on AGV guidepath utilization.**

As illustrated in Fig. 2(a), partitioning zones of relatively with larger sizes reduces the likelihood of a deadlock occurring because only one AGV can occupy a single zone at any given time. Conversely, as demonstrated in Fig. 2(b), small-sized zone sizes increase space utilization with a higher likelihood of deadlock occurrence. To formulate small-sized zones that preclude collisions and deadlocks, it is crucial to precisely identify potential problem areas.

Numerous studies have investigated zone-control and zone-partitioning techniques. In [6] and [7], zone-control approaches are presented using zone-partitioning methods that create zones where junctions occur in the AGV guidepath. In [8] and [9], dynamic zone-control algorithms in which zones are reformulated during AGV operation to balance the load among AGVs are proposed. Małopolski [10] suggested a square topology that divides AGV layouts into squares, with each square being larger than one AGV. This method is easily applicable to grid-type layouts commonly found in distribution warehouses but is more challenging when applied to manufacturing site layouts with irregular configurations.

Using this topology, Małopolski [10] introduced a method termed chain of reservation (COR) that requests all zones along an AGVs' path. Zhao et al. [1] proposed a method termed dynamic resource reservation (DRR) in a square topology that focuses on managing shared resource zones between AGVs. Zając and Małopolski [11] presented a novel method termed structural online control policy (SOCP) within this topology. This approach classifies square zones as deadlock-risk or deadlock-free zones. This method effectively controls AGVs using two policies with these zones. Zhao et al. [12] proposed a spare-zone-based hierarchical motion-coordination algorithm for grid layouts. In this method, AGVs reserve spare zones for potential conflicts with other vehicles. By utilizing these spare zones during conflict situations, the algorithm effectively resolves deadlocks without rerouting grid-type layouts.

To effectively prevent deadlocks from occurring in practical applications, it is imperative to concurrently consider both AGV geometry data and guidepath topology. For instance, in scenarios with two unconnected links as illustrated in Fig 3(a), there is no risk of collision and deadlock when small-sized vehicles are in operation. In contrast, as depicted in Fig 3(b), the operation of large-sized vehicles can potentially lead to collisions and deadlocks. At manufacturing sites, AGV guidepaths are often designed after finalizing the placement of major manufacturing equipment, which can result in unexpected issues in the system. To effectively address these, a comprehensive consideration of both AGV geometry and guidepath topology is essential. To the best of our knowledge, no existing methodology fully combines AGV guidepath topology and AGV geometry to tackle the

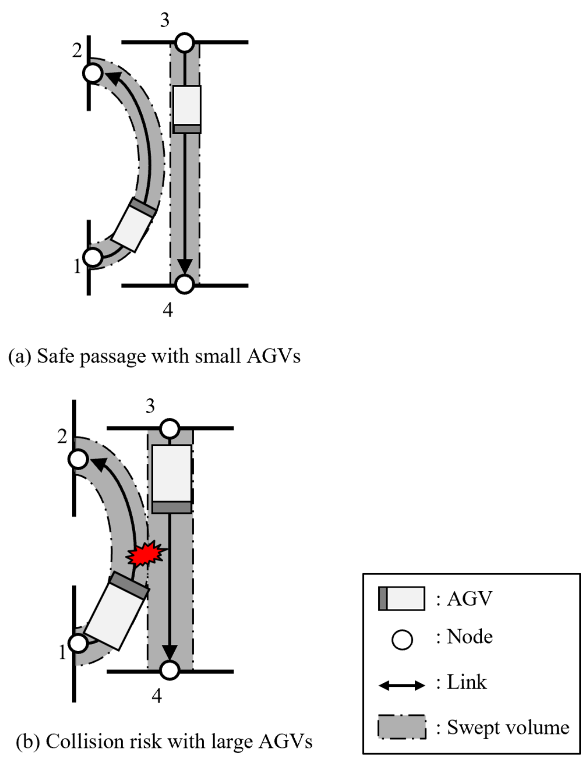


FIGURE 3. Impact of AGV geometry size on collision risk.

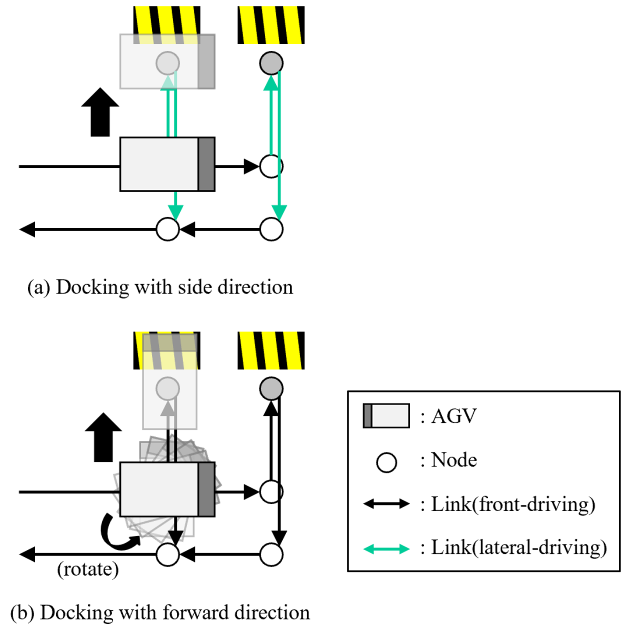


FIGURE 4. AGV docking situations.

associated practical challenges. This is because, with regard to AGV guidepath topology, previous research results are based on the assumption that there is sufficient space between nodes and links.

This paper presents a method for deadlock avoidance that considers both AGV geometry and guidepath topology data. This method consists of two main components: a zone-partitioning technique and a deadlock avoidance algorithm. The zone-partitioning technique divides the AGV guidepath into zones while taking into consideration the AGV geometry and guidepath topology. This ensures high space utilization by identifying problematic areas in the guidepaths beforehand. Further, the deadlock algorithm predicts and mitigates potential AGV deadlocks using these zones. The method's efficacy was confirmed in comparison to other modern approaches.

The remainder of this paper is organized as follows. Section II presents the problem statement, emphasizes the challenges presented by irregular layouts in real-world AGV guidepaths, and describes the zone-partitioning technique considering both AGV geometry and guidepath topology. Section III introduces the deadlock avoidance algorithm, which utilizes the generated zones to manage the AGV movement and prevent deadlocks in the system. Section IV provides a sample layout and simulation results and compares the proposed algorithm with other methods. Finally, the conclusions are presented in Section V.

II. ZONE-PARTITIONING TECHNIQUE

A. PROBLEM STATEMENT

This study represents the AGV guidepath network as a directed graph, $G = <N, L>$. N denotes the set of all nodes in the graph and can be represented as $N = \{n_i : 1 \leq i \leq D\}$. Each node $n_i \in N$ represents a topological point that can accommodate one AGV, and AGVs rotate to align their driving direction on these nodes. L denotes the set of links that connect two nodes and can be represented as $L = \{l_jk : n_j, n_k \in N, j \neq k\}$. AGVs cannot rotate on the links between nodes. In an actual plant, nodes can be marked with QR landmarks, and links can be installed as wires. A node connected to any equipment or a charging station is termed a docking node.

In this guidepath layout, multiple AGV vehicles ($A = \{a_r : 1 \leq r \leq R\}$) operate simultaneously to transport goods between workstations. While docking with a workstation, an AGV often moves in the lateral direction (Fig. 4(a)). However, if the AGV must travel along a link, it must align with the direction of the link (Fig. 4(b)). The driving direction of AGVs is closely related to collision-prone areas, and a common strategy for reducing collision-prone areas is to move in the side direction. To account for this, we marked the front and side driving links in black and green, respectively, considering the driving directions of all the links in the guidepath network.

Qi et al. [13] identified two types of collision situations in AGV systems: the pursuit of collision, which occurs when two or more vehicles move in succession, and cross collision, which occurs when two or more vehicles enter a cross-section. These types of collisions can be easily prevented in actual plants by operating sensors mounted on AGVs and managing intersections. However, as previously mentioned, owing to the flexibility of AGVs, AGV guidepaths are often designed after the layout of manufacturing sites is already fixed. Consequently, irregular collision-prone areas can occur in real-world manufacturing sites.

Herein, irregular layout patterns that result in collision-prone areas are classified into three patterns: node–node (NN), node–link (NL), and link–link (LL) patterns, as shown in Fig. 5. The NN pattern occurs when two unconnected nodes are closer than the diagonal length of an AGV, whereas the NL pattern occurs when a node is located close to an unconnected link and is often observed in environments with acute-angled link connections. The LL pattern occurs in the form of intersections, overlaps, and parallel links. Notably, the crossing and intersections of links are considered distinct scenarios. In the case of crossing, an AGV can move in multiple directions through a node at the center point. However, for an intersection, a vehicle can only move in the direction in which it travels, as shown in Fig. 5(c)–(ii). Therefore, crossing links does not create an LL pattern because the node at the center separates these links.

B. DETECTING COLLISION-PRONE AREAS IN IRREGULAR LAYOUTS

This study proposes the concept of swept volume to detect collision-prone areas in the AGV guidepath. A swept volume is the physically collision-prone area when an AGV uses a component in the AGV guidepath. Each node and link in the AGV guidepath has a swept volume. According to the definition of a node, AGVs cannot move on a node but rotate to change their driving direction. Conversely, they cannot rotate on a link. Therefore, two types of swept volumes, driving swept and rotational swept, are proposed, as shown in Fig. 6.

- *Definition 1 (Driving Swept Volume):* For every link $l_{ij} \in L$, a driving swept volume DSV_{ij} is defined as the collision area that occurs when an AGV moves on l_{ij}.

For a node $n_k \in N$ in the AGV guidepath, a link l_{ik} is termed an inlink of n_k if the end node of the link is n_k, and a link l_{kj} is termed an outlink of n_k if the starting node of the link is n_k. The set of inlinks and outlinks are represented as $L_k^{in} = \{l_{ij} : n_i, n_j \in N \text{ and } j = k\}$ and $L_k^{out} = \{l_{ij} : n_i, n_j \in N \text{ and } i = k\}$, respectively. Rotations occur for every inlink–outlink pair at n_k.

- *Definition 2 (Rotational Swept Volume):* For a node n_k, $|L_k^{in}| \times |L_k^{out}|$ rotations exist. Every rotation at n_k ($R_k(i,j)$, where $l_{ik} \in L_k^{in}$ and $l_{kj} \in L_k^{out}$), generates a swept volume. The rotational swept volume RSV_k is defined as the union of all swept volumes that occur during rotations at n_k.

After determining the swept volumes for every node and link in the AGV guidepath, it is possible to identify all potential collision and deadlock areas by detecting the irregular patterns in the guidepath. For instance, in Fig. 7(a), although nodes n_1 and n_2 are not directly connected, their swept volumes overlap, which classifies this as an NN pattern. Similarly, in Fig. 7(b), while link $l_1 2$ and node n_3 are not directly connected, their swept volumes intersect, which indicates an NL pattern. Moreover, in Fig. 7(c), if the swept volumes of links without a direct connection overlap, they are labelled as following an LL pattern.

C. VIRTUAL NODE GENERATION

This study proposes a virtual node concept to improve the utilization of an AGV guidepath. To emphasize the utility of virtual nodes, we assume that two vehicles want to enter links l_{12} and l_{34} simultaneously in the layout in Fig. 8(a). The straightforward approach does not allow one vehicle to enter the link until the other vehicle exits the other link (Fig. 8(b)), which has low utilization of the guidepath. However, this limitation can be addressed by managing collision-prone areas on the links using virtual nodes. Using this concept, we can accurately identify when the first vehicle passes through the collision-prone area, as shown in Fig. 8(c).

- *Definition 3 (Virtual Node):* A *virtual node* τ_{ij}^d is a logical point on a link l_{ij}. It is generated at a specific point where the overlap of the swept volumes is detected. Herein, d represents the distance from the starting node (n_i) of link l_{ij}.

Notably, virtual nodes are not physical nodes marked by the designer on the actual site. On the contrary, they are generated only at the intersection of swept volumes. To understand this concept, an example of virtual node generation is shown in Fig. 9, where the intersection of RSV_3 and DSV_{12} generates a virtual node τ_{12}^d on the link l_{12}. The virtual node τ_{12}^d has a rotational swept volume, denoted RSV_{12}^d, which is the same as the DSV_{12} at a position d away from the start node of l_{12}.

After generating virtual nodes for all possible collision areas on the links in an AGV guidepath, the set of all the virtual nodes is denoted T. To simplify the process, a new node set U is defined by combining nodes from N and T, i.e., $U = N \cup T$. This allows the detection of every collision by considering the RSVs of the nodes in U only. This study defines two adjacency levels between nodes: neighbor and alias.

- *Definition 4 (Neighbor/Alias):* For all $n_i, n_j \in U$, where $i \neq j$, if $RSV_i \cap RSV_j \neq \emptyset$, then n_i and n_j are in a neighboring relationship. If $POS(n_i) \in RSV_j$ or $POS(n_j) \in RSV_i$, where $POS(n)$ is the position of node n, then n_i and n_j have an aliased relationship. As a sufficient condition, nodes in an aliased relationship are always in a neighbor relationship.

When a collision can occur during the rotation of an AGV at two nodes, the nodes are said to be in a neighboring relationship. However, when two nodes are in an aliased relationship, AGVs cannot be located on those nodes simultaneously.

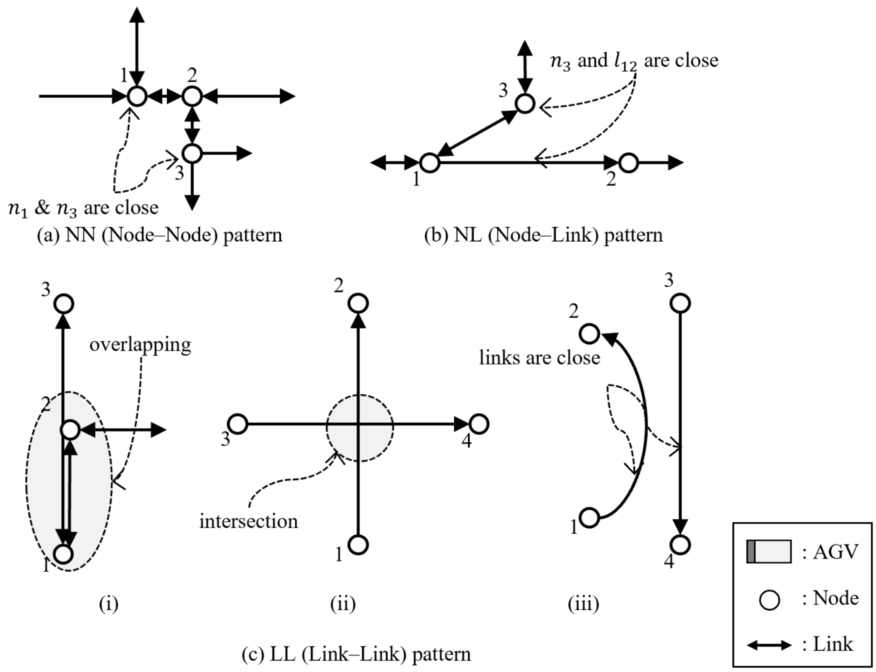


FIGURE 5. Irregular layout patterns.

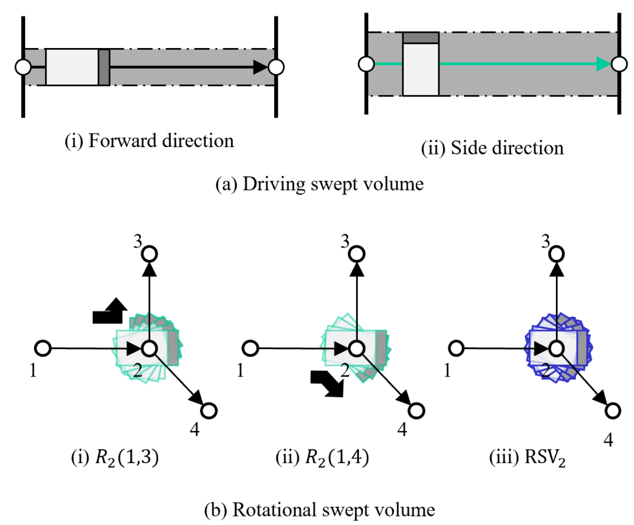


FIGURE 6. Swept volume.

D. GEOMETRIC ZONE-PARTITIONING TECHNIQUE

Adjacency levels were established for all the nodes in an AGV guidepath (U). After grouping the nodes according to aliased relationships, zones were created for each alias node group. Fig. 10 shows the zone generation results for the irregular patterns in Fig. 5. Rectangles with dotted lines indicate the zones. Zones were generated for original and virtual nodes ($\tau_{ij}^d \in T$), as shown in Fig. 10(c).

Several studies have defined a zone as a non-overlapping region for one vehicle. However, this study introduces the concept of neighboring zones as zones containing nodes that are in a neighboring relationship with a node in a zone. The example in Fig. 11 shows that space utilization increases when zones overlap. Consider two vehicles moving toward nodes n_1 and n_3; if only one vehicle can occupy a zone at a time, as shown in Fig. 11(a), only one vehicle can enter the zone. Conversely, as shown in Fig. 11(b), both vehicles can enter their respective zones without conflict because these zones do not overlap. This shows that overlapping zones (neighboring zones) improve space utilization and AGV system efficiency.

After generating all the zones in an AGV guidepath, the next step is to assign attributes to each zone. First, zones containing docking nodes should be identified and designated as docking zones, which serve as the start or end points for the AGV paths. Notably, neighboring zones for docking zones should be avoided because these zones cannot be used until AGVs in the docking zones are moved. Additionally, bidirectional sections exist in an AGV guidepath, as shown in Fig. 12(a), because AGVs can move bidirectionally along links. These bidirectional sections can cause head-on deadlocks between vehicles, as shown in Fig. 12(b). To prevent deadlocks, zones in bidirectional sections are merged to create bottleneck zones, as shown in Fig. 12(c). Notably, consecutive bottleneck zones cannot exist because they merge if they are contiguous.

Fig. 13(a) shows a sample AGV guidepath layout. Fig. 13(b) shows the zones generated using the zone-partitioning technique proposed in this study for the layout shown in Fig. 13(a). The docking and bottleneck

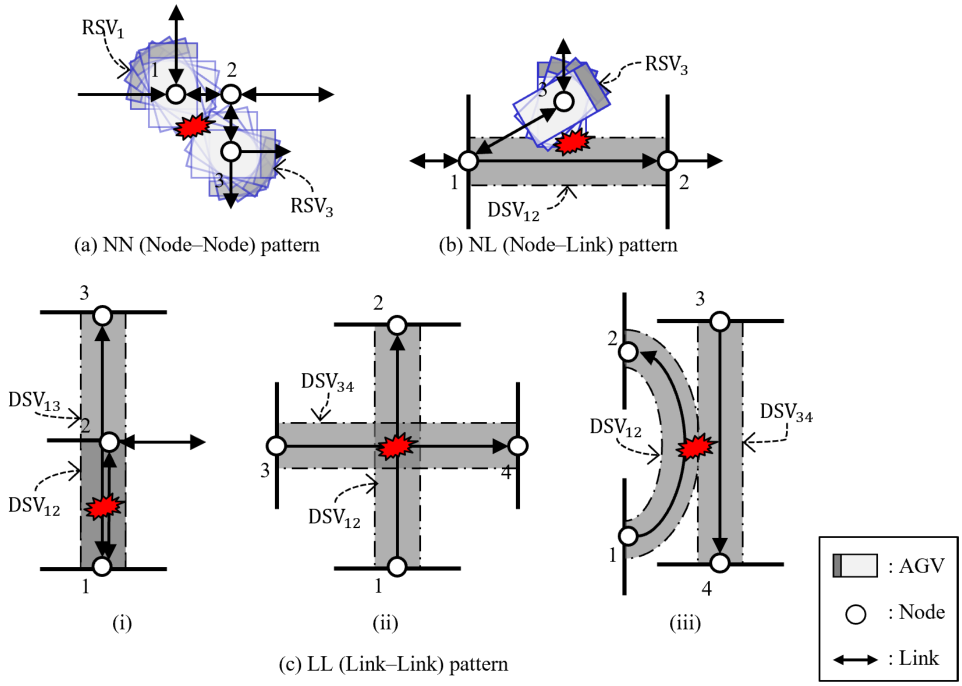


FIGURE 7. Collision detection results using swept volume for the patterns in Fig. 5.

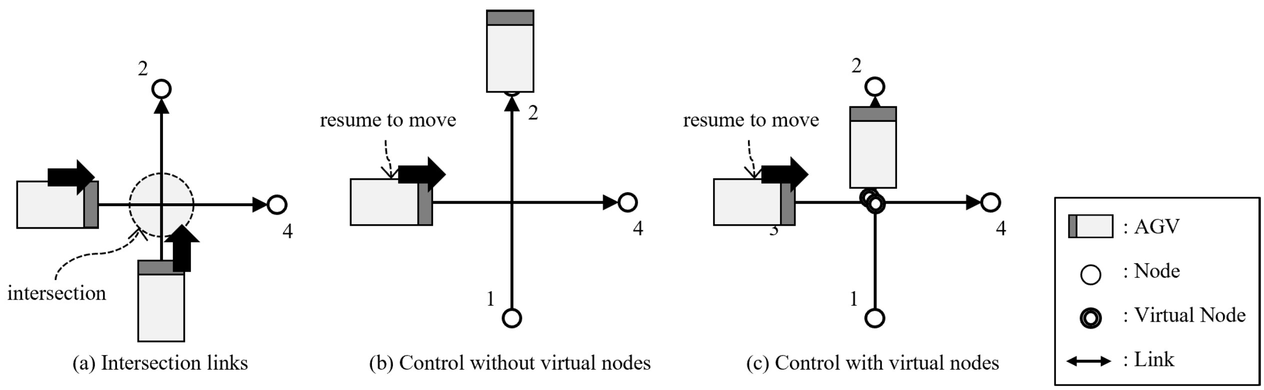


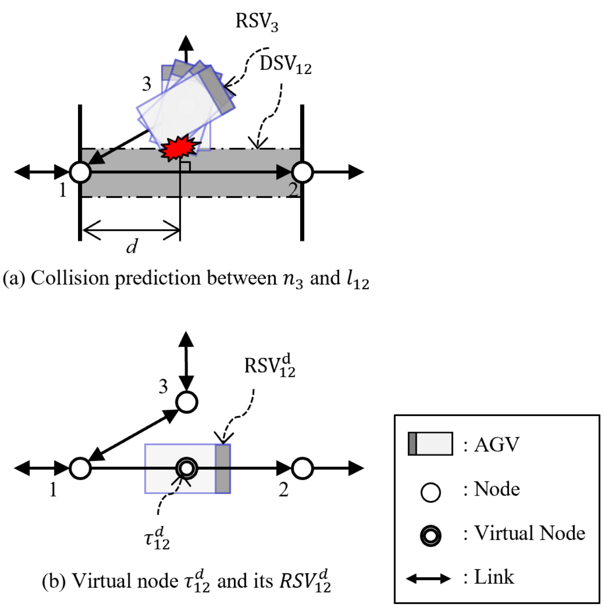
FIGURE 8. Example of virtual node utilization.

zones are highlighted in yellow and green, respectively. The red connections between zones indicate the neighboring relationship between the zones. The overall zone-partitioning technique is as follows:

1) Read the input data and create $G = <N, E>$ as an AGV guidepath.
2) Calculate all swept volumes (RSV, DSV) for nodes and links in G.
3) Check intersections between swept volumes.
4) Generate virtual nodes at intersection points on the links.
5) Establish all aliased and neighboring relationships between nodes (U), and construct node sets by grouping nodes into aliased relationships.
6) Generate zones with node sets.
7) Identify the connections between the zones based on the connection information of the nodes belonging to each zone.
8) Identify docking zones.
9) Identify bidirectional sections and generate bottleneck zones.
10) Establish neighboring relationships between zones based on the neighboring relationships of nodes belonging to each zone.

III. DEADLOCK AVOIDANCE ALGORITHM

In this section, we present the deadlock avoidance algorithm using the zones generated by the proposed zone-partitioning method. When a transport task is assigned to an AGV a_r, the zone path of AGV a_r can be denoted $ZP = \{z_0, z_1, z_2, \ldots, z_k\}$,



(a) Collision prediction between n_3 and l_{12}

(b) Virtual node τ_{12}^d and its RSV_{12}^d

FIGURE 9. Procedure for generating virtual nodes.

where z_0 and z_k are the docking zones. We propose two functions, OC and EN, for collision avoidance, where $OC: Z \rightarrow A \cup \{0\}$ verifies the occupation status of zones and $EN: Z \rightarrow A \cup \{0\}$ verifies whether any AGV is entering a zone. If $OC(z) = a_r$, zone z is occupied by AGV a_r, whereas $OC(z) = 0$ denotes that zone z is not occupied. Similarly, if $EN(z) = a_r$, AGV a_r enters zone z, whereas $EN(z) = 0$ indicates that no AGV enters. These functions can be employed to verify whether AGV a_r can enter zone z without causing collisions. The collision-checking algorithm is outlined in Algorithm 1.

Algorithm 1 Collision Check: *CC*

Input: AGV a_r and checking nodes z
Output: Blocking AGV a'

if $OC(z) \neq 0$ **then**
  return $OC(z)$
else if $EN(z) \neq 0$ **then**
  return $EN(z)$
else
  for all $\acute{z} \in z.Neighbors$ **do**
    if $OC(\acute{z}) \neq 0$ and $OC(\acute{z}) \neq a_r$ **then**
      return $OC(\acute{z})$
    else if $EN(\acute{z}) \neq 0$ **then**
      return $EN(\acute{z})$
    end if
  end for
end if
return 0 /* No Blocking AGV */

Theorem 1: For AGV a_r, if $CC(a_r, z)$=0, zone z is collision-free.

Algorithm 2 Deadlock Avoidance: *DA*

Input: AGV a_r and its current step on the zone paths i
Output: Next destination zone z for AGV a_r

if $CC(a_r, z_{i+1}) \neq 0$ **then**
  Add $a_r \rightarrow CC(a_r, z_{i+1})$ to the block list
  return *Null* /* To avoid collisions, it does not move */
end if

if $i = 0$ **then**
  if $PR(z_1) \neq 0$ **then**
    Add $a_r \rightarrow PR(z_1)$ to the block list
  else if $PR(Z_2) \neq 0$ **then**
    Add $a_r \rightarrow PR(z_2)$ to the block list
  else
    $PR(z_1) \leftarrow a_r$
    $PR(z_2) \leftarrow a_r$ /* Request first two zones */
  end if
else
  $OC(z_{i-1}) \leftarrow 0$ /* Leave the previous zone */
  $EN(z_i) \leftarrow 0$ /* Arrive at the current zone */
  $PR(z_i) \leftarrow 0$ /* Cancel request to the current zone */
  $OC(z_i) \leftarrow 0$ /* Occupy the current zone */
  if $PR(z_{i+2}) = 0$ **then**
    $PR(z_{i+2}) \leftarrow a_r$
  else
    Add $a_r \rightarrow PR(z_{i+2})$ to the block list
  end if
end if

if $PR(a_r, z_{i+2}) = a_r$ **then**
  $EN(z_{i+1}) \leftarrow a_r$
  return z_{i+1} /* Now AGV a_r starts to move to z_{i+1} */
else
  return *Null* /* No available path to move */
end if

Proof: Only two types of collisions occur in an AGV guidepath: collisions in the same zone and collisions not in the same zone. First, collisions in the same zone can be avoided by verifying the occupation and entry status of the zone, that is, $OC(z) = 0$ and $EN(z) = 0$. Second, collisions that are not in the same zone can be avoided by checking the occupation and entry status of the neighboring zones. If the neighboring zones are free ($OC(\acute{z}) = 0$ and $EN(\acute{z}) = 0$,where $\acute{z} \in z.Neighbors$), AGV a_r is free from collisions with other AGVs that are not in the same zone z. Therefore, if $CC(a_r, z) = 0$, both the same zone and its neighboring zones are collision-free for AGV a_r, and thus, zone z is collision-free for AGV a_r.

The proposed zone-control algorithm adopts the quasi-two-step request policy (QTRP) for zone reservations, as suggested by Kim and Kim [15], in which AGVs request two zones (z_1,z_2) at the initial step ($i = 0$) and only z_{i+2} at subsequent steps ($i > 0$). This policy ensures

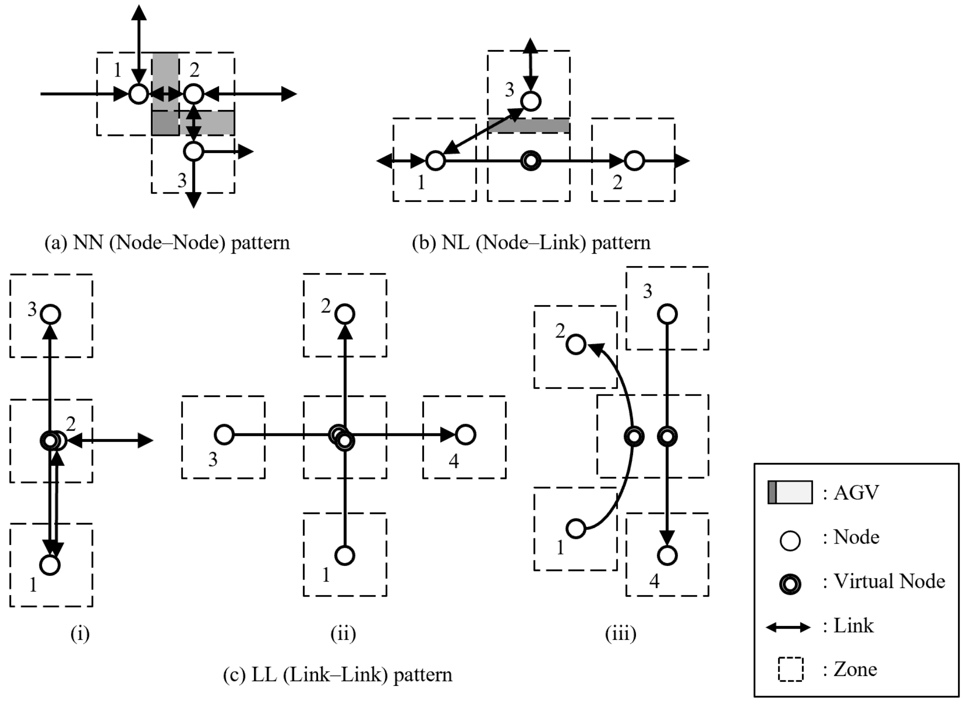


FIGURE 10. **Zone-partitioning results for the patterns in Fig. 5.**

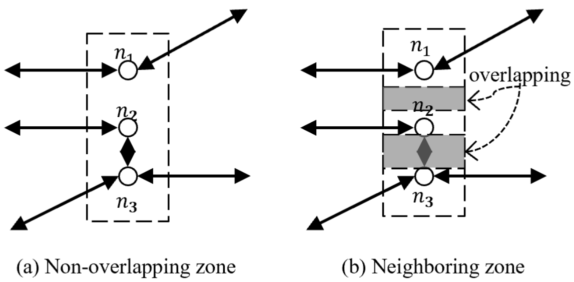


FIGURE 11. **Example of neighboring zone.**

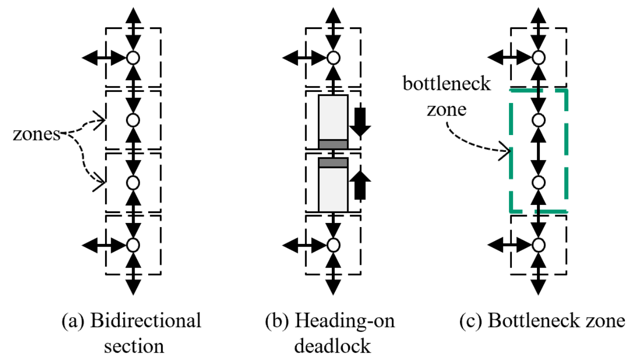


FIGURE 12. **Examples of bidirectional and bottleneck zones.**

high resource utilization without permanent blocking. The function $PR: Z \rightarrow A \cup \{0\}$ represents the request status of zones, where $PR(z) = a_r$ denotes that zone z is requested by AGV a_r, and $PR(z) = 0$ indicates that no AGV requests the zone. The deadlock avoidance algorithm is outlined in Algorithm 2.

In Algorithm 2, AGV a_r does not proceed if the next zone exhibits a collision risk (Lines 1–4). Lines 6–13 demonstrate the initial request step, and Lines 16–24 show the residual request step. AGV a_r starts to move to the next zone (z_{i+1}) if it successfully requests the next two-step zone (z_{i+2}) (Lines 27–29). If AGV a_r fails to request zone z_{i+2}, it records the AGV causing a blockage in the block list. If a cycle ($a_1 \rightarrow a_2, a_2 \rightarrow a_3, \ldots, a_r \rightarrow a_1$) is detected in the block list, one of the AGVs in the cycle changes its path to resolve the cyclic block situation. This strategy prevents deadlocks and ensures smooth AGV system operation by dynamically adjusting AGV routes when a potential deadlock scenario is identified, thus contributing to efficient material handling and transportation processes.

Theorem 2: Any cyclic block situations detected from *DA* can always be resolved.

Proof: Consider a cyclic block situation detected by *DA*. Because AGVs request two zones based on the QTRP, at least one requested zone for AGVs is present in the cyclic block. Let the current zone where an AGV is located be denoted z_i, where i is the index of the zone in the zone path. According to the definition of bottleneck zone, consecutive bottleneck zones do not exist. Therefore, if z_i is a bottleneck zone, the next requested zone (z_{i+1}) must be a non-bottleneck zone, and vice versa. Because non-bottleneck zones have more than two connected zones, candidate zone paths exist to resolve cyclic block situations.

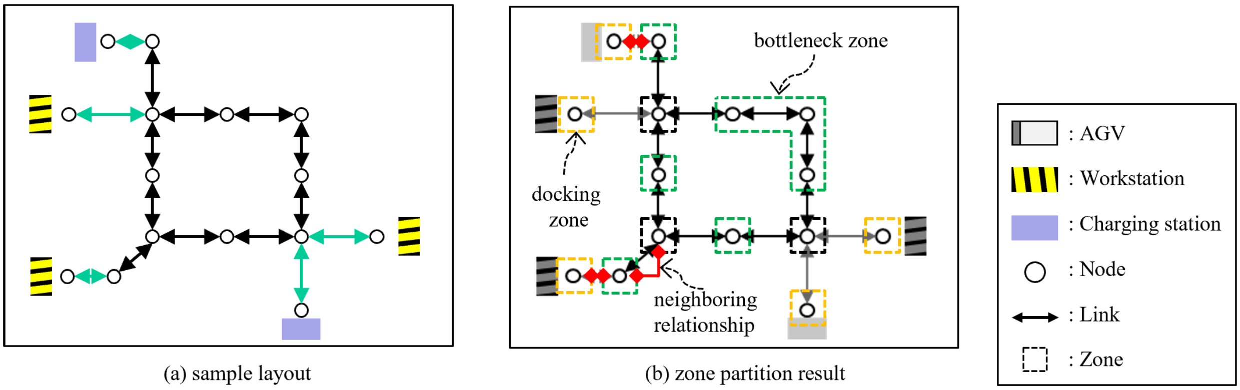


FIGURE 13. Example of zone-partitioning for sample layout.

TABLE 1. AGV parameters.

Parameter	Value
Length[m]	0.9
Width [m]	0.7
Speed [m/s]	1
Accleration[m/s^2]	2
Body turn speed [deg/s]	18
Wheel turn speed [deg/s]	12.5
Part capacity [pcs]	1

Fig. 14 shows a scenario with two AGVs and two transport tasks, where circles indicate request situations (blue: a_1, red: a_2). Initially, each vehicle requests two zones in the zone path (Fig. 14(a)), and after arriving at the next zone, it requests the next two-step zone (z_3) and continues moving (Fig. 14(b)). However, the next two-step zone z_4 of AGV a_2 is occupied by AGV a_1, and AGV a_2 records the blocking situation ($a_2 \rightarrow a_1$) in the block list (Fig. 14(c)). Eventually, AGV a_1 also records its blocking situation ($a_1 \rightarrow a_2$) in the block list (Fig. 14(d)). A cycle between two vehicles is detected in the block list, and AGV a_1 is rerouted. This example demonstrates the dynamic adjustment of AGV routes by the proposed method to resolve cyclic blocking situations, effectively preventing deadlocks and ensuring smooth AGV system operation. By continuously monitoring and adapting to potential deadlock scenarios, AGV systems can maintain efficient material handling and transportation processes.

IV. RESULTS AND DISCUSSION

A. EXPERIMENTAL SETUP

This section presents the simulation results from experiments on a map emulating the AGV layout found in actual manufacturing sites. We opted to compare our proposed method with two state-of-the-art methods (COR [10] and SOCP [11]), both applicable in maps that have irregular patterns. This study is primarily concerned with collision and deadlock avoidance in AGV systems, implying that other AGV system logics, such as dispatching and routing, remain consistent throughout the simulation. The central controller assigns tasks to AGVs based on the nearest-vehicle-first (NVF) logic, and each AGV's route is generated using the Dijkstra algorithm. The specifications of the AGV are listed in Table 1. For simulation environments, we used the commercial software '*Pinokio*', developed by Carlo, Republic of Korea.

B. SAMPLE LAYOUT AND ITS ZONE-PARTITIONING RESULT

The layout of the AGV guidepath used in this study is illustrated in Fig. 15. There are 28 workstations, each serving as both the starting and ending point for each transport task. Ten AGVs are responsible for transferring tasks. Within the layout, there is an unconnected layout marked by blue links, creating intersections within the central area of the map. Our proposed zone-partitioning technique automatically identifies areas prone to collisions. Subsequently, the AGV guidepath is divided into zones, as shown in Fig. 16. This layout incorporates the irregular patterns discussed in Section III. Many NN patterns are detected in the bay sections on the right side of the guidepath, which are susceptible to collisions when AGVs are in rotation. Secondly, we observed a NL pattern on the left side of the guidepath, arising from a node and a node being too close without a connection. Finally, three LL patterns are identified at the intersections within the central portion of the layout. Docking and bottleneck zones are denoted in yellow and green, respectively, and red connections signify adjacent zones with potential collision risks.

C. EXPERIMENT RESULTS

The effectiveness of the proposed method was assessed through simulations comparing its performance with that of COR and SOCP using 10 task scenarios and the default parameters listed in Table 1. Each task scenario had a daily workload ranging from 100 to 1000 transport requests.

Delivery time refers to the duration from when a vehicle is allocated to a transport task to when that task is completed. It is crucial to optimize the delivery time to enhance the efficiency of transport tasks within the AGV system. Delivery time comprises two key components: pre-drive time (the time

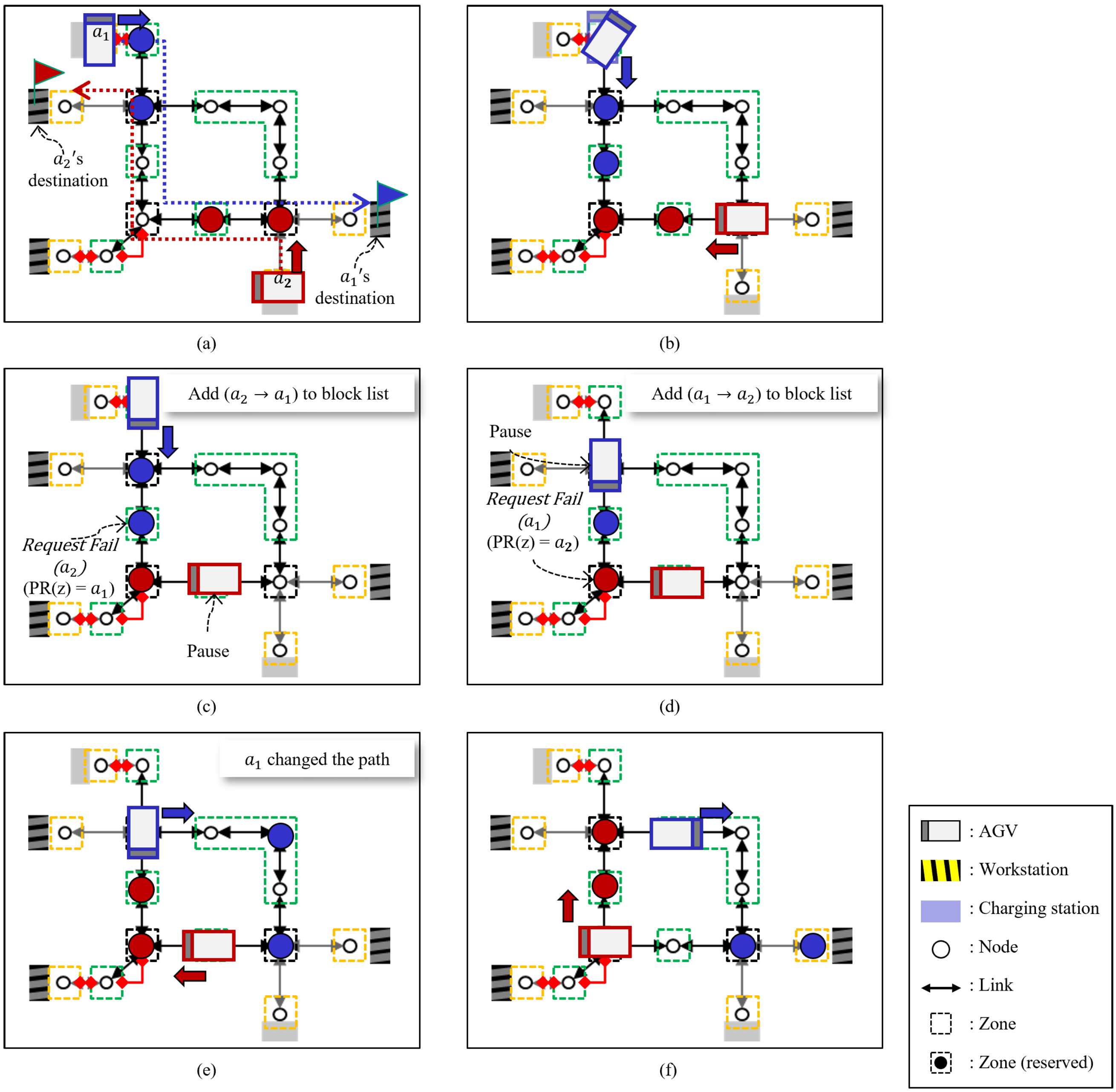


FIGURE 14. Example of deadlock avoidance algorithm.

required to arrive at the pick-up location) and main-drive time (the time taken to travel from the pick-up location to the destination).

As depicted in Fig. 17, the delivery time for both COR and SOCP gradually increased with an increase in the number of transport requests. Specifically, the delivery time for COR increased sharply at the 400-transport request level, whereas for SOCP, this increase occurred at the 600-transport request level. In contrast, our proposed method consistently managed up to 1,000 transport requests with an average delivery time of 3.84 min.

The COR method employs a strategy where it reserves all zones on both the pre-drive and main-drive routes at the departure point to prevent deadlocks until the return trip. However, as illustrated in Fig. 18, this operational approach contributed to increasing the pre-drive time, particularly in scenarios where the workload exceeded a certain threshold. In contrast, the SOCP method follows a strategy where it requests all deadlock-risk zones on each route (pre-drive or main-drive) just before beginning a route. As observed in Fig. 18, 19, this method increased both the pre-drive and main-drive times.

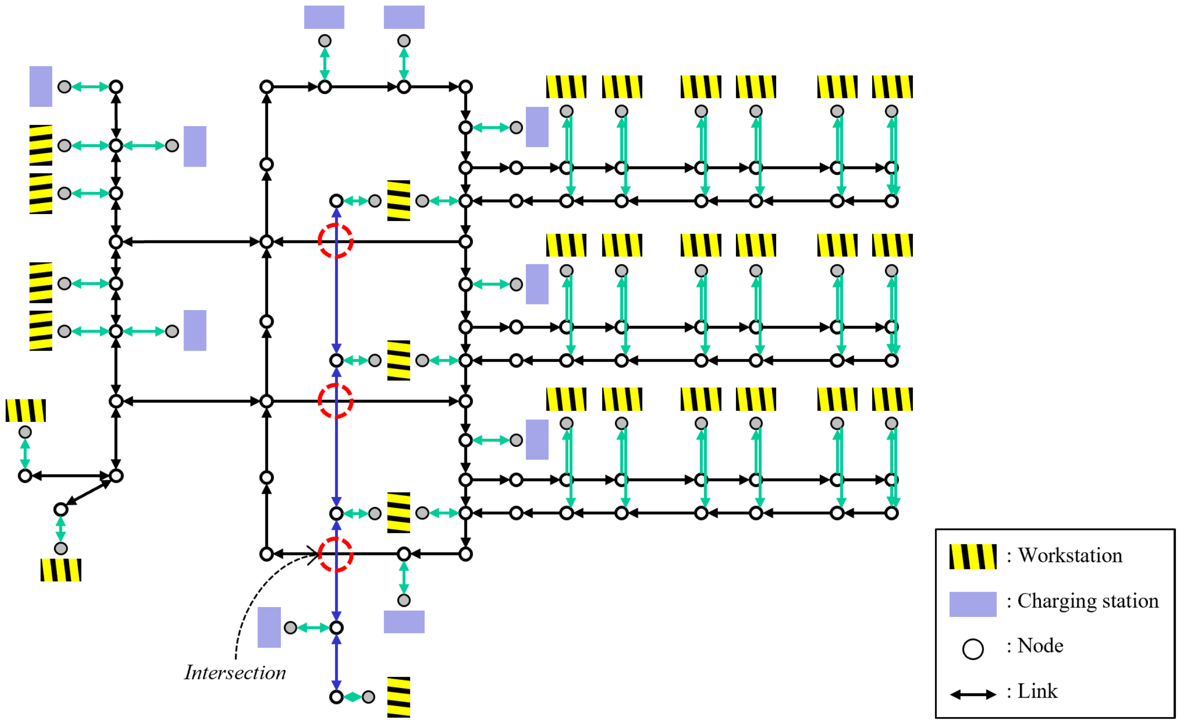


FIGURE 15. Sample layout with irregular patterns.

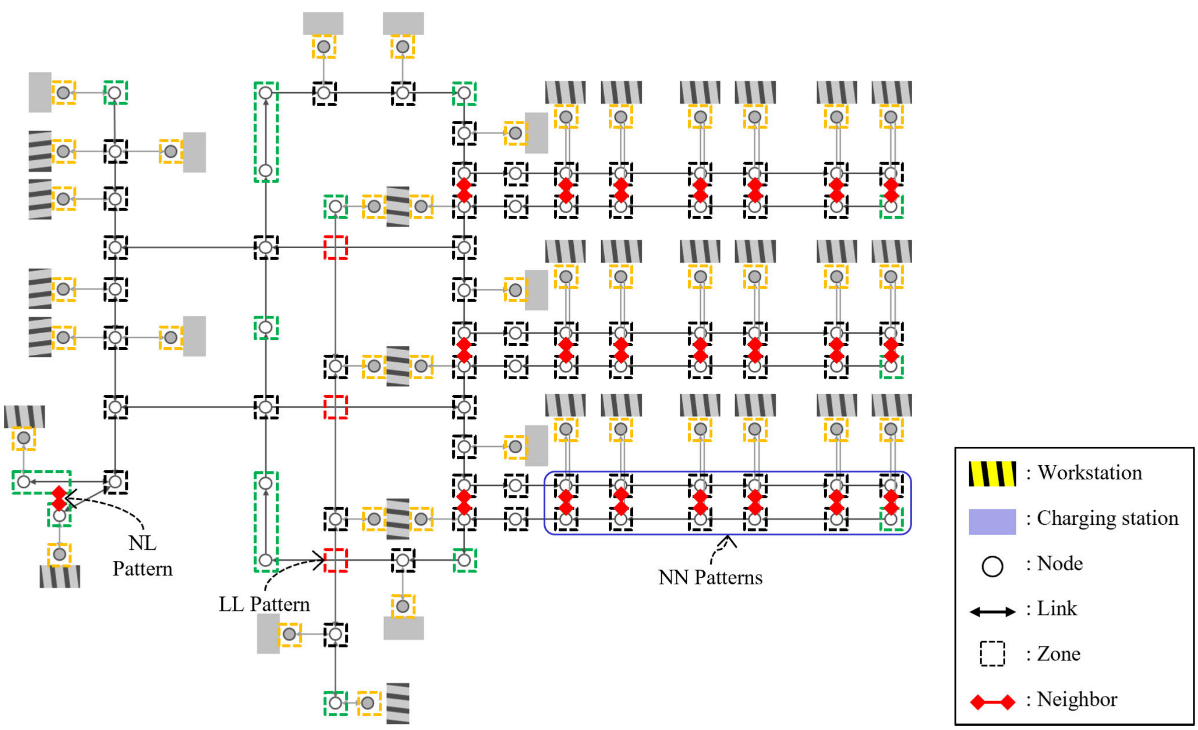


FIGURE 16. Zone-partitioning results of the layout in Fig. 15.

On the other hand, the proposed method reserved only two zones, thus geometrically ensuring deadlock prevention before departure. Consequently, the average delivery time for executing transport tasks was reduced by 6-58% compared to COR and 9-85% compared to SOCP, as shown in Fig. 17.

The increase in delivery time can be attributed to the fact that vehicular congestion increases with an increase in the

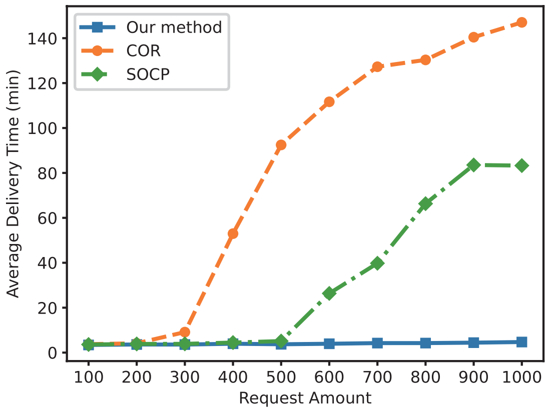


FIGURE 17. Average delivery time.

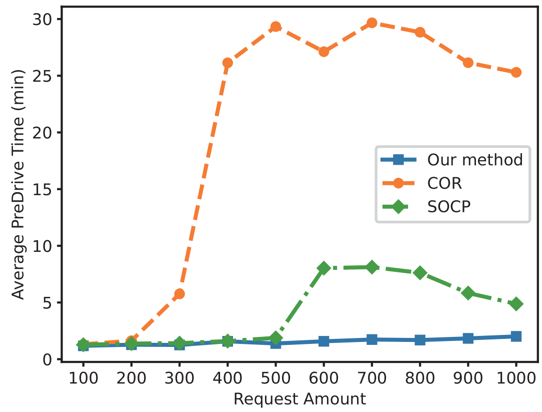


FIGURE 18. Average pre-drive time.

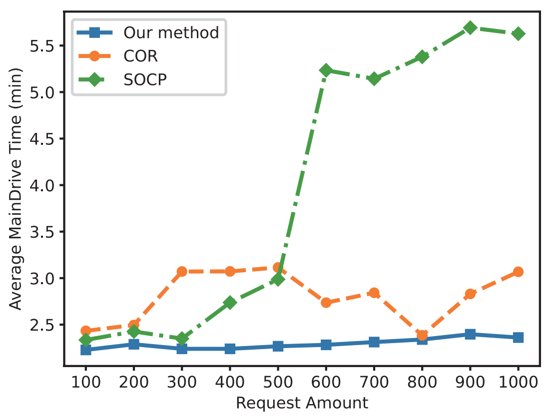


FIGURE 19. Average main-drive time.

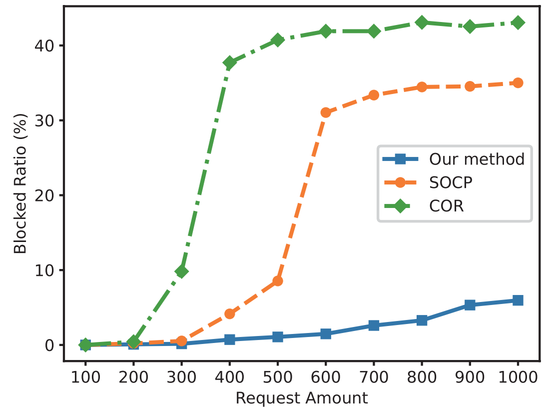


FIGURE 20. Blocked ratio.

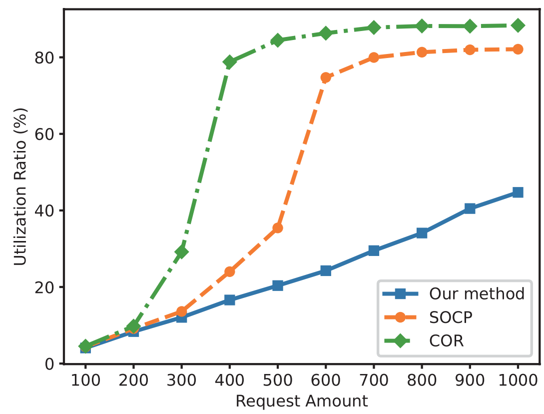


FIGURE 21. Utilization ratio.

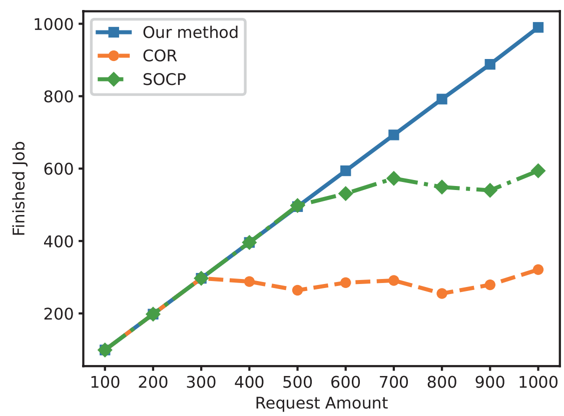


FIGURE 22. Throughput.

number of transport requests. As seen in Fig. 20, for other logics (i.e., SOCP and COR), the proportion of vehicles that were stopped due to deadlock resolution increased to over 30% with an increase in the workload demand, resulting in blocked time. In contrast, the approach proposed by this study effectively maintained a blocked time ratio of less than 10% across all experiments. Consequently, from the perspective of AGV utilization, our proposed method exhibited an average of 49.27% lower operational rate in congested transport scenarios (when there were more than 600 transport requests per day), as shown in Fig. 21, implying that fewer vehicles could effectively handle these tasks.

Our proposed method successfully processed all transport requests in every transport task scenario, on the map with irregular layout patterns, as indicated in Fig. 22. The simulation results demonstrate that the proposed zone-control algorithm manages AGVs more effectively than the other methods. However, compared with alternative methodologies, the proposed approach faces limitations, as it requires additional information, such as the AGV travel directions on all links and AGV geometric data.

V. CONCLUSION

This study proposes a zone-control algorithm that could detect and avoid collisions and deadlocks in irregular AGV guidepath. We found that the AGV geometry information and topology of the AGV guidepath must be considered simultaneously for perfect collision and deadlock avoidance. Using the swept volume concept, we identified all collision-prone areas and partitioned the AGV guidepath into zones using this information. Then, we presented a novel zone-control algorithm for efficiently resolving collisions and deadlocks by leveraging zones with high space utilization. Simulation results showed that the proposed method completes the same number of tasks 58–85% faster than competing approaches, proving its efficiency. This zone-control approach could be applied to any type of AGV guidepath layout containing irregular patterns to effectively prevent collisions and deadlocks.

In future studies, we will improve AGV system performance by integrating a path-planning algorithm into our proposed approach. In our zone-control method, all the AGVs in transit have their zone paths. The overlapping zone paths for nearby zones that AGVs will visit in the near future may indicate the potential for future traffic congestion. By utilizing this information to significantly adjust AGV routes, we expect to reduce congestion and enhance the overall system performance.

REFERENCES

[1] Y. Zhao, X. Liu, G. Wang, S. Wu, and S. Han, "Dynamic resource reservation based collision and deadlock prevention for multi-AGVs," *IEEE Access*, vol. 8, pp. 82120–82130, 2020, doi: 10.1109/ACCESS.2020.2991190.

[2] M. De Ryck, M. Versteyhe, and F. Debrouwere, "Automated guided vehicle systems, state-of-the-art control algorithms and techniques," *J. Manuf. Syst.*, vol. 54, pp. 152–173, Jan. 2020, doi: 10.1016/j.jmsy.2019.12.002.

[3] M. P. Fanti, "Event-based controller to avoid deadlock and collisions in zone-control AGVS," *Int. J. Prod. Res.*, vol. 40, no. 6, pp. 1453–1478, Jan. 2002, doi: 10.1080/00207540110118073.

[4] R. L. Moorthy, W. Hock-Guan, N. Wing-Cheong, and T. Chung-Piaw, "Cyclic deadlock prediction and avoidance for zone-controlled AGV system," *Int. J. Prod. Econ.*, vol. 83, no. 3, pp. 309–324, Mar. 2003, doi: 10.1016/S0925-5273(02)00370-5.

[5] J.-W. Yoo, E.-S. Sim, C. Cao, and J.-W. Park, "An algorithm for deadlock avoidance in an AGV system," *Int. J. Adv. Manuf. Technol.*, vol. 26, nos. 5–6, pp. 659–668, Sep. 2005, doi: 10.1007/s00170-003-2020-4.

[6] 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, doi: 10.1080/002075498192526.

[7] Q. Li, J. T. Udding, and A. Pogromsky, "Zone-control-based traffic control of automated guided vehicles," in *Coordination Control of Distributed Systems*. Cham, Switzerland: Springer, 2015, pp. 53–60.

[8] 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, doi: 10.1016/S0166-3615(99)00068-8.

[9] 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, doi: 10.1016/j.cie.2008.07.007.

[10] W. Małopolski, "A sustainable and conflict-free operation of AGVs in a square topology," *Comput. Ind. Eng.*, vol. 126, pp. 472–481, Dec. 2018, doi: 10.1016/j.cie.2018.10.002.

[11] J. Zajac and W. Małopolski, "Structural on-line control policy for collision and deadlock resolution in multi-AGV systems," *J. Manuf. Syst.*, vol. 60, pp. 80–92, Jul. 2021, doi: 10.1016/j.jmsy.2021.05.002.

[12] Y. Zhao, X. Liu, S. Wu, and G. Wang, "Spare zone based hierarchical motion coordination for multi-AGV systems," *Simul. Model. Pract. Theory*, vol. 109, May 2021, Art. no. 102294, doi: 10.1016/j.simpat.2021.102294.

[13] 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, doi: 10.1016/j.simpat.2018.07.015.

[14] Z. Xing, X. Chen, X. Wang, W. Wu, and R. Hu, "Collision and deadlock avoidance in multi-robot systems based on glued nodes," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 7, pp. 1327–1330, Jul. 2022, doi: 10.1109/JAS.2022.105710.

[15] C.-O. Kim and S. S. Kim, "An efficient real-time deadlock-free control algorithm for automated manufacturing systems," *Int. J. Prod. Res.*, vol. 35, no. 6, pp. 1545–1560, Jun. 1997, doi: 10.1080/002075497195119.

KWANWOO LEE received the B.S. and M.S. degrees in industrial engineering from Ajou University, Suwon, South Korea, in 2018 and 2022, respectively, where he is currently pursuing the Ph.D. degree with the Department of Industrial Engineering. His current research interests include modeling and simulation, simulation-based verification of AMHS design, and digital twin for manufacturing systems.

SANGCHUL PARK received the B.S., M.S., and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1994, 1996, and 2000, respectively. From 2002 to 2004, he was a Research Engineer with the ITM Department, DaimlerChrysler. He is currently a Professor with the Industrial Engineering Department, Ajou University, South Korea. His research interests include modeling and simulation, combat simulation for defense, and digital manufacturing systems.