https://doi.org/10.1016/j.aei.2019.100998

Smart robotic mobile fulfillment system with dynamic conflict-free strategies considering cyber-physical integration

C.K.M. LEE [a], BingBing LIN[a,*], K.K.H. NG [a,b,c], Yaqiong LV[d], W.C. TAI[a]

[a] *Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China*
[b] *School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore*
[c] *Delta-NTU Corporate Laboratory for Cyber-Physical System, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore*
[d] *School of Logistics Engineering, Wuhan University of Technology, Wuhan, China*
* *Corresponding author.*

Email Address: ckm.lee@polyu.edu.hk (C.K.M. LEE), bingbilin3@gmail.com (BingBing LIN), kkh.ng@connect.polyu.hk (K.K.H. NG), LVYA0001@e.ntu.edu.sg (Yaqiong LV), jade.wc.tai@polyu.edu.hk (W.C. TAI)

Abstract

Smart mobile robots are deployed to the warehouse environments to improve the efficiency, because of its characteristics of high automation and flexibility characteristics. However, the trajectory planning is a great challenge especially a number of mobile robotics operates in the warehouse simultaneously. This paper proposes a cyber-physical system model for smart robotic warehouse to implement the workflow data collection and procedure monitor. A two-stage decoupled method is presented to find a conflict-free path for the mobile vehicles in the warehouses, after distributing destinations to mobile robots to minimize the total travel distance. The improved A* algorithm is applied to find paths from the source node to the destination node for single mobile vehicles in the domain of smart logistics. The collision is detected by comparing the occupying time window of each mobile vehicle. Three collision avoidance strategies are developed to solve the conflicts and the candidate path with the minimal completion time is selected as the final determined route. The contribution of the paper is to propose a CPS-enabled robotic warehouse with dynamic conflict-free strategy to self-configure the path to optimize warehouse operation efficiency.

Keywords: Mobile robot control; conflict-detection method with time window; A* algorithm with greedy best-first-search; smart mobile robotic fulfillment system; bi-directional robot path layout; cyber-physical system

Acknowledgement

The research was supported by *the Hong Kong Polytechnic University*, Hong Kong, *Nanyang Technological University*, Singapore and *Wuhan University of Technology*, China. Our gratitude is also extended to the Research Committee and the *Department of Industrial and Systems Engineering, the Hong Kong Polytechnic University*, Hong Kong and *The Innovation and Technology Commission, The Government of the Hong Kong SAR*, Hong Kong for support of this project (ITS/275/16FX). The authors wish to acknowledge the funding support from Delta Electronics Inc and the National Research Foundation (NRF) Singapore under the Corporate Laboratory @ University Scheme (Ref. RCA-16/434; SCO-RP1) at Nanyang Technological University, Singapore. The authors would like to express their appreciation to *the anonymous case company* for their assistance with the data collection.

Smart robotic mobile fulfilment system with dynamic conflict-free strategies considering cyber-physical integration

1. Introduction

Globalization and international trade have increased the complexity of global supply chains. Due to the upsurge of online shopping, there is high demand for responsive logistics to fulfill customer requirements. Compared to the traditional logistics orders, the fragmented electronic orders are placed irregularly and the Stock Keeping Units (SKUs) have various types, which puts great pressure on the Logistics Service Providers (LSPs). The diversification of goods requires warehouse workers to spend more time on order picking, so the order fulfillment rate may be below the standard level. The Traditional Material Handling System (MHS) in LSPs limits the growth of operational efficiency, management of warehouse operations and flexibility in handling customer orders and requirements. The goods picking process in traditional MHS includes the procedure of handling pallets using pallets, conveyors and carousels [1, 2]. However, the inbound logistics performance is constrained by the productivity of human workers. In today's competitive market in LSPs, efficiency in MHS for LSPs through automation and shifting to human-robot collaborative workplaces and value creation in Software-as-a-service (SaaS) can bring notable competitive advantages in productivities, operational cost efficiency, safe working environments and value chains in E-commerce [3].

Given the rise of online service innovation, firms are gradually realizing that responsive and agile logistics operations in real time are critical factors to achieve smart supply chain operations. Internet-of-Things (IoT) solutions in smart warehouses allow organizations to achieve real-time monitoring and control of the performance of third-party LSPs [4, 5]. With the support of digitalized operations, Cyber-Physical System (CPS) enabled warehouse systems offer proximate control of physical operations in virtualized space to achieve real-time and responsive operations of inbound logistics procedures [6]. CPS integrates the communication devices, control processing units and sensory machines and robots to support the monitoring, coordination, control and management of operations [7]. Implementing CPS in smart robotic warehouse systems can enhance the productivity, flexibility and agility of the supply chain to allow resource optimization. The digitalized service in connectivity components in the system provides operational efficiency and accurate control of connected entities by real-time sensing technologies [6, 8-10].

Although the use of autonomous mobile robots in warehouse systems has brought about notable advantages including high productivity, efficient space utilization and the reduction of operational costs, the high automation and flexibility of mobile vehicles gives rise to a resource competition problem [11-13]. The development of embedded computing with wireless communication has speeded up the automation and intelligence of the logistics industry. In international LSPs or in-house logistics providers, like Amazon and Alibaba, a large quantity of smart mobile robots are deployed to improve the effectiveness and efficiency of the handling procedure [14]. However, the traffic control of multiple mobile robots is still a major concern in the automatic warehouse. As the path of the vehicle is no longer fixed, the traffic control becomes a major challenge in robotic warehouse system design. When large quantities of mobile vehicles share the same working environment, interruptions such as collisions, conflicts, deadlocks and blockages may occur. It is of importance to employ and develop an efficient path planning strategy for collision avoidance in the motion coordination of multiple mobile robots in warehouses.

The rest of the paper is organized as follows. The review of the state-of-the-art in smart robotic warehouse system is presented in **Section 2**, while the CPS-enabled architecture and model of smart robotic warehouse system are described in **Section 3**. The proposed A* algorithm with greedy best-first-search and collision avoidance strategies is illustrated in **Section 4**. Case analysis and algorithm performance are elaborated in **Section 5**. The conclusions are raised in **Section 6**.

2. Related works

The CPS system, which was first proposed by the US National Science Foundation in 2006, plays an important role in the automatic manufacturing industry, turning shop floors into factories with increased safety and security [15]. Conventional warehouse system relies on conveyors, carousels, or high-speed sorter to transport goods, which requires the warehouse layout to be fixed, and is usually costly and not expandable [16]. E-commerce warehouse typically faces the requirements of handling smaller orders, large assortment, tight delivery schedule and varying workloads[17]. Modern warehouse uses mobile robots to increase the warehouse flexibility, but its complexity also raises, which calls for control system revolution [18]. As modern warehouse management faces challenges of assuring inventory accuracy, optimizing picking procedure and space utilization, and managing procedures , implementing CPS with warehouse management system (WMS) can establish a cooperative relationship between human and smart machines so as to build an agile smart logistics [19]. CPS adopts service-oriented architecture (SOA) to manage large-scale system to provide reliable performance and improve adaptability [20]. CPS can be applied to manufacture, healthcare, military and other fields [21, 22], but there are new challenges brought about for the applications. For example, hardware devices and design methods can affect the real-time performance in multi-robots CPS, and uncertainties in the environment place a burden on the robustness, reliability and safety of CPS [23].

Traffic control methods for multiple robots can be classified as non-cooperative or cooperative [24, 25]. The non-cooperative algorithms cope with tricky issues [26], while the cooperative approaches assume uninterrupted information exchange between devices. The cooperative methods can be further divided into centralized and decentralized architecture. The centralized approach relies on a central unit to gather all task information, mobile vehicle positions and the motion status [27, 28]. In contrast, decentralized control increases the scalability and autonomy of mobile robots by peer-to-peer communication and individual decision makers [29, 30]. Several research articles are further reviewed to study their insights into multiple robots traffic control, which is summarized in **Table 1**.

There are various strategies for decentralized multiple mobile robots control. Herrero-Perez and Martinez-Barbera [31] presented a two-level topological map to represent the working environment for the vehicles. A high-level map is for the global description, while a second-level one perceives a local workplace. In order to solve the task allocation and the traffic control problems, a distributed Petri net is utilized to detect deadlock. The routing path is sub-optimal, because of the nature of the navigation process. A supervisor algorithm and navigation scheme was proposed by Demesure, Defoort, Bekrar, Trentesaux and Djemaï [29]. The supervisor generates a presumed path based on the global information, and then a priority policy is employed to solve the collision of the presumed trajectories. As the approach has to solve two optimization problems at an updated time, the calculation is complex and time-consuming. Nishi, Ando and Konishi [32] proposed an augmented Lagrangian decomposition and coordination method to minimize the total transportation time. Other relevant works detect the collision by comparing the spaces the mobile robots occupy. Purwin, D'Andrea and Lee [24] described the assignment of a nonintersecting reserved zone for each vehicle. The mobile robot can only request a moving area within its reserved area, so that the requested spaces never overlap each other. However, the performance of the method depends on the used waypoints, the cost function and the accuracy of the reserved/requested area. Draganjac, Miklić, Kovačić, Vasiljević and Bogdan [30] constructed a state lattice to connect current state with possible nearby states. A low-priority mobile robot gives way to a high-priority mobile robot, so that each mobile robot is safe in its own private area. The safety space allocated to the vehicle is significantly larger than the minimal required zone; thus, the number of unnecessary stops is increased.

Despite the heavy computational load and the low failure tolerance property, the centralized architecture can find an optimal solution and provide a comprehensive performance overview [33]. Centralized control is further classified as coupled and decoupled. The coupled methods are complete, treating the whole traffic control procedure as a whole and composite system. Reveliotis [34] employed zone control to resolve conflicts by allowing only one mobile robot in a zone at a time. Banker's algorithm is developed to prevent resource competition. The performance is not efficient, since a mobile robot has to wait until a shared space is released by another mobile robot. The column generation technique is one of the methods to optimize the NP-completeness formulation [35]. The meta-heuristic scheme is another useful tool to obtain a near-optimal solution [36]. A two-stage Ant Colony Algorithm (ACA) was analyzed to solve the mixed-integer mathematical problem [37]. These coupled mechanisms have high computational demand and the growth of the computation time is intractable with the increment of the number of mobile robots.

Decoupled approaches are prevailing to reduce the computation complexity of the coupled methods, which break the routing problem into path planning and motion coordination phase. Some researchers tend to solve the collision using reactive strategies considering real-time operations. Zhang, Guo, Chen and Yuan [14] designed a smart control model consisting of a car-following model, overtaking model, and collision warning and avoidance model to regulate the traffic. Different rules are formulated to guarantee the safety, but the model is based on a two-lane factory layout and does not apply to a narrow warehouse. Maza and Castagna [28] focused on employing the waiting strategy to solve head-on conflict, ignoring other collision types like cross conflict and node-occupancy collision. A Colored Resource-oriented Petri Net (CROPN) was offered to avoid deadlock and blocking [38]. The deadlock avoidance policy makes sure that one mobile robot can only arrive a node after the other mobile robot leaves the same node. If a lane is damaged, the CROPN method does not work well. Gawrilow, Köhler, Möhring and Stenzel [27] demonstrated a generalized arc-based label-setting algorithm to resemble Dijkstra's algorithm, but there is no detailed result discussion to prove the performance of the method. Dynamic programming combined with Dijkstra's shortest path algorithm was studied as well [39, 40]. However, the heuristic function to optimize the number of maneuvers is not illustrated clearly. A time window algorithm was provided by Smolic-Rocak, Bogdan, Kovacic and Petrovic [41], but they only noticed the head-on conflict. Zhang, Guo, Chen and Yuan [14] categorized the collision classifications into four summarized collision types and provided a collision-free route planning solution. The single path-finding approach, Dijkstra's algorithm, wastes resources to perform blind searching and the determined collision-free route is not always optimal.

Table 1

Author(s)	Methodology	Advantage(s)	Limitation(s)
Herrero-Perez and Martinez-	The environment is represented by a two-level topological map. To coordinate	The work provides a control with high	The routing path is sub-optimal.

Barbera [31]	multiple robots, a decentralized control is adopted to solve navigation conflicts, while the task allocation and traffic problem is solved by the distributed Petri net.	autonomy and scalability. The task planning is simplified.	
Demesure, Defoort, Bekrar, Trentesaux and Djemaï [29]	A supervisor algorithm is used to generate a presumed path, and then a priority policy are introduced for collision control.	The proactivity of the supervisor algorithm and the reactivity of the motion planner is guaranteed.	The calculation is time-consuming.
Nishi, Ando and Konishi [32]	A distributed method adopting an augmented Lagrangian decomposition and coordination technique is proposed to minimize the total transportation.	In large-scale problem, the computation time is short.	The method is not applicable for
Purwin, D'Andrea and Lee [24]	A decentralized algorithm assigning noon-overlapping reserved area to agents is introduced and a cost-based negotiation process is presented to resolve conflicts.	The approach is robust with regard to wireless latency, bandwidth limitation, error correction, and network breakdown.	The performance depends on the waypoints, the cost function and the reserved area.
Draganjac, Miklić, Kovačić, Vasiljević and Bogdan [30]	A state lattice is developed to determine the vehicle's private zone and a priority scheme is performed to avoid conflicts.	The methodology is fully decentralized and application invariant.	The number of unnecessary stops is increased.
Reveliotis [34]	A zone control approach is provided to plan paths and a banker's algorithm is modeled to solve resource competition.	The robustness of conflict resolution is emphasized by the structural and performance-oriented control.	The performance is not efficient enough.
Zhang, Zhu and Lv [15]	A smart control model based on cyber-physical system is designed, which includes car-following model, overtaking model, and collision warning and avoidance model.	Vehicles with urgent tasks can overtake others. The system can predict and avoid collisions without stopping vehicles.	The layout is two-lane, so the model is not applicable to a narrow warehouse.
Gawrilow, Köhler, Möhring and Stenzel [27]	A generalized arc-based label-setting algorithm is used to find conflict-free routes.	The algorithm avoids collisions, deadlocks, and livelocks in route computation rather than in the execution time.	The performance of the methodology is not discussed in detail.
Smolic-Rocak, Bogdan, Kovacic and Petrovic [41]	A time window elongation method is proposed to determine the shortest path.	Circular paths are allowed.	The method only solves head-on conflict.
Zhang, Guo, Chen and Yuan [14]	The collisions are classified and the different collision resolutions are presented.	The efficiency of the automated warehouse system is increased.	The method using Dijkstra's algorithm performs blind search.

Abundant research has been made to either investigating the characteristics of CPS or exploring trajectory planning for robotic mobile fulfillment system. As the CPS is a new concept, most researchers focus on the theoretical study rather than the real-life applications. Some traffic control approaches are based on the mathematical model and their computation complexity is high. Their methods may not be applicable to real life situation. In this research, CPS on the warehouse system enables practitioners to centralize and control the mobile robot through the coupling of the physical material handling with the digitized process for trajectory planning as base for optimizing the warehouse efficiency. Moreover, a number of research resolves conflicts by controlling one mobile robot to stop in front of the collision point. The acceleration and deceleration consumes lots of energy, so a wait-before-startup strategy is presented in this paper to improve energy conservation. Zhang, Zhu and Lv [15] developed a CPS-based control model for material handling while the layout is a two-lane uni-direction configuration. To improve the space utilization, a one-lane bi-direction layout is implemented in this research.

3. CPS-enabled Smart Robotic Warehouse System

3.1. System framework for a Cyber-physical robotic warehouse system

The proposed CPS-based smart robotic warehouse system consists of five layers, including the connection layer, the conversion layer, the cyber layer, the cognition layer, and the configuration layer, as shown in **Fig. 1**. Each tier performs its respective duties, and cooperates with other tiers to exchange information and data.

The connection layer, as the bottom layer, utilizes the plug and play sensors on the hardware devices such as workstation, charging station and mobile vehicle, to obtain accurate and reliable data. The sensor network gives the devices the ability to self-connect and self-sense, establishing tether-free communication. The selection of sensors and controllers is the main consideration in the first layer.

The conversion layer applies several mechanisms for data-to-information conversion. The raw data are transformed and standardized to obtain device-related knowledge, such as the speed and position of mobile robots, the progress of workstations, and the status of the charging station. The conversion layer is responsible for bringing self-awareness to hardware devices.

The cyber layer is used to gather massive information for building the virtualization twin of the real world. The digital twin model in this middle layer connects the physical world with cyberspace, so various data analytics approaches can be employed to additionally extract valuable information. Comprehensive insight can be gained for further decision-making in the upper layer.

The cognition layer properly presents the extra analytic knowledge to expert users. As the information transport from the cyber layer provides better synthetic simulation of the monitored system, experts can make rational decisions for different procedure situations. Both historical records and real-time information supports collaborative diagnostics.

The configuration layer acts as the resilience control system in the CPS framework. This stage performs supervisory control by giving feedback from cyberspace to the physical world. Self-configured and self-optimized operations between workstations, mobile robots and order fulfillment requests can be archived at this level. The intelligence of the devices is finally shown, so the warehouse productivity can be enhanced.

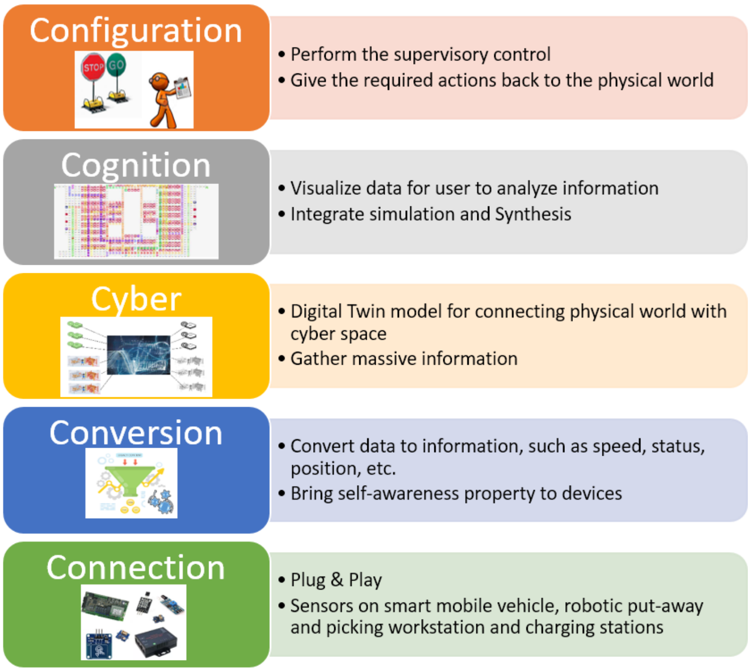

Fig. 1. Framework of a CPS-based smart robotic mobile fulfillment system

3.2. A Cloud-based robot control and warehouse management system

The cloud platform serves as the brain of the smart robotic warehouse system, reflecting the intelligence and cognition of the whole system [42, 43]. The cloud-based system utilizes a SaaS model to offer scalable usage, real-time visibility and seamless integration [44]. There are two main subsystems (marked as green in **Fig. 2**) in the cloud: WMS and Robot Control System (RCS). WMS is responsible for daily in-coming and out-going warehouse operations, managing and controlling all relevant resources to ensure smooth functioning, while the RCS focuses more on the coordination control of mobile robots, supervising their status and issuing commands. The proposed model is shown in **Fig. 2**.

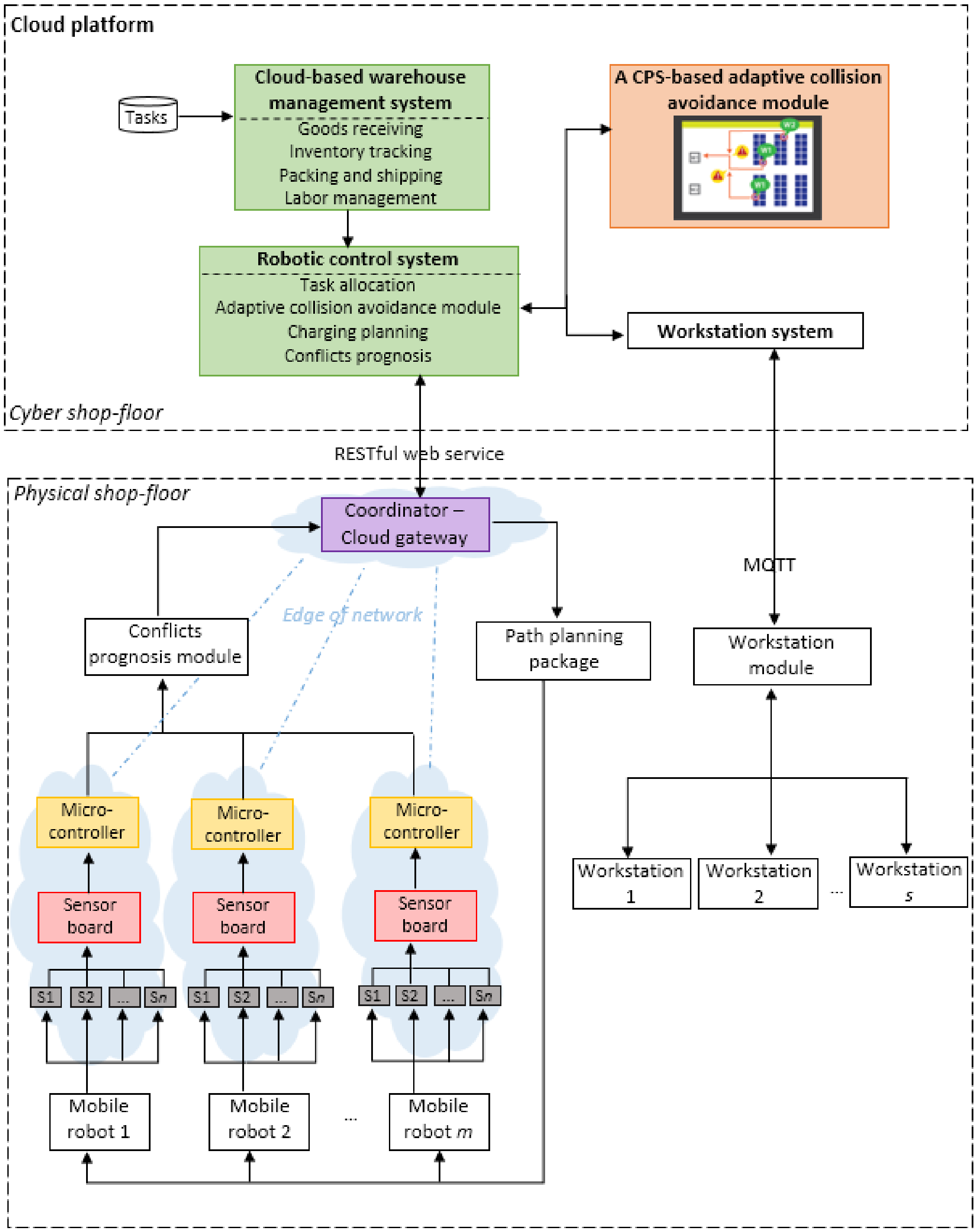

Fig. 2. A CPS-based smart robotic mobile fulfillment model

3.2.1. Cloud-based warehouse management system

Cloud-based warehouse management system (shown in upper part of **Fig. 2**) plays a vital role in order fulfillment, taking on workflow organization and procedure monitoring. It tracks inventory flow, order completion and labor management. The aim of WMS is to optimize resources allocation to improve the overall performance of the supply chain to maintain high customer satisfaction. WMS performs an inventory cycle count to guarantee inventory records' accuracy in the database and to call for error handling actions for missing or broken items. As WMS helps managers monitor workers' performance and warehouse workflow, it can present an analytic report to give the management team some valuable suggestions.

When receiving goods, the system determines the storage location according to assignment policies related to the product category, space utilization, turnover rate and group clustering. Goods frequently bought together are stored on the same rack to save the picking time, so the efficiency can be improved. Moreover, a re-slotting strategy is carried out at a predefined period. Locations of racks are rearranged based on historical records, on-hand orders and goods turnover. WMS measures orders by their similarity, and then distributes them to different workstations, while the picking process is handled by the RCS. Once order picking is completed, WMS instructs the packing and labeling operations for packages, and generates shipping lists based on destinations and order category. Generally, WMS services as a stock guardian in the warehouse model. It performs quality assurance procedure to check the quantity of incoming goods, followed by continuously tracking the stocking life cycle of inventory. When goods are to be dispatched, WMS is responsible for the packing and shipping process to ensure the right goods are delivered to the right customers.

3.2.2. Robot control system

The robot control system (shown in upper part of Figure 2) is a centralized system to gather all information related to mobile robots and to decide the actions of mobile robots in various situations. Receiving assigned tasks from the upper system WMS, the RCS makes an overall decision to determine the optimal mobile robot for the given tasks. As the mobile robots may collide with one another during the task execution period, the RCS needs to coordinate their motion carefully to guarantee safety. The strict control relies on the latency-sensitive real-time data information from the physical shop floor.

Mobile robots are in charge of the replenishment and pick-up procedure. Diverse factors like robot location, robot status, and battery level, are taken into consideration by the RCS to allocate tasks. In some special conditions, the RCS governs several mobile robots to cooperate to accomplish a complex task. The division of labor strategy reduces the task difficulty and puts less stress on the individual intelligence of mobile robots. The RCS provides a path-planning algorithm for on-duty mobile robots. The RCS predictively simulates the continuous motion of mobile robots, the system prognoses possible conflicts by the estimated time-space occupation of mobile robots in a prior way. The conflict prognosis function is a crucial feature of the RCS, since it reduces the probability of accidents. In addition, the RCS schedules a charging plan for all mobile robots. The charging scheme is adjustable and customized according to different user requirements, and can be rescheduled afterwards. For example, when the battery level of a mobile robot drops below 65%, the RCS will allocate it to move to the charging station. If multiple mobile robots compete for the charging station resources, the RCS may queue them to give way to the emergent mobile robot. Mobile robot with lower battery level gets a higher priority to use the charging station, while mobile robot having higher electricity power can continue executing tasks first and recharge afterwards. Therefore, the resources can be optimally utilized to improve comprehensive performance.

3.3. Edge of network on the physical shop-floor

The edge of network (whose main components are marked as blue in **Fig. 2**) is built up in the physical layer to transmit information to cyberspace. Its main duty is to control and monitor the shop-floor facilities, and to offload part of the workload of the cloud system. As edge of network allows physical devices to automatically sense and react to the working environment, the computing complexity of the cloud system can be reduced. The combination of edge of network and the cloud system is beneficial to overall flexibility and robustness. In the event of hardware or system failure, the shop floor can quickly recover from the interruption and resume normal operation. To increase the awareness of the warehouse environment condition, sensors on machine devices was deployed to measure the operating status of machine devices continuously. Then, all outputs of sensors are integrated to a micro-controller in each machine. All data from the micro-controllers are transmitted to the cloud server through a gateway.

3.3.1. Data acquisition on mobile robots

Mobile robots are valuable assets of the physical shop floor, so are one of the important edge devices in the edge of network. The inherent characteristic of the edge of network requires mobile robots to have built-in intelligence to manage data and make decisions locally. Data acquisition is the cornerstone for mobile robots to process data. Mobile robots rely on data acquisition to extract reliable real-time data and to surveil the surrounding environment.

A multi-sensor system is adopted in mobile robots to realize automation. A camera is installed at the bottom of the mobile robot to scan the Quick Response (QR) code stuck on the ground, so mobile robots can locate and map the shop floor. A laser range finder is used to detect the distance between a mobile robot and obstacles, and then corresponding actions can be taken by the mobile robot to avoid potential collisions. A motor board controls the motion of mobile robots and monitors the simultaneous motion speed. Since the Programmable Logic Controller is tolerant of extreme conditions, it is embedded to handle the input/output of sensors. The plug-and-play property makes it advantageous to set up sensors and to track the generated data easily.

3.3.2. Wireless sensor network

Wired connection limits the expansion and relocation of a system, so a Wireless Sensor Network (WSN) is set up to transmit data acquired on physical devices. A WSN consists of a large quantity of sensor nodes providing sensor data and several gateways centrally organizing collected data. Sensor nodes are spatially distributed on the shop floor to get a general picture of the environment. These nodes are capable of self-organization and local computing.. When the sender sends packet out, it waits for the acknowledgement from the upper node. If the sender does not receive thee acknowledgement, the sender will resend the lost packet. MQTT is a communication protocol in the network, which supports the setting of Quality of Service (QoS). The level of QoS is set to guarantee the retransmission of the lost packet. As some devices continue sending packets in a specific high frequency, the loss of one or two packets does not affect the overall performance of the system. Therefore, a WSN is stable and reliable in the case of single node failure.

Mobile robots serve as sensor nodes in a WSN, increasing the network flexibility and mobility. Mobile robots share information by the embedded wireless communication device. Data gathering in sensor nodes is sometimes event-driven to decrease the packet size and to reduce energy consumption. A WSN in a physical shop floor adopts hierarchical topology to manage sensor nodes and gateways, so the system supports a dynamic structure and is immune to inference. Message Queuing Telemetry Transport (MQTT) is chosen as the communication protocol in a WSN to standardize data exchange. Since MQTT is built on the TCP/IP protocol, the connection is reliable. Moreover, token authentication is added to strengthen network security. As a publish/subscribe protocol, MQTT has the advantages of low network bandwidth, efficient information updating and low power consumption.

3.4. Adaptive collision avoidance module

An adaptive collision avoidance module (marked as orange in **Fig. 2**) is an indispensable component in the cloud-based RCS, coordinating the movement of mobile robots for resource competition. Although mobile robots are equipped with a laser sensor to detect obstacles in front, possible obstacles on both sides are hard to be observed by mobile robots. Thus, the RCS firstly takes a conflicts prognosis approach to detect possible collisions, and then adopts the collision avoidance module to adapt to the competition situation. As mobile robots keep moving to complete assigned tasks, collision avoidance modules should have the capacity for situation awareness to timely respond to changes in the environment. Detailed steps of the adaptive collision avoidance module are described in section 4.

The location of mobile robot changes with time, so a collision avoidance module is proposed based on time window. Several avoidance strategies are summed up in the module to guarantee that only one mobile robot occupies a space location at one time. Constraints of mobile robots are considered to enhance the feasibility of the module. As the cloud system continuously supervises the physical environment condition, the collision avoidance module can timely adjust the robots' behavior to prevent potential conflicts. The feedback from the CPS-based module is further transmitted to the physical shop floor to reflect the cognition of the central controller.

4. A CPS approach for collision detection with time window

When new orders are coming, the WMS in the CPS model receives them and schedules them by received dates and customers' priorities. These orders then further pass to the RCS. RCS allocates the orders to different workstations. Once the workstations set to work, RCS subdivides order lines into task set F and fetches the robots' data from the cloud database. As the warehouse layout is predesigned, the layout information is stored in the database. The grids ID of the layout and the initial grids of mobile robots are compared to determine the initial coordinate of mobile robots. The coordinates of workstations, mobile robots, racks and aisles form the nodes set V. Fetching the allowable directions of each node with the robot status, an edge set E is established to represent the nodes adjacency relationship. The CPS approach for collision detection with time window is formulated using a directed graph $G = (V, E)$ with a set of nodes V and a set of arcs E.In the model, we consider that a set of tasks $t \in F$ is handled by a set of mobile robots $i, j \in I$ using path planning and collision avoidance approaches. Initial coordinate and dwell coordinate of each mobile robot are denoted as $(x_i^o, y_i^o) \in V_i$ and $(x_i^d, y_i^d) \in V_i$, while the mobile rack's coordinates are described as $(\mathrm{x}_r, y_r) \in V$. A predetermined path contains a set of travel nodes $p_i = (u_i^o, \dots, u_i^n)$, which start from node u_i^0 and end at node u_i^n. The estimated arrival and departure time for mobile robot $i \in I$ on node $u \in V_i$ are illustrated by an interval of $[\mathrm{AT}_{iu}, DT_{iu}]$. Regarding the choices of paths p_i and p_j, the model considers the conflict detection on the intersected nodes $\forall \mathrm{u} \in V_i \cap V_j$. The threshold for the collision detection τ is also introduced in the conflict detection. The notations and variables are explained in **Table 2**.

Table 2

Notation and variables

Notations	Explanation
i, j	The ID of mobile robot, $i, j \in I$
t	The ID of the task $t \in F$
r	The ID of mobile rack, $r \in R$
u, v, π	Nodes $u, v, \pi \in V$
p_i	Path $p_i \in P_i$ for mobile robot $i \in I$

x_i^d, y_i^d	The abscissa and ordinate of the mobile robot's dwell coordinate $(x_i^d, y_i^d) \in V_i$
x_i^o, y_i^o	The abscissa and ordinate of the mobile robot's initial coordinate $(x_i^o, y_i^o) \in V_i$
x_i^n, y_i^n	The abscissa and ordinate of the mobile robot's destination coordinate $(x_i^n, y_i^n) \in V_i$
x_r, y_r	The abscissa and ordinate of the mobile rack's dwell coordinate $(\mathrm{x}_r, y_r) \in V$
AT_{iu}	The estimated arrival time for mobile robot $i \in I$ on node $u \in V_i$, $\mathrm{AT}_{iu} \geq 0$
DT_{iu}	The estimated departure time for mobile robot $i \in I$ on node $u \in V_i$, $\mathrm{DT}_{iu} \geq 0$
WT_i	The total waiting time for mobile robot $i \in I$
ψ_i	The number of waiting for mobile robot $i \in I$
Ψ	The maximum number of waiting
ω	The constant waiting time
φ, τ	The threshold for the collision detection
β_i	The priority factor for mobile robot $i \in I$
w	The predetermined reward factor for mobile robot
ET_i	The estimated time for current candidate path for mobile robot $i \in I$, $ET_i \geq 0$
OT_{it}	The operation time of mobile robot $i \in I$ on task $t \in F$
c_n	The turning cost of mobile robot operations
c_l	The travelling cost of mobile robot with lifting operations in aisle area
c_φ	The travelling cost of mobile robot without lifting operations in aisle area or storage area
c_θ	The travelling cost of mobile robot from initial coordinate to dwell coordinate
S_i	The node search list for mobile robot $i \in I$
σ	The first node of the node search list for mobile robot $i \in I$, $\sigma \in S_i \cap V$
M	Large artificial variable
z_{iju}	1, if mobile robot i enter on node u before mobile robot j (not necessary immediately); 0, otherwise
δ_{it}	1, if task $t \in F$ is assigned to mobile robot $i \in I$; 0, otherwise
γ_{ir}	1, if mobile robot $i \in I$ carries rack $r \in R$; 0, otherwise

4.1. Improved A* algorithm for single mobile robot path planning

Dijkstra's algorithm is one of the famous approaches to find the shortest path from the starting node to the destination node in a directed graph whose edges' costs are all nonnegative. As a greedy algorithm, the efficiency of Dijkstra's algorithm is $O(|E| + |V| \log|V|)$, where $|E|$ represents the number of edges in the graph, $|V|$ the number of vertexes. Unlike Dijkstra's algorithm favoring the node close to the source node, the Greedy Best-First-Search method considers the node closer to the destination first. However, this approach does not guarantee a proof-of-optimal condition. To further improve the computation efficiency, we proposed an integrated approach of Dijkstra's algorithm and the Greedy Best-First-Search approach. Although the improved A^* algorithm is a heuristic method, it is guaranteed that the best path can be found by using the algorithm and its time efficiency is O(|E|).

The cost $f(u)$ from the starting node to the destination node in the A^* algorithm consists of two kinds of costs—one denoted as $g(u)$ which is the exact cost from the dwell node (x_i^d, y_i^d) to the starting node (x_i^o, y_i^o) and the other denoted as $h(u)$ is the estimated cost from the current node (x_i^d, y_i^d) to the destination node (x_i^n, y_i^n). The heuristic function of $h(u)$ is stated in Equation (1). The turning time of the movement is taken into consideration in the $g(u)$ cost calculation to make the route cost more close to reality. Since the heuristic function $h(u)$ has an influence on the performance of the algorithm, it is important to choose the calculation function. In the warehouse environment, a mobile vehicle has to make a U-turn to change its direction, so possible moving directions are due-east, due-west, due-north and due-south. Manhattan distance is a common cost for the graph with square grids allowing four directions of movement. Therefore, the improved A^* algorithm use Manhattan distance as the heuristic function and will find all possible paths from the starting node to the destination node.

$$h(u) = |x_i^n - x_i^d| + |y_i^n - y_i^d| \quad (1)$$

The improved A* algorithm reserves all the possible paths with the same distance from the start node (x_i^o, y_i^o) to each intermediate node (x_i^d, y_i^d), and searches repeatedly until it traverses to the destination node (x_i^n, y_i^n). The rack grid occupied by the rack is available for a non-load mobile robot, but not accessible for a loaded mobile robot. Therefore, the possible intermediate nodes (x_i^d, y_i^d) should be decided not only by the available moving directions, but also by the lifting status of the mobile vehicle. The feasible routes from the source node u_i^o to the goal node u_i^n are transmitted to the mobile robot for storage. The route with the minimal cost will be taken into consideration firstly.

The pseudo code of the A* algorithm with greedy best-first-search is shown in **Algorithm 1 錯誤!** 找不到参照來源。.

Algorithm 1. The pseudo code of the improved A* algorithm

Input	*G=(V,E)*, task $t \in F$, Initial coordinate for each mobile robot x_i^0, y_i^0, costs $c_n, c_l, c_\varphi, c_\theta$
Output	$p_i, \delta_{it}, \mathrm{AT}_{iu}, DT_{iu}$
1	Assign each task $t \in F$ to each mobile robot using the task allocation variables δ_{it}
2	Determine the initial and destination coordinates of the mobile robot x_i^0, y_i^0 and x_i^n, y_i^n, respectively
3	Initialize an empty set of possible paths P_i
4	Put the initial coordinate of the mobile robot x_i^0, y_i^0 in the search list S_i
5	**While** the node search list S_i is not empty **do**
6	Fetch the first node σ in the node search list S_i regarding the cost c_θ
7	Put the path p_i to the path set P_i if the top node σ is equivalent to the destination node (x_i^n, y_i^n)
8	Search any adjacent nodes that connect to the dwell node (x_i^d, y_i^d)
9	**For each** adjacent nodes that connect to the dwell node (x_i^d, y_i^d)
10	Compute the cost function $f(u)$ $f(u) = \begin{cases} h(u) + c_\theta + c_n, if\ mobile\ robot\ is\ turning\ on\ node\ u \\ h(u) + c_\theta + c_l, if\ mobile\ robot\ is\ lifting\ a\ rack\ and\ travel\ to\ the\ next\ node\ v \\ h(u) + c_\theta + c_\varphi, if\ mobile\ robot\ is\ not\ lifting\ a\ rack\ and\ travel\ to\ the\ next\ node\ v \end{cases}$
11	Put the adjacent node to the node search list S_i if it has not been visited

4.2. Collision-detection method with time window

Once all mobile robots on assignment receive the routes from their source nodes (x_i^o, y_i^o) to their destination nodes (x_i^n, y_i^n) generated by the improved A* algorithm, the possible collision between every two mobile robots is detected iteratively for any pair of mobile robots $\forall i, j \in I, i < j$. The assigned time slot of a mobile vehicle in each node on the routes is stored for further processing, so the overlapping movement of the vehicles can be examined by comparing their time window. There are mainly three types of collisions in the smart robotic warehouse environment: stay-on conflict, head-on conflict and cross conflict in **Fig. 3**, **Fig. 4** and **Fig. 5**, respectively.

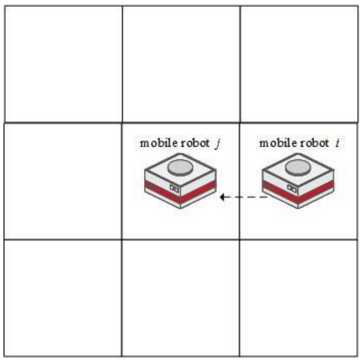

Fig. 3. stay-on conflict

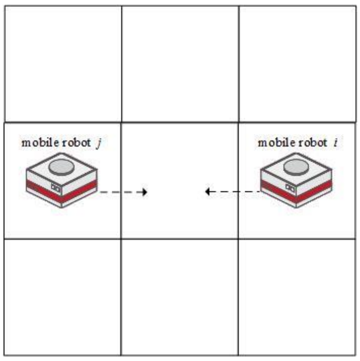

Fig. 4. Head-on conflict

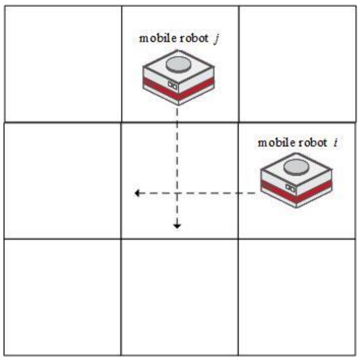

Fig. 5. Cross conflict

The time window is illustrated by an interval of $[\mathrm{AT}_{iu}, DT_{iu}]$. As for the stay-on and cross conflict, any pair of p_i and p_j should satisfy Equation (2). The departure time with threshold $DT_{iu}+\varphi$ of mobile robot i on node u must be earlier than the arrival time AT_{ju} of mobile robot j if mobile robot i will first cross or arrive at node u before mobile robot j. The head-on conflict can be detected using Equation (3). The algorithm evaluates all pair of modes $(u, v) \in E_i \cap E_j$ and computes any head-on conflict between the departure time of mobile robot i and arrival time of mobile robot j for all pair of mobile robots $i, j \in I$. Valid routes must satisfy Equations (2) and (3).

$$DT_{iu} + \varphi \leq AT_{ju} - M(1 - z_{iju}), \forall i, j \in I, i \neq j, \forall u \in V_i \cap V_j \tag{2}$$

$$DT_{iu} + \varphi + \tau \leq AT_{ju} - M(2 - z_{iju} - z_{ijv}), \forall i, j \in I, i \neq j, \forall (u, v) \in E_i \cap E_j \tag{3}$$

4.3. Avoidance strategies

Once conflicts are detected, conflict avoidance strategies are adopted. There are various strategies available to guarantee the safety of the smart robotic warehouse system. Three approaches are proposed in this paper: go-away, detour, and wait-before-startup. The priority of the mobile vehicle is computed by Equation (4). The weight factors are w_1 to w_4. The proposed priority function consists of weighted value of task assignment variable δ_{it}, the current status of rack lifting γ_{ir}, estimated time of current candidate path ET_i and the incumbent operation time of mobile robot i on task t.

$$\beta_i = \sum_{t}^{F} w_1 \delta_{it} + \sum_{r}^{R} w_2 \gamma_{ir} + w_3 ET_i + \sum_{t}^{F} w_4 OT_{it}, \forall i \in I \tag{4}$$

Table 3 explains the conflict resolution corresponding to the mobile robot conflict scenarios. The final chosen path is the path, which not only does not collide with other mobile vehicles, but also makes the mobile robot arrive at its destination node as fast as possible. A preferred solution is to change the route of mobile robot with low priority. However, if the collision cannot be avoided by altering the route of a mobile robot with low priority, the route of a mobile robot with high priority will be altered to ensure safety. If the conflict cannot be prevented by all three strategies, the task scheduler will re-dispatch tasks of the mobile vehicle with low priority. In the proposed approach, we compare any combinations of mobile robots j and i for all mobile robots $j, i \in I, j \neq i$ in the evaluation of the collision scenarios two time per second. Therefore, we can yield a solution that covers all collision scenarios by permutations with more than two mobile robots.

Table 3

Strategies and the corresponding conflicts

Strategy	Stay-on Conflict	Head-on Conflict	Cross Conflict
Go-away Strategy	√	X	X
Detour Strategy	X	√	√
Wait-before-startup Strategy	X	√	√

Go-away strategy

The go-away strategy redirects mobile robot j to another adjacent node and allows mobile robot i to remain on the same path for operations. Stay-on conflict happens when there are no other methods to prevent the collision except asking the idle mobile vehicle to give way to the running mobile vehicle. The go-away algorithm is similar to the improved A* algorithm, but the destination is indeterminate. The destination grid of the idle mobile robot will be the grid, which is not one of the path nodes of other vehicles in the Storage Area. The new route will go through the process of collision detection to make sure that the two mobile vehicles have no conflicts.

Detour strategy

The detour strategy attempts to select a candidate route from the previous generated routes from the improved A* algorithm. The routes obtained from the improved A* algorithm are ordered based on their cost. The path with lower cost from the source node to the destination node has a higher priority to be selected. The candidate routes of mobile robot j are evaluated one-by-one to validate if the new route has a conflict with mobile robot i's route. Once the candidate route is conflict-free with mobile robot i's path, it is chosen as the determined detour path.

Wait-before-startup strategy

The wait-before-startup strategy aims to reduce energy wastage, since it consumes a lot of electricity during the starting moving and braking period. This strategy puts off the startup of the mobile vehicle, so the mobile vehicle does not need to pause after it runs. The waiting time WT_i is computed by Equation (5), which is a multiplication of the number of waiting ψ_i and a constant waiting time ω. The number of waiting ψ_i should not exceed the Ψ value. Therefore, $\psi_i = [0, \Psi]$. Once the route adding the waiting time is conflict-free with other mobile vehicles' routes, it is the determined path with the waiting before start-up strategy.

$$WT_i = \psi_i \omega \tag{5}$$

Once all candidate routes are generated by these strategies, their arrival time in the destination node is compared. The route which provides an earlier arrival time is finally decided as the moving path for mobile robots, such that the mobile robots can move to the destination with the least required operation time.

5. Numerical analysis and computational results

5.1. Shop-floor layout and functions of mobile robots

The shop-floor is divided into different grids. Normally, the size and number of grids are determined by the mobile vehicle and the warehouse space, so the grid is set to be large enough to accommodate a mobile vehicle. Each grid is marked with the two-dimensional QR code label posted on the ground, such that the mobile vehicles can quickly verify its current location by reading the QR code label.

There are basically four functional areas in the topological-represented warehouse system: Picking and Replenishing Workstation, Aisle Area, Storage Area and Charging Station. The Picking and Replenishing Workstation is the workstation area where the operators pick up the goods from the arrived rack and put them on the specific workstation carbons to meet the order request, while the Aisle Area is the aisle space where the autonomous vehicles can freely move no matter whether the mobile robot lifts the rack or not. The Storage Area is the zone in which to place racks for storing commodities. As shown in **Fig. 6 錯誤! 找不到參照來源。**, the rack's bottom is 31.5 cm from the ground and the height of the mobile vehicle is 29.5 cm, so a mobile robot can pass through under the rack in the Storage Area when it is free of loads. Finally, the Charging Station is where the mobile vehicle can charge when its power is less than 40%. Once the mobile robot arrives at the charging grid, it will recharge automatically until its battery level approaches 80%.

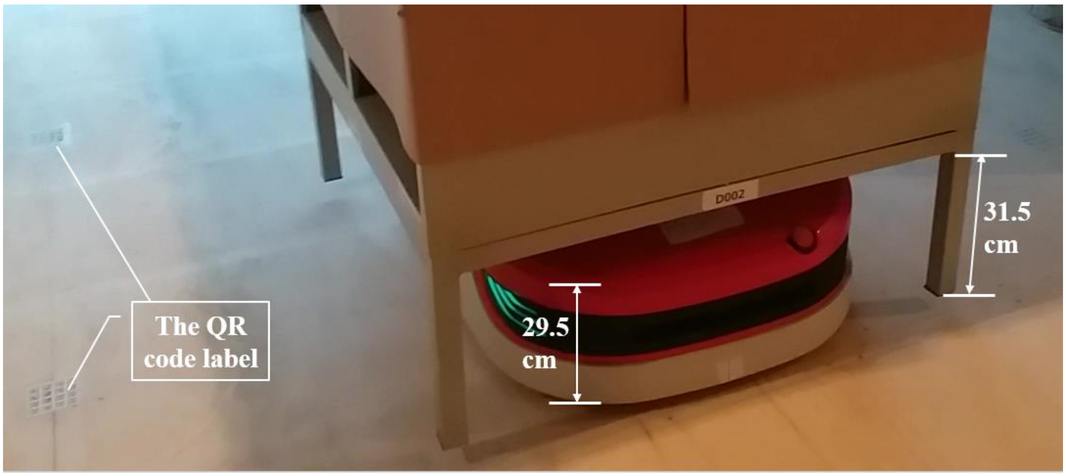

Fig. 6. A mobile vehicle lifting a rack

The mobile vehicle is responsible for two kinds of tasks. On one hand, empty mobile robots should travel from the source grid to the destination grid and then lift the rack there. The empty vehicle can move under the racks. On the other hand, the loaded mobile robots carrying the rack aims at transporting the goods from the source node to the workstation for operators to pick up and then send the rack back to the storage area. Since the loaded mobile robots should avoid the storage area, which is occupied by the racks, the routing path will be different for empty vehicles and vehicles lifting a rack

The following assumptions have been made in the algorithm design process.

- The warehouse supports bi-directional movement
- The size of the grid is fixed and is large enough to accommodate a mobile vehicle
- A mobile vehicle can only execute one task at a time
- The speed of the mobile vehicle is constant and the vehicle can stop immediately
- The mobile vehicle needs to make a U-turn to change the moving direction and the turning time is constant

The computation was performed using an Intel Core i5-6500 3.20GHz CPU and 16 GB RAM in the *Windows 10 Enterprise 64-bit* operating environment. The improved A* algorithms were coded using *Java* language.

5.2. Case analysis

. The cyber physical integration for warehouse operations is illustrated as following. Each grid's type, ID, cost and coordinate were stored in the cloud database. Five mobile robots operates in the warehouse. The mobile robots' initial coordinates were got by scanning the QR code stuck on the floor. Data related to the robot's position, status, loading, battery level were sensed and transmitted to the CPS-enabled warehouse' cloud by WSN. Four new tasks were inputted into the WMS to test the functions of the CPS. After the preprocessing the tasks, WMS delivers these tasks to RCS. RCS fetches mobile robots' status and coordinate to perform task allocation. Receiving tasks from RCS, each robot individually conducts path planning operation. Then, the candidate routes were transported to RCS by network, and RCS calls for adaptive collision avoidance module to resolve collisions. Finally, the conflict-free paths were sent back to each mobile robot, so the robot could start moving.

The proposed algorithm was compared with traditional methodology, which uses FCFS strategy to assign destinations and applies the wait-before-collision-node approach to avoid collisions. This traditional common method is similar to the wait-before-startup strategy except for the waiting location. This approach waits at the grid node right before the collision node, while the previous method waits at the source position to save power. The experiment was designed with mobile vehicles and the destinations were densely distributed with 10*20 grids..

In the dense-distributed experiment, one mobile vehicle was designed to block the way during the first 30 seconds. The initial coordinates of mobile robots were x_i^o, y_i^o, where $|I| = 5$, is (1, 2), (3, 2), (2, 6), (1,3) and (3, 8), respectively, while the destination nodes were located at (5, 18), (5, 17), (7, 18) and (4, 16). The destination allocation of FCFS strategy was $x_1^n, y_1^n = (5{,}18)$, $x_2^n, y_2^n = (5{,}17)$, $x_3^n, y_3^n = (7{,}18)$ and $x_4^n, y_4^n = (4{,}16)$, while the destination allocation by improved A* algorithm was $x_1^n, y_1^n = (4{,}16)$, $x_2^n, y_2^n = (7{,}18)$, $x_3^n, y_3^n = (5{,}18)$ and $x_4^n, y_4^n = (5{,}17)$. The predetermined paths by the A[*] algorithm of our algorithm are shown in **Fig. 7**.

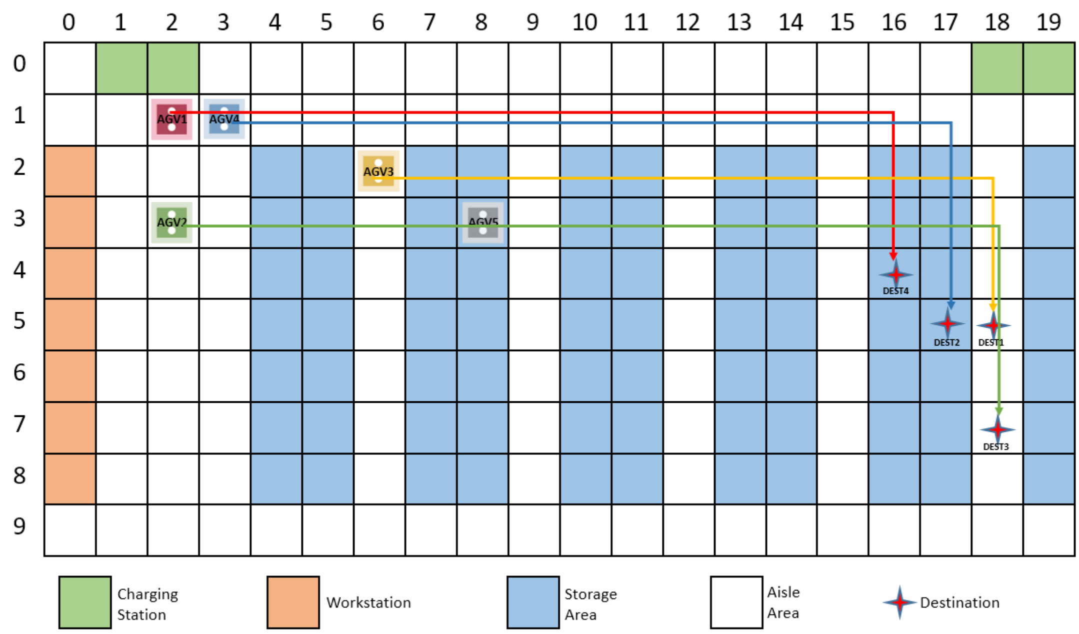

Fig. 7. The predetermined paths of our algorithm in the dense-distributed experiment

When implementing the collision detection and avoidance step iteratively in our proposed algorithm, there were several collisions detected and solved by the corresponding strategy. Firstly, mobile robot 1 had a collision with mobile robot 4 at node (1, 3), so the wait-before-startup strategy was applied for mobile robot 1. There was also a cross collision between mobile robots 2 and 3. For mobile robot 3, the detour strategy provided a solution with a completion time of 24 s while the wait-before-startup strategy gave a path with a completion time of 25 s, so the detour strategy was accepted. Meanwhile, mobile robot 2 found that the idle mobile vehicle mobile robot 5 is an obstacle in its operating path, which was a stay-on conflict, so it asked mobile robot 5 to go away. The completion time comparison is depicted in **Table 4** and the time window of mobile robots for the proposed methodology is shown in **Fig. 8**. Although the completion time of some destinations in our algorithm did not outperform the traditional method, the overall performance of our algorithm is better than the traditional one.

Table 4

Completion time comparison of the dense-distributed experiment

Destination	Completion Time (s)		
	Traditional	Proposed	Difference
(5, 18)	30	24	6
(5, 17)	48	22	26
(7, 18)	21	24	-3
(4, 16)	20	24	-4
Total	119	94	25

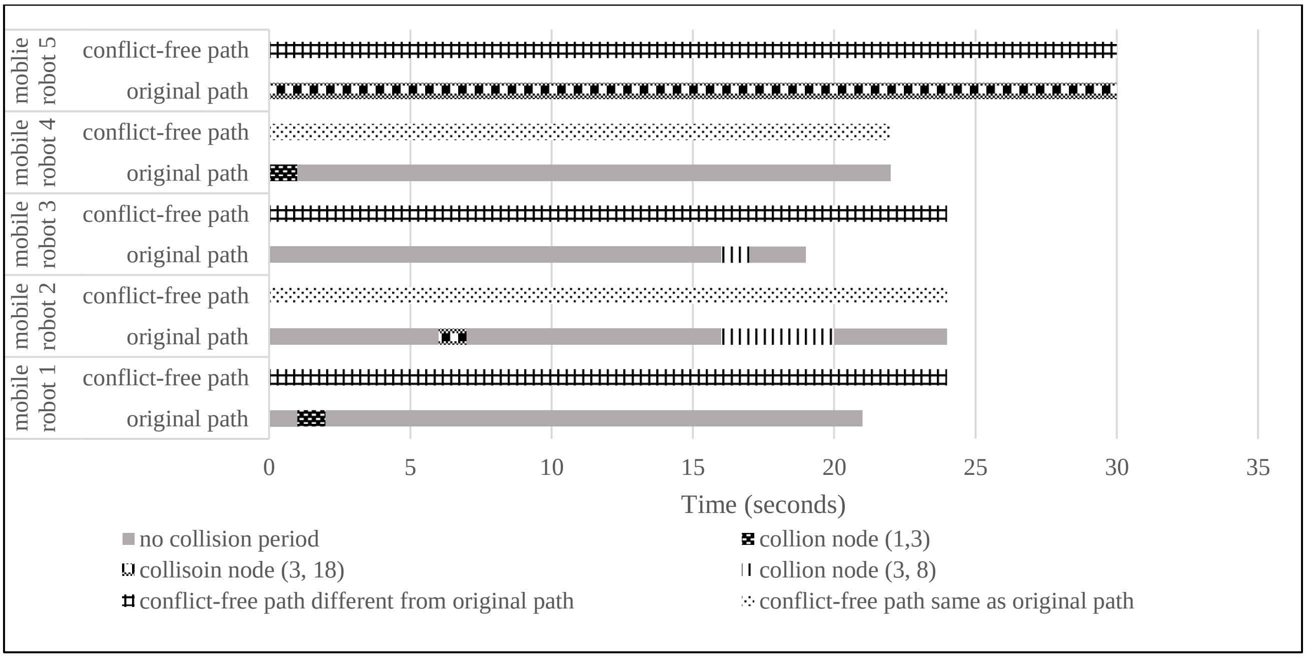

Fig. 8. Time window of mobile robots for the proposed methodology regarding the dense-distributed experiment

5.3. Computational analysis of algorithm performance

Twenty instances were provided to evaluate the algorithm performance. The computational results are shown in **Table 5**, illustrating the computation time and the task completion time of the traditional methodology and the proposed algorithm. The number of mobile robots was set as 2, 4, 6, 8 and 10 respectively, without any idle mobile robot on the shop floor. The initial coordinates of the mobile robots and the destination nodes were randomly generated. There was no duplicate node coordinate and the distance between any initial coordinate and any destination node was restricted to be larger than a predetermined constant. In most cases, the completion time of the proposed methodology was shorter than under the traditional approach. The computation time of the proposed methodology was slightly longer than the traditional method on average, as the proposed methodology adopted different strategies to solve the conflicts. However, the computation time difference could be ignored compared to the completion time difference. The proposed A* algorithm achieved 21% improvement in the required completion time on average compared to the A* algorithm.

Table 5

Computational performance with time comparison

# Mobile Robots	**Completion Time (s)**		**CPU (ms)**	
	A* algorithm	Improved A* algorithm	A* algorithm	Improved A* algorithm
2	**26**	**26**	**5**	11
2	**25**	28	**3**	5
2	37	**28**	8	**3**
2	**27**	**27**	2	**3**
4	62	**57**	9	**8**
4	58	**45**	12	**5**
4	76	**61**	**5**	11
4	86	**48**	10	**3**
6	110	**95**	14	**13**
6	89	**58**	**8**	12
6	127	**96**	**12**	17
6	100	**90**	**14**	19
8	158	**138**	**14**	17
8	132	**93**	22	**14**
8	140	**103**	**13**	**13**
8	157	**110**	**28**	29
10	208	**166**	**33**	46
10	148	**128**	**11**	20
10	185	**147**	11	**9**
10	188	**150**	**8**	13
Average	106.95	**84.7**	**12.1**	13.55

Bold value: best value among all algorithms

6. Discussion

Current robotic warehouses like Amazon and Alibaba also uses mobile robots to transport goods in a warehouse. The basic operation procedures are similar, but the proposed CPS-enabled warehouse system puts an emphasis on creating a digital counterpart of the physical world. As D'Andrea and Wurman [45] presented that wireless communication was a key issue in the Kiva system, since the navigation between vehicles must be transported through multiple access points. The coordination of multiple mobile robots was mainly via packet negotiation, so the stability of the network played a vital role in the design of robotic system. The proposed system has a tolerance on the network reliability, because the algorithm generates a final route with conflict-free characteristic. Moreover, according to the study of Li and Liu [46] and Alibaba warehouse video released from Business Insider [47], the collision detection algorithm could only stop mobile vehicles to avoid collisions. The proposed algorithm does not adopt the reactive approach by stopping mobile robots to resolve conflicts, but to adopt a proactive approach by finding out the alternative route to avoid collisions.

A CPS-enabled smart robotic warehouse system can timely supervise the real-time physical shop floor, as the virtual entity is synchronous with the physical world. Some researchers introduced collision avoidance algorithms considering kinetic characteristics of mobile robots. However, the kinetic features are constrained by mechanical aging problem. A valid avoidance module at the early stage might be not applicable to the late stage. A digital replica on the CPS warehouse system is capable of testing the real-life status of mobile robots and predicting their motion. The proposed collision avoidance module reduces computation time and alleviates the computational burdens on the system, so the control commands can be smoothly transmitted to mechanical assets which ensures the effective operations of the system.

7. Conclusion

The CPS-based smart robotic warehouse system is proposed to transform the traditional warehouse operations to smart logistics. The resource allocation problem of multiple mobile robots and the computation of conflict-free path planning are the main concern in this paper. An improved A^* algorithm was developed to find all possible paths from the source node to the destination node. Once the predetermined paths are confirmed, their time window can be compared to find different types of collision: stay-on, head-on and cross conflict. The collisions are solved by three different strategies: go-away, detour and wait-before-startup strategy.

The contribution of this article is outlined below. First, a system architecture of a CPS-enabled robotic warehouse is established. The CPS-enabled robotic warehouse is designed for third-party agencies for the implementation of outsourced elements of a firm's distribution center. The cloud-based system allows firms to review the logistics flow and information flow during the order fulfillment procedure. Second, an improved A* algorithm with greedy first-best-search is developed to achieve better solution quality in real-time computation for path planning. In order to cope with the need for a responsive design of a smart robotic warehouse and multiple robot control for order fulfillments, advancement in computational intelligence could help to achieve better convergence and real-time computation. Third, collision avoidance strategies in multiple mobile robots control to provide a solution with conflict and deadlock-free routing are proposed. Possible collisions are caught by a time window-based detection mechanism. Considering the idle mobile robot, the conflict is resolved by the go-away, detour or wait-before-startup strategy, depending on the total travel time.

Further work is required. First, the path planning may also consider the capability of workstations to maintain a balance of workforce capacities. Second, a more efficient conflict-free route will be addressed to further optimize the algorithm. The kinematic constraints of the mobile vehicles also need to be studied to make the algorithm more practical. Finally, heuristic conflict-free approaches can be beneficial for further performance improvement.

References

[1] C.K.M. Lee, S.Z. Zhang, K.K.H. Ng, Development of an industrial Internet of things suite for smart factory towards re-industrialization, Advances in Manufacturing, 5 (2017) 335-343.

[2] C.K.M. Lee, Y. Lv, K.K.H. Ng, W. Ho, K.L. Choy, Design and application of Internet of things-based warehouse management system for smart logistics, International Journal of Production Research, 56 (2018) 2753-2768.

[3] C.K.M. Lee, C. Yi, N. Kam Hung, Big Data Analytics for Predictive Maintenance Strategies, in: C. Hing Kai, S. Nachiappan, A. Muhammad Dan-Asabe (Eds.) Supply Chain Management in the Big Data Era, IGI Global, Hershey, PA, USA, 2017, pp. 50-74.

[4] K.K.H. Ng, M.H.M. Tang, C.K.M. Lee, Design and development of a performance evaluation system for the aircraft maintenance industry, 2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), IEEE, Singapore, Singapore, 2015, pp. 564-568.

[5] K.L. Keung, C.K.M. Lee, K.K.H. Ng, C.K. Yeung, Smart City Application and Analysis: Real-time Urban Drainage Monitoring by IoT Sensors: A Case Study of Hong Kong, 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2018, pp. 521-525.

[6] P. Zheng, T.-J. Lin, C.-H. Chen, X. Xu, A systematic design approach for service innovation of smart product-service systems, Journal of Cleaner Production, 201 (2018) 657-667.

[7] P. Zheng, H. wang, Z. Sang, R.Y. Zhong, Y. Liu, C. Liu, K. Mubarok, S. Yu, X. Xu, Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives, Frontiers of Mechanical Engineering, 13 (2018) 137-150.

[8] P. Zheng, C.-H. Chen, S. Shang, Towards an automatic engineering change management in smart product-service systems – A DSM-based learning approach, Advanced Engineering Informatics, 39 (2019) 203-213.

[9] P. Zheng, Y. Lin, C.-H. Chen, X. Xu, Smart, connected open architecture product: an IT-driven co-creation paradigm with lifecycle personalization concerns, International Journal of Production Research, 57 (2019) 2571-2584.

[10] P. Zheng, X. Xu, C.-H. Chen, A data-driven cyber-physical approach for personalised smart, connected product co-development in a cloud-based environment, Journal of Intelligent Manufacturing, (2018).

[11] D. Roy, S. Nigam, R. de Koster, I. Adan, J. Resing, Robot-storage zone assignment strategies in mobile fulfillment systems, Transportation Research Part E: Logistics and Transportation Review, 122 (2019) 119-142.

[12] B. Zou, X. Xu, Y. Gong, R. De Koster, Evaluating battery charging and swapping strategies in a robotic mobile fulfillment system, European Journal of Operational Research, 267 (2018) 733-753.

[13] B. Zou, Y. Gong, X. Xu, Z. Yuan, Assignment rules in robotic mobile fulfilment systems for online retailers, International Journal of Production Research, 55 (2017) 6175-6192.

[14] Z. Zhang, Q. Guo, J. Chen, P. Yuan, Collision-Free Route Planning for Multiple AGVs in an Automated Warehouse Based on Collision Classification, IEEE Access, 6 (2018) 26022-26035.

[15] Y. Zhang, Z. Zhu, J. Lv, CPS-Based Smart Control Model for Shopfloor Material Handling, IEEE Transactions on Industrial Informatics, 14 (2018) 1764-1775.

[16] P. Wurman, R. D'Andrea, M. Mountz, Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses, 2008.

[17] N. Boysen, R. de Koster, F. Weidinger, Warehousing in the e-commerce era: A survey, European Journal of Operational Research, 277 (2019) 396-411.

[18] M. Premm, S. Kirn, A Multiagent Systems Perspective on Industry 4.0 Supply Networks, in: J.P. Müller, W. Ketter, G. Kaminka, G. Wagner, N. Bulling (Eds.) Multiagent System Technologies, Springer International Publishing, Cham, 2015, pp. 101-118.

[19] J. Posada, C. Toro, I. Barandiaran, D. Oyarzun, D. Stricker, R.d. Amicis, E.B. Pinto, P. Eisert, J. Döllner, I. Vallarino, Visual Computing as a Key Enabling Technology for Industrie 4.0 and Industrial Internet, IEEE Computer Graphics and Applications, 35 (2015) 26-40.

[20] (!!! INVALID CITATION !!! [20-22]).

[21] A.E. Lee, The Past, Present and Future of Cyber-Physical Systems: A Focus on Models, Sensors, 15 (2015).

[22] L.D. Xu, L. Duan, Big data for cyber physical systems in industry 4.0: a survey, Enterprise Information Systems, 13 (2019) 148-169.

[23] J. Wan, Advances in Cyber-Physical Systems Research, Transactions on internet and information systems, 5 (2011).

[24] O. Purwin, R. D’Andrea, J.-W. Lee, Theory and implementation of path planning by negotiation for decentralized agents, Robotics and Autonomous Systems, 56 (2008) 422-436.

[25] K.K.H. Ng, C.K.M. Lee, S.Z. Zhang, K. Wu, W. Ho, A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion, Computers & Industrial Engineering, 109 (2017) 151-168.

[26] C. Tomlin, G.J. Pappas, S. Sastry, Conflict resolution for air traffic management: a study in multiagent hybrid systems, IEEE Transactions on Automatic Control, 43 (1998) 509-521.

[27] E. Gawrilow, E. Köhler, R.H. Möhring, B. Stenzel, Dynamic Routing of Automated Guided Vehicles in Real-time, in: H.-J. Krebs, W. Jäger (Eds.) Mathematics – Key Technology for the Future: Joint Projects Between Universities and Industry 2004–2007, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 165-177.

[28] S. Maza, P. Castagna, Conflict-free AGV routing in bi-directional network, ETFA 2001. 8th International Conference on Emerging Technologies and Factory Automation. Proceedings (Cat. No.01TH8597), 2001, pp. 761-764 vol.762.
[29] G. Demesure, M. Defoort, A. Bekrar, D. Trentesaux, M. Djemaï, Decentralized Motion Planning and Scheduling of AGVs in an FMS, IEEE Transactions on Industrial Informatics, 14 (2018) 1744-1752.
[30] I. Draganjac, D. Miklić, Z. Kovačić, G. Vasiljević, S. Bogdan, Decentralized Control of Multi-AGV Systems in Autonomous Warehousing Applications, IEEE Transactions on Automation Science and Engineering, 13 (2016) 1433-1447.
[31] D. Herrero-Perez, H. Martinez-Barbera, Modeling Distributed Transportation Systems Composed of Flexible Automated Guided Vehicles in Flexible Manufacturing Systems, IEEE Transactions on Industrial Informatics, 6 (2010) 166-180.
[32] T. Nishi, M. Ando, M. Konishi, Distributed route planning for multiple mobile robots using an augmented Lagrangian decomposition and coordination technique, IEEE Transactions on Robotics, 21 (2005) 1191-1200.
[33] H. Hu, K.K.H. Ng, Y. Qin, Robust Parallel Machine Scheduling Problem with Uncertainties and Sequence-Dependent Setup Time, Scientific Programming, 2016 (2016) 13.
[34] S.A. Reveliotis, Conflict resolution in AGV systems, IIE Transactions, 32 (2000) 647-659.
[35] N.N. Krishnamurthy, R. Batta, M.H. Karwan, Developing conflict-free routes for automated guided vehicles, Operations Research, 41 (1993) 1077-1090.
[36] K.K.H. Ng, C.K.M. Lee, F.T.S. Chan, Y. Lv, Review on meta-heuristics approaches for airside operation research, Applied Soft Computing, 66 (2018) 104-133.
[37] M. Saidi-Mehrabad, S. Dehnavi-Arani, F. Evazabadian, V. Mahmoodian, An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs, Computers & Industrial Engineering, 86 (2015) 2-13.
[38] N. Wu, M. Zhou, Shortest Routing of Bidirectional Automated Guided Vehicles Avoiding Deadlock and Blocking, IEEE/ASME Transactions on Mechatronics, 12 (2007) 63-72.
[39] K.C.T. Vivaldini, M. Becker, G.A. Caurin, Automatic routing of forklift robots in warehouse applications, Proceedings of the 20th International Congress of Mechanical Engineering, Gramado-RS, Brazil, Citeseer, 2009.
[40] K.C.T. Vivaldini, J. Galdames, T.B. Pasqual, R.M. Sobral, R.C. Araújo, M. Becker, G.A.d.P. Caurin, Automatic routing system for intelligent warehouses, IEEE International Conference on Robotics and Automation, Citeseer, 2010, pp. 1-6.
[41] N. Smolic-Rocak, S. Bogdan, Z. Kovacic, T. Petrovic, Time Windows Based Dynamic Routing in Multi-AGV Systems, IEEE Transactions on Automation Science and Engineering, 7 (2010) 151-155.
[42] P. Zheng, X. Xu, S. Yu, C. Liu, Personalized product configuration framework in an adaptable open architecture product platform, Journal of Manufacturing Systems, 43 (2017) 422-435.
[43] P. Zheng, Y. Lu, X. Xu, S.Q. Xie, A system framework for OKP product planning in a cloud-based design environment, Robotics and Computer-Integrated Manufacturing, 45 (2017) 73-85.
[44] P. Zheng, X. Xu, S.Q. Xie, A weighted interval rough number based method to determine relative importance ratings of customer requirements in QFD product planning, Journal of Intelligent Manufacturing, 30 (2019) 3-16.
[45] R. D'Andrea, P. Wurman, Future challenges of coordinating hundreds of autonomous vehicles in distribution facilities, 2008 IEEE International Conference on Technologies for Practical Robot Applications, 2008, pp. 80-83.
[46] J.-T. Li, H.-J. Liu, Design Optimization of Amazon Robotics, Automation, Control and Intelligent Systems, 4 (2016) 48-52.
[47] Business Insider, Inside Alibaba's smart warehouse staffed by robots.