remote sensing

Article

Dynamic Order Picking Method for Multi-UAV System in Intelligent Warehouse

Changwan Han [1,†], Hyeongjun Jeon [1,†], Junghyun Oh [1] and Heungjae Lee [2,*]

[1] Department of Robotics, Kwangwoon University, Seoul 01897, Republic of Korea
[2] Department of Electrical Engineering, Kwangwoon University, Seoul 01897, Republic of Korea
* Correspondence: hjlee@kw.ac.kr
† These authors contributed equally to this work.

Abstract: For the logistics environment, multi-UAV algorithms have been studied for the purpose of order picking in warehouses. However, modern order picking adopts static order picking methods that struggle to cope with increasing volumes of goods because the algorithms receive orders for a certain period of time and pick only those orders. In this paper, by using the modified interventionist method and dynamic path planning, we aim to assign orders received in real-time to multi-UAVs in the warehouse, and to determine the order picking sequence and path of each UAV. The halting and correcting strategy is proposed to assign orders to UAVs in consideration of the similarity between the UAV's picking list and the orders. A UAV starts picking orders by using the ant colony optimization algorithm for the orders initially assigned. For additional orders, the UAV modifies the picking sequence and UAV's path in real time by using the k-opt-based algorithm. We evaluated the proposed method by changing the parameters in a simulation of a general warehouse layout. The results show that the proposed method not only reduces completion time compared to the previous algorithm but also reduces UAV's travel distance and the collapsed time.

Keywords: dynamic order picking; unmanned aerial vehicle; multi-UAV system; intelligent warehouse

check for updates

Citation: Han, C.; Jeon, H.; Oh, J.; Lee, H. Dynamic Order Picking Method for Multi-UAV System in Intelligent Warehouse. *Remote Sens.* **2022**, *14*, 6106. https://doi.org/10.3390/rs14236106

Academic Editors: Francesco Nex and Davide Antonio Cucci

Received: 11 November 2022
Accepted: 28 November 2022
Published: 1 December 2022

1. Introduction

With the start of the Fourth Industrial Revolution, research and development of robots, AI, and IoT have been studied. Especially, through AI and Unmanned Aerial Vehicle (UAV), robots can perform complex tasks on behalf of humans [1,2]. As the core technologies of autonomous UAV such as mapping, path planning, and localization are studied, UAVs are being used in various fields such as security, logistics, surveillance, and quarantine [3–8]. In those fields, the need for logistics UAVs is increasing due to a lack of manpower, an aging population, and an increase in the volume of goods. The task that requires the automation of UAVs in a warehouse environment is order picking. Order picking is the task of receiving orders from customers with the Warehouse Management System (WMS) and retrieving items belonging to the order from the storage locations of the items [9]. Since the cost of order picking accounts for more than 60% of the entire warehouse operating costs [10], the importance of efficient processing for order picking is high. Creating a group of orders with similar items in a warehouse and efficiently determining the picking sequence in which the items are picked to minimize travel distance [11] is a general strategy for reducing overall travel time. For efficient operation of multi-UAVs for order picking, order batching, which groups orders into a group, and order sequencing, which determines the picking order of items in a batch, are important [12]. To achieve this purpose, the Multi Robot System is needed, in which numerous cooperative UAVs are handled in organized schemes [13].

To deal with the various types and large amounts of logistics increased by the introduction of e-commerce, the dynamic order picking method of processing orders received at a fixed time is being studied rather than the static order picking method [14–16]. Dynamic order picking processes arrived orders every moment, reducing the overall completion time and minimizing the travel time of the order picker. As a result, "dynamic order picking" responds swiftly to customer demands and improves customer service. In the previous method, there was a "milk-run" method [14,17], which modifies only the batch of orders without changing the original path of the picker, but the use of this fixed-length path formed a longer path for dealing with a single order. Instead, an interventionist method of modifying the path of a picker and assigning a new order was studied [16]. This method enables efficient order batching and order sequencing, but it could not be applied to multi-UAVs since it was an algorithm for a single picker.

Therefore, the primary goal of our study is an efficient dynamic order picking method to process increasing orders using the multi-UAVs with limited flight time. We propose the "Halting and Correcting Order batch" (HC) strategy and k-opt-based order sequencing method to efficiently pick orders received in real-time at the warehouse with multi-UAVs. The proposed method allocates continuously arriving orders to multi-UAVs with limited capacity. The proposed HC strategy is the extended algorithm [16] of the interventionist dynamic allocation scheme to assign orders to multi-UAVs, and modify each picking list for efficient logistics. In order to perform order sequencing of a frequently modified UAV's peaking list in real time, we interpret this order sequencing as the Dynamic Traveling Salesman Problem (DTSP), a NP-hard problem. The proposed Ant Colony Optimization (ACO) and k-opt-based order sequencing change the batch configuration by considering the UAV's current position and the location of orders that are not picked up before the order is added. The multi-UAVs efficiently move along the shortest path, not a fixed path like the milk-run method. The main contributions of the proposed method are as follows:

- The proposed method uses dynamic allocation based on interventionist order picking, and uses an efficient order picking method compared to previous logistics algorithms to achieve a smaller completion time. Through this, the warehouse operation's performance is improved.
- The proposed HC strategy uses similarity information between orders allocated to the UAV and has a smaller average travel distance than previous algorithms. Through this, it contributes to increasing the operating time of each UAV compared to the charging time of the UAV.
- By applying the k-opt-based DTSP solving algorithm to the logistics environment, the proposed method can quickly calculate the task sequence for the order. Through this, the sequence is changed in real time even when the order is frequently changed in the picking list of the UAV, and the optimal path is guaranteed.

The remainder of the paper is organized as follows. Section 2 introduces the previous methods for order picking and the definition of the DTSP. Section 3 defines the assumption of a dynamic order picking situation in a logistics warehouse and the parameters of a warehouse and a robot. Section 4 describes the proposed HC strategy based on interventionist dynamic allocation for multi-UAVs and the dynamic TSP based on k-opt on how to apply the logistics environment (problem definition). Section 5 compares the proposed method with the previous strategy under various constraints. In Section 6, we discuss results and future research.

2. Related Works

2.1. Dynamic Order Picking

With the introduction of e-commerce for business-to-consumer (B2C) [18], consumers can individually choose their destination for orders, change items, cancel orders, etc. [14]. With this situation, logistics management and order picking have become more important and complicated as orders increase. It is assumed that order picking covered in our paper is performed on a general picker-to-parts system [19]. In this system, pickers move along the aisles in the warehouse to pick up the requested items. However, previous methods had limitations in terms of efficient work processing due to waiting time. Therefore, dynamic order picking is being studied to deal with the explosive increase in order [14].

Static order picking is a method in which an order is received at a specific time and picked by the order picker during the pick cycle. Orders arriving during this pick cycle are not processed for a certain period of time, but will be processed after the previous order has been picked [20]. Delays for orders received during this pick cycle will cause delivery timings to be delayed. On the other hand, if a new order arrives during the pick cycle, dynamic order picking adds the order to the picking list and modifies the path of the order picker [14]. By processing orders received at every moment, dynamic order picking can reduce the overall completion time and travel time during the pick cycle. As a result, the dynamic order picking satisfies the need for a fast response to consumers and improves the quality of customer service [16].

2.2. Method of Dynamic Order Picking

Dynamic order picking is typically constructed based on two method types: dynamic order batching and dynamic pick lists. Compared to the previous methods for static environments, it is difficult to compare the performance of dynamic methods because the methods using dynamic order picking have different configurations and purposes in warehouses. Comparison of method type and assumptions for the following dynamic-order picking algorithm is summarized in Table 1.

Table 1. Comparison of method type and assumption of the proposed method for previous methods.

	Nieuwenhuyse et al. (2009) [21]	**Henn et al. (2012) [22]**	**Rubrico et al. (2011) [15]**	**Gong et al. (2008) [14]**	**Giannikas et al. (2017) [16]**	**Proposed**
Method type	Dynamic order batch	Dynamic order batch	Dynamic pick lists	Dynamic pick lists	Dynamic pick lists	Dynamic pick lists
Picker type	Multiple	Single	Multiple	Multiple	Single	Multiple
Item size of order	Variable	Variable	One	Variable	Variable	Variable
Allow to change picking list	No	No	Yes	Yes	Yes	Yes
Allow to change picker's route	No	No	Yes	No	Yes	Yes

2.2.1. Dynamic Order Batching

Most of the previous works allow to change batch in the order batch step for grouping similar orders. These methods took into account the probabilistic characteristics of the customer's order to determine the size of the batch or the batch's time window. This method is largely classified into two types: fixed time window batching (FTWB) and variable time window batching (VTWB). The former is a strategy of making and picking a batch from orders received within a fixed time. The latter waits to make a single batch until a predetermined number of orders arrive [21] or delays the picking start time of the new batch until the predetermined conditions are met [22].

2.2.2. Dynamic Pick Lists

A pick list is a list of items that pickers must pick, and dynamic pick lists are a method of modifying this pick list for UAVs that are already working [15]. This online rescheduling strategy allows UAVs to modify the picking schedule to consolidate new orders with the remaining items. Rubrico [15] assumes that a new order consists of one item, and when a new order comes in while processing an remained order, it reorganizes the batch by combining the arrived order with orders in the pick list. The steepest descent insertion method and the multistage rescheduling strategy were used. On the other hand, Gong [14] applies a polling system that considers the location of each item as a queue. If item location of new order is picker's current or downstream location, Gong allows the picker to add the order to the pick list and uses a fixed path by the milk-run method. As a result, the completion time is reduced compared to the previous method, but the use of the fixed path generates an inefficient path.

2.3. Interventionist Order Picking Strategy

Interventionist order picking [16] is a dynamic order picking method that corresponds to the dynamic pick list strategy. This method suggests that when a new order arrives, it updates the pick list of pickers that have already departed or makes a new batch with the orders that have not yet been picked. In this method, three decisions have been proposed on order allocation.

1. Interventionist—accept all (IAA): This method waits in the warehouse's depot until new order arrives. If N orders arrive, they are assigned to a picker to start picking. The picker in operation is assigned a new order until the capacity conditions allow. This method increases the processing time of orders that were already on the pick list. Therefore, both the task completion time and the travel distance increase.
2. Interventionist—order completion time (IOCT): This method, like the IAA, waits in the warehouse's depot until new order arrives. If N orders arrive, they are assigned to a picker to start picking. It is a method that adds a new order if the capacity condition is satisfied and the completion time increased by additional orders does not exceed a certain threshold. In this method, in order to obtain the completion time, it is necessary to find the optimal travel path in advance, and this computational cost increases the computation of the algorithm.
3. Interventionist—re-batching (IRB): This method also waits for N orders. When a new order comes in, a new batch is created with either (1) orders that have been assigned to the picker but have not been picked yet or (2) orders that have not been assigned yet. Although it is possible in various ways, it requires a lot of computation time even for a single picker.

Although this strategy succeeded in lowering the overall order picking completion time, the travel distance of the algorithm in the warehouse increased. Furthermore, there is a disadvantage that it cannot solve the order assignment for general multiple pickers.

2.4. DTSP

When a new order is assigned to a picker with dynamic order picking, a new path must be created that passes through the location of the order added to the existing path. The problem of creating a new path can be defined as the DTSP [23]. In the case of general TSP, if a node is a location to be passed through, it is a problem to visit each node exactly once, considering the list of nodes and the distance between nodes, and generate the shortest recursive path. However, in the case of the DTSP, whenever a new node is added, removed, or the location of a node is changed, there is a problem of modifying the old path to a new optimal path.

For the DTSP problem, after a new node is added to a batch, which is a set of nodes, a path can be created using the TSP solution algorithm once again. However, recalculating a new optimal path for a batch changed by previous algorithms takes a lot of time. In order to reduce the high time cost due to this recalculation, using the information on the optimal path before the change is required. However, the performance varies with the initial parameters for the new node, and it can converge to the local minima that is dominant in the optimal path before the change. Recently, the following biologically-inspired DTSP solution algorithms have been studied to solve this problem; ACO, Particle Swarm Optimization (PSO), and Genetic Algorithm (GA) [24,25].

The ACO is an algorithm that uses the volatility of an ant's pheromone to generate an optimal path through stochastic path selection. However, when applied to a dynamic environment, the pheromone information is confused due to newly added and removed nodes. To address this problem, representative research uses a strategy that reflects the influence of changed nodes on pheromones that existed in previous paths [26]. However, if the information on the existing path is dominant, the newly created path can also be greatly affected, potentially causing problems. To solve this problem, a method that has a significant effect on pheromone information by adding a new ant rather than an existing one was proposed [27,28].

The PSO is an algorithm in which multiple optimizers exchange information with each other and perform optimization at the same time. Each optimizer means a solution to a problem, and after evaluating the information of each optimizer, the best path is found by passing the best information to another optimizer. In order to apply the PSO in a dynamic environment, there is a limitation to the fixed parameters of the algorithm. For this, it can converge to the path including the changed node by stochastically changing the parameters [24]. As another method, solving the DTSP by combining the pheromone information of the ACO with the PSO has been proposed [29].

2.5. Multi-UAV Task Allocation

In dynamic order picking, the execution method of multiple UAVs is related to the multi-UAV dynamic task allocation [30]. In this task allocation, the "dynamic" of the task given to the UAV refers to the delayed arrival of the task, the addition of the new UAV, and the change in the constraints of the UAV's mission performance by an external agent [31,32]. When new tasks arrive, there are previously scheduled tasks that have not yet been performed. The UAV is assigned a task suitable for the current environmental conditions and replans the travel path as it proceeds with the previously assigned work. Considerations between the performance of multi-UAV dynamic task allocation are resultant average task allocation time, task completion time, and travel distance of each UAV [30].

The travel distance of each UAV is associated with the limited resources (e.g., battery) of the UAV. Since UAVs have long charging times and limited operating times, efforts have been made to minimize the energy consumption of UAVs in this environment from a realistic point of view [33,34]. Recently, the works have been studied that not only process efficiently the target task, but also minimize the travel distance of the UAV [34]. Therefore, in this paper, we check the travel distance of each UAV to prove that the proposed algorithm is suitable for multi-UAV systems.

3. Problem Definition

The parameters for the proposed logistics environment are composed of the parameters of the optimization model mainly used in the static order picking and the multi-UAV. This parameter is used to analyze the performance of the proposed method under various conditions. In these parameters, c_{min} used in the interventionist method is the amount of orders allocated to generate the initial path when the picker starts from the depot and performs the picking task. As the c_{min} increases, the number of orders that the UAV can be allocated decreases.

Parameters of warehouse:

i: Index of an item
P: Number of items present in the logistics warehouse
p_i: Location of item i
k: Index of orders
O_k: An index set of items belonging to the kth order
n_{init}: Number of initial customer orders
n_{last}: Total orders, last customer orders
s_k: The number of items that the kth order has
S: Maximum number of items a customer's order can have
γ: Order growth rate
n_γ: Number of orders arriving per cycle
n_d: Number of depots
c_{min}: Minimum number of orders for order batch creation

Parameters of Multi-UAV:

n_r: The number of UAVs
u: The index of a UAV
l_u: Current location of UAV u
v_{UAV}: The speed of the UAV
c_u: Load capacity of orders currently held by the UAV
c_{max}: Maximum order load on the UAV
PL_u: Current picking list of UAV u
NPL_u: List of items that have not yet passed through within the current pickup list of UAV u

Additional assumptions related to the warehouse and multi-UAVs are as follows:

1. Each item is independent of other items and orders.
2. The speed of the UAV is constant.
3. The UAV has the ability to pick each item by itself.
4. The aisle is wide enough to prevent collisions between UAVs.
5. UAVs are equally assigned to each depot.
6. Each UAV starts from the depot, picks up the item on the picking list, and then moves to the starting depot.
7. It is assumed that the picking time of the UAV is constant. Therefore, no matter which algorithm is applied, the entire pickup time is the same, so the pickup time is ignored in the entire execution time of the algorithm.
8. UAVs are assigned orders and order sequences from the WMS. Each UAV calculates an optimal path along this sequence and then autonomously flies between shelves.

In this paper, we adopt the traditional layout [35] of the picker-to-part warehouse. We assume the horizontal configuration of multiple blocks as the layout. The warehouse environment is expressed in the form of a grid map, such as Figure 1. The UAV moves in four directions on this grid-shaped map, circling each shelf.

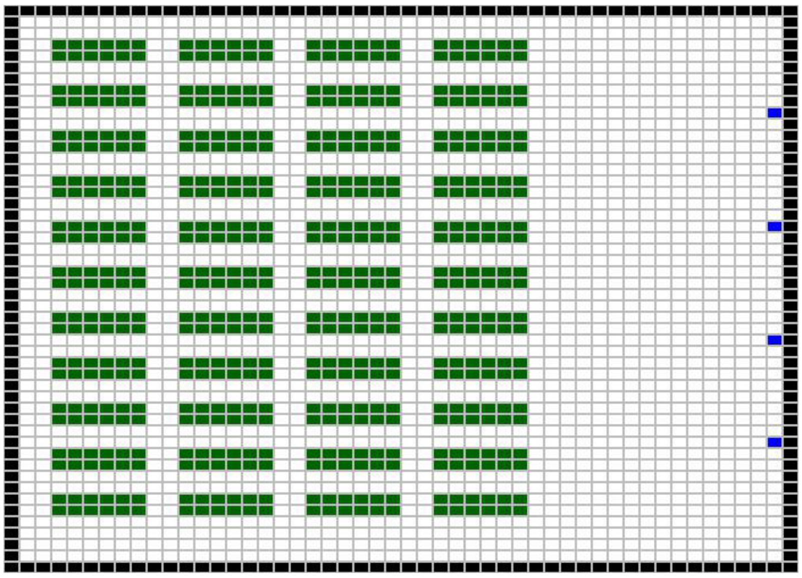

Figure 1. Warehouse layout. Green blocks mean shelves on which items are contained in a warehouse. The blue blocks mean depots in which the robot starts driving and returns. The white blocks are free areas where the robot can move. The black blocks mean stuck areas where the robot cannot move.

4. Proposed Method

4.1. Multi-UAV Task Allocation and Sequencing Strategy for Dynamic Order Picking

The proposed method is a dynamic order picking method for a logistics system using multi-UAVs. It determines the allocation of an order based on the interventionist order picking strategy [16] whenever a new order arrives, such as in Figure 2. Since the proposed method uses multi-UAVs, unlike the previous method, it determines which UAV to assign to a new order. After a new order is assigned, the visit sequence of the orders on the UAV's pick list must be recalculated to minimize the travel distance. However, since it is a multi-UAV environment, this recalculation of the UAV's visit order causes congestion in the logistics system and delays due to high computational costs. From this point of view, we apply the DTSP algorithms that modify existing paths in the form of combinatorial optimization rather than path recalculation for additional orders. Through this, the proposed method generates a new path with new orders at a low computational cost.

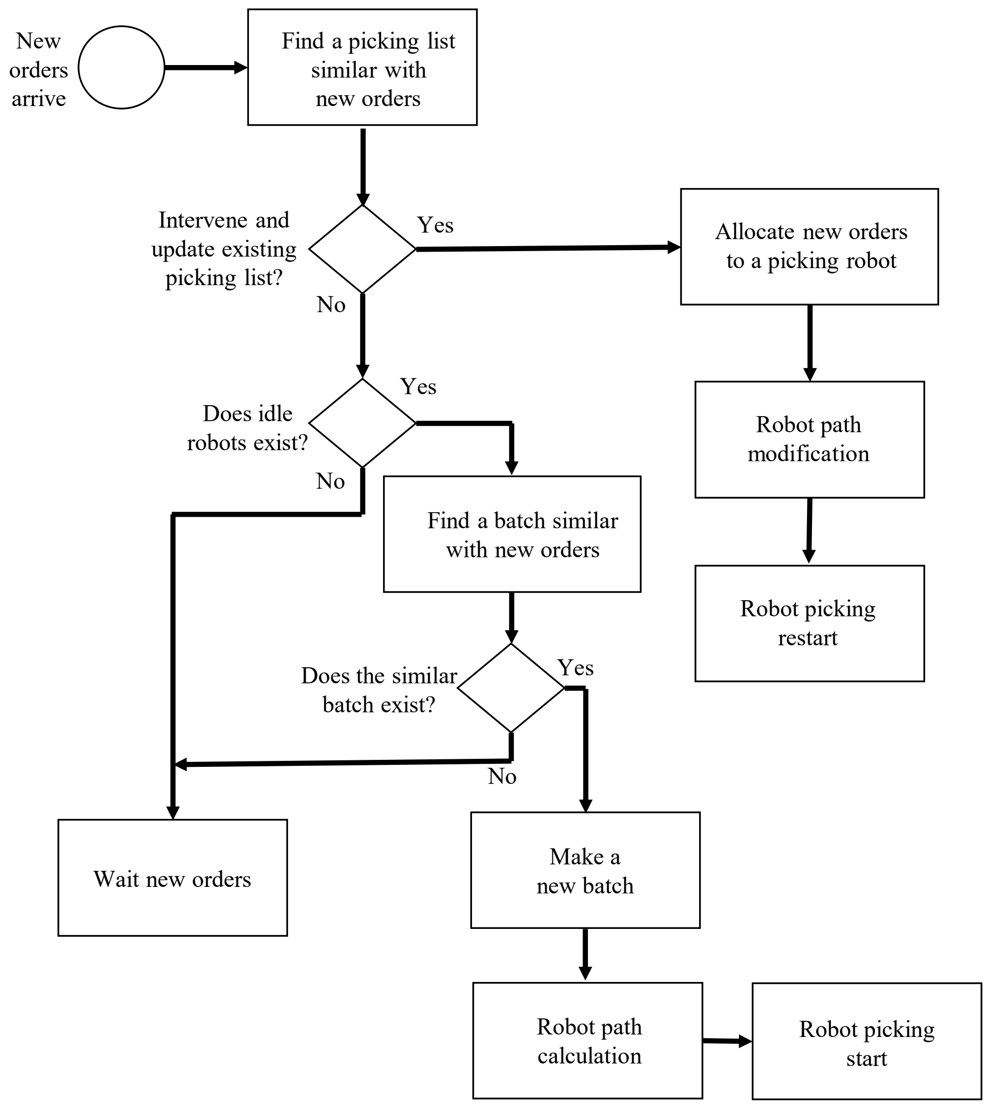
New orders arrive
Find a picking list similar with new orders
Intervene and update existing picking list?
Yes
No
Allocate new orders to a picking robot
Robot path modification
Robot picking restart
Does idle robots exist?
Yes
No
Find a batch similar with new orders
Does the similar batch exist?
Yes
No
Wait new orders
Make a new batch
Robot path calculation
Robot picking start

Figure 2. Proposed method flowchart based on similarities between new order and the pickup list. The proposed strategy either assigns a new order to the UAV or combines a new order with not allocated orders to make a new batch.

4.2. Halting and Correcting Strategy

The proposed task allocation method is the HC strategy, an extension of the existing interventionist order picking strategy to multi-UAVs. The proposed method uses the IOCT strategy [16] included in the interventionist order picking strategy to determine whether to add a new order to the UAV's picking list. The proposed strategy uses the order batch in Figure 3 and an additional step to deal with orders that are continuously waiting.

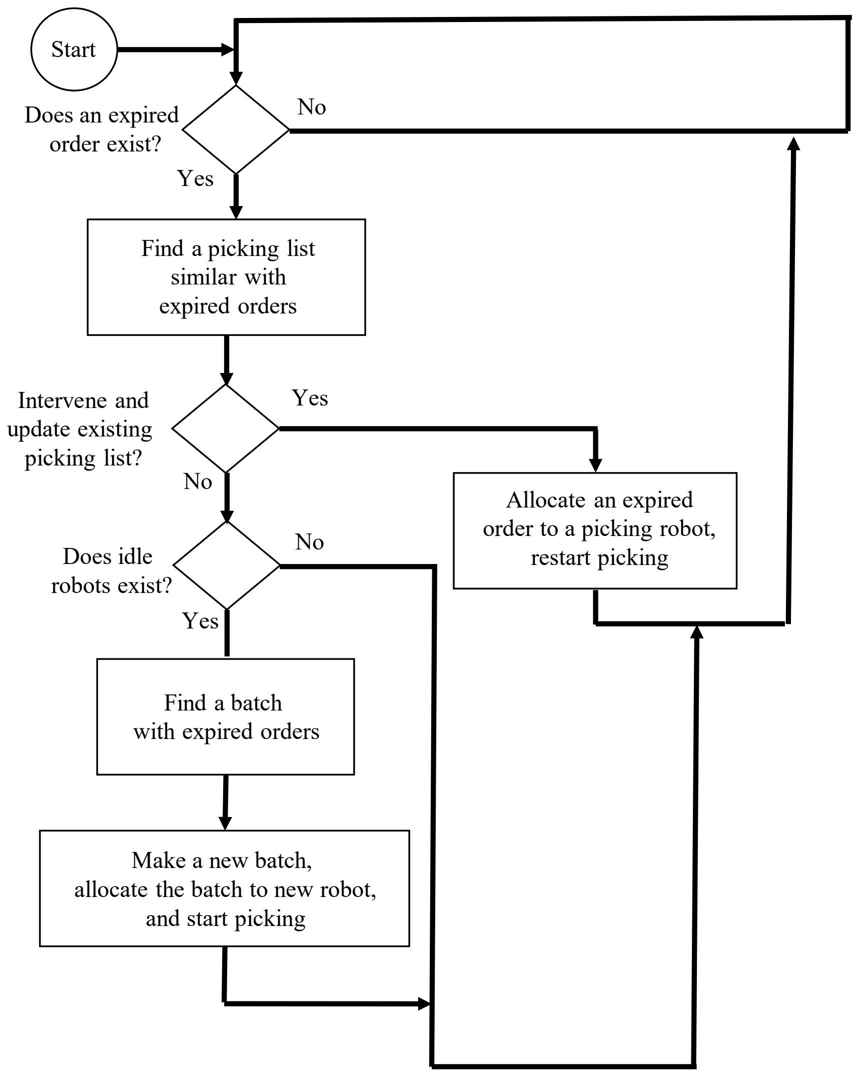
Start
Does an expired order exist?
No
Yes
Find a picking list similar with expired orders
Intervene and update existing picking list?
Yes
No
Allocate an expired order to a picking robot, restart picking
Does idle robots exist?
No
Yes
Find a batch with expired orders
Make a new batch, allocate the batch to new robot, and start picking

Figure 3. Waiting orders processing flowchart. Orders that have been waiting for a long time until the expired time must be processed to meet the fast completion time of the warehouse.

4.2.1. Finding a Picking List Similar with New Orders

When a new order arrives, the proposed method checks whether the new order has an item similar to one on the picking lists of the UAVs currently picking. Before picking starts, first calculate the path distance between the item and the location of the shelf where the item is in the warehouse. In order for the path distance to reflect the optimal distance for the actual UAV to travel, the shortest path distance calculated through the A* algorithm [36] is used in the grid map.

To find a robot u' with the most similar pick list to the new order k', the current position l_u of each UAV u in operation and a list of items not currently passed through in the UAV's picking list NPL_u are required. Construct $Z_u = \{u\} \bigcup NPL_u$, which is the union of u and NPL_u. Then, calculate the average distance between each item i' in $O_{k'}$, which is the item set of order k' and the elements j of Z_u with the minimum distance by (1).

$$S_{k',Z_u} = \frac{1}{|O_{k'}|} \Sigma_{i' \in O_{k'}} \min_{j \in Z_u} d(i', j) \tag{1}$$

The smaller the S_{k',Z_u}, the higher the positional similarity between the existing UAV's picking list and the new order. Taking this into account, the UAV u' with the smallest distance value among the S_{k',Z_u} for all UAVs in (2) is chosen as the UAV to be intervened.

However, if this distance is above the threshold τ_i, the proposed method does not add an order to the picking list of the UAV u'. This is done in order not to increase the completion time of the picking cycle of the existing UAV too much. Furthermore, when a new order is added, only UAVs that do not exceed the UAV's maximum payload c_{max} become intervened candidate UAVs. If there are no candidate UAVs, the proposed method does not modify the existing pick list.

$$u' = argmin_u S_{k',Z_u} \tag{2}$$

4.2.2. Constructing the Batch Similar to New Orders

If a new order is not assigned to a picking UAV and there are no idle UAVs, the proposed method waits for a new order. Afterwards, when a new order arrives, if there is an idle UAV and the similarity between the new order and the batch is high, a new batch is made and started. If several UAVs are in the idle state, the UAV with the smallest cumulative travel distance is selected.

For efficient batch generation, batches between waiting orders are constructed with high similarity using the seed algorithm [37]. The seed order s of the seed algorithm is changed from the first waiting order to the new order. This algorithm finds the top c_{min} orders with high similarity to the seed order. In this case, the similarity is calculated using the average of the minimum distances between items of seed order O_s and items of the waiting order O_k in (3).

$$C_{s,k} = \frac{1}{|O_s|} \Sigma_{i' \in O_s} \min_{j \in O_k} d(i', j) \tag{3}$$

If there are top c_{min} orders with high similarity above the threshold τ_n, a new batch is created and the batch is assigned to the UAV. If there are no c_{min} similar orders above the threshold, new orders are delayed.

4.2.3. Processing Waiting Orders with Expired Time

In the waiting orders, the waiting time is recorded from the initial arrival time. Waiting for orders is necessary for efficient order picking, but long waiting delays completion time for the entire warehouse. Therefore, we process the corresponding order if the waiting time exceeds a fixed expired time T_e such as Figure 3. This order is called an expired order.

If additional orders cannot be assigned due to the capacity limit of the picking UAVs, the expired order is used as a seed order for other waiting orders. Through the seed algorithm used in Section 4.2.2, the batch with the maximum size of c_{min} is configured and assigned to the idle UAV. At this time, the restrictions on the similarity threshold τ_n and the minimum order quantity c_{min} are released, and if there is no idle UAV, the order is delayed until the idle UAV exists.

If picking UAVs have empty capacity to pick orders, the similarity between expired orders with picking list of UAVs is once again the same as the comparison scheme between UAVs and seed orders used in Section 4.2.1, but with a threshold higher than τ_i. If the average distance S_{k',Z_u} satisfies the constraint for that threshold, it assigns a order to the UAV. Otherwise, it creates a batch without restrictions on the waiting order described above. This strategy is applied as a response to reduce the total completion time.

4.3. Order Sequencing

The UAV in the warehouse uses different path generation algorithms depending on when the proposed method allocates the order. When each UAV is assigned a batch for the first time, it determines the travel path based on the variant ACO algorithm MAX-MIN Ant System (MMAS) [38]. On the other hand, when the UAV is in operation, if a new order is added to the pick list, the path to the remaining items and items in the new order must be recalculated. In this dynamic case, if a new path is calculated based on the ACO algorithm, there is a possibility that the UAV may stop for a long time between path creations or may not return an appropriate path due to the high computation cost of the TSP problem. To prevent this, we interpret this order addition and path creation process as the DTSP and modify the current path by applying a k-opt-based algorithm. Through this, the proposed method creates a new picking path with a low computational cost.

4.3.1. Normal Order Sequencing

When UAVs waiting at the depot are assigned a new batch, they determine the picking order of the items based on the location of the items that make up the batch. The positions of the items are set as nodes in the TSP. Since determining the picking order for these fixed items is the same as the definition of the TSP algorithm, we use MMAS, a prominent TSP solving algorithm. The MMAS applied in this paper starts m ants from the node corresponding to the depot, not the random location. The pheromone τ_{ij} is maintained in the trail between i -th node n_i and j -th node n_j in all N nodes. Ants stochastically select the next node based on this pheromone information. The probability that the ant k at node n_i selects node n_j as the next node is affected by the distance information between the pheromone and the node as shown in (4):

$$p_{ij}^k = \begin{cases} \dfrac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{l \in N_i^k} \tau_{il}^\alpha \eta_{il}^\beta} & \text{if } j \in N_i^k, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

d_{ij} is the distance between n_i, n_j and is reflected in this probability in the form of $\eta_{ij} = 1/d_{ij}$. The parameters α and β reflect the importance of pheromones and distance information in selecting nodes as weights between τ_{ij} and η_{ij}. The node set N_i^k is an unvisited node among adjacent nodes of ant k. Nodes that do not belong to the set have been visited once, so the probability of the visit is set to zero and no more visits. Based on the above probabilities, each ant generates a path R_m. In the next process, the pheromone update process, elite ants are selected $(f(R_m))$ to reduce the convergence time of the optimal path. The elite ant is selected as the ant that generated the shortest path at that time.

At the end of the path generation process, the pheromone information needs to be updated. First, the pheromone is reduced using the evaporation rate ρ, and it is updated with the sum of the pheromone left by the m ant creating a path and the pheromone of the elite ant. The pheromone of node n_i, n_j is updated via (5):

$$\tau_{ij} \leftarrow [(1-\rho)\tau_{ij} + \Delta\tau_{ij}^{best}]_{\tau_{min}}^{\tau_{max}} \tag{5}$$

τ_{min}, τ_{max} is the lower limit and upper limit of the pheromone value, respectively. Maintaining the pheromone above a certain level keeps the possibility of creating various paths. In the case of $\Delta\tau_{ij}^{best}$, it is expressed as (6):

$$\Delta\tau_{ij}^{best} = \begin{cases} 1/L_{best} & \text{if } (i,j) \text{ belongs to the best tour,} \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

L_{best} means the distance of the best path in the current cycle, and the pheromone is updated through the above equation and a new path is checked in the next cycle. If the elite ant is not updated for a certain period, the current path is judged to be the optimal path and the calculation is terminated early to reduce the calculation cost.

4.3.2. Dynamic Order Sequencing

Dynamic order sequencing is necessary when assigning a new order to a UAV that is already working on it. In the case of the previous TSP solution algorithm, the algorithm was solved using the location of the new order and orders remaining on the picking list to generate the optimal path. Because original path information is not used, it takes a lot of time to generate results. To solve this problem, the closer the initial path is to the optimal solution, the higher performance k-opt is used to generate new paths, which reduces computational costs.

The main idea is that the k-opt algorithm can find the shortest distance and improve the path by rearranging and releasing the twisted path as one of the local optimization algorithm techniques. It is an algorithm that improves the path through iterative exchange by comparing the process of swapping any k edges among the edges of a given path with all possible valid combinations. If the algorithm is used after creating the initial path, it can be used in the warehouse environment to solve dynamic order sequencing with low computation time. The proposed method proceeds as shown in Figure 4. A circle represents a node, and a dotted line represents an edge. Figure 4a is the result of static order sequencing. If the Figure 4a is changed to Figure 4b, a new order (blue circle) arrives to UAV and one edge closest to new order (red dotted line) is judged as a twisted edge. Then, remove twisted edge (red dotted line) and create two edges, including a blue circle. Finally, a path including a new node is generated as shown in Figure 4c. Thereafter, different edges are randomly selected and the path is compared using the above method. After that, the same method is applied to the positions of all added nodes and an optimal path including all nodes is generated.

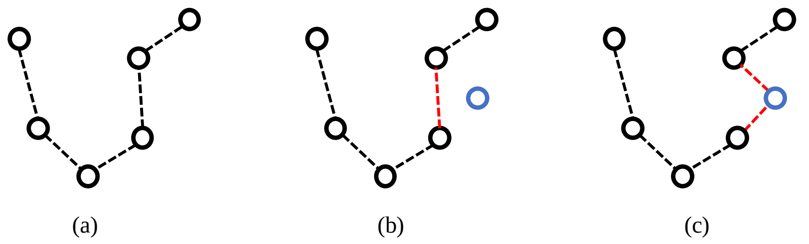

Figure 4. This is the progress of the proposed k-opt algorithm. The circle indicates the location of the ordered item, and the dotted line indicates the edge from one item to another. The blue circle is the location of the newly added order, and the red dotted line is the edge twisted by the newly added order. (**a**) Optimal path generated by using the given order position. (**b**) The new order is assigned. (**c**) The result of proposed k-opt.

The UAV determines the visit sequence of each item in the warehouse using two different algorithms according to the current state of PL_u. When an UAV u is assigned an order for the first time, it computes the path R via the static TSP solution of PL_u and Section 4.3.1. If the UAV in operation does not exceed c_{max}, based on Section 4.2, a new order of V is allocated. Dynamic order sequencing is performed using NPL_u including the newly allocated order. When creating a path, the UAV's current position l_u is used as the starting point, and the depot is used as the ending point. Remove the edge passed through in the R and create a new path as shown in Figure 4 using one of the newly added orders. Proceed in the same manner for all edges, and find the path with the shortest length among them. Repeat the above until all new orders are included in the path. The shortest path created with all the added orders is defined as R_{best}, and an efficient logistics system is established through the path. The overall framework can be checked through Algorithm 1. I, r, and m are the maximum number of iterations, the number of nodes, and the number of ants, respectively.

The proposed method has an advantage in terms of time complexity instead of having a optional solution in the length of the path. Calculating the time complexity, the method of Section 4.3.1 is $O(I \times m^2 \times r)$, and the proposed method is $O(r^2)$. In general, because the number of agents such as ant in the ACO is greater than the number of nodes, algorithms for normal order sequencing have higher time complexity than proposed methods. The results can be confirmed through next section.

Algorithm 1 Proposed method.

```
Input: path $R$, allocated tasks $V$
Output: Best path $R_{best}$
if $R$ is empty then
    Check the node $N = \{n_0, n_1, n_2, ..., n_r\}$
    for $i = 1$ to $I$ do
        if $i = 1$ then
            Generate $m$ random ants
        else
            Move the ants by evaporate map—(4)
        end if

        for $k = 1$ to $m$ do
            ant $k$, make a path $R_k$
            Find the best path $R_{best} = f(R_k)$
            Check the elite ant $k_{elite}$
        end for

        Update the pheromone and elite pheromone— (5)
        if the $R_{best}$ has not changed in five iterations then
            return $R_{best}$
        end if
    end for
else if $R$ exists, get $V$ then
    for $v$ in $V$ do
        Find the nearest edge in $R_{best}$
        Using proposed k-opt to make new $R_{best}$
    end for
    return $R_{best}$
end if
```

5. Experiment

The performance of the suggested algorithm is confirmed in this experiment by comparing it to the previous online logistics algorithm. The suggested technique's performance is verified by comparing its order batch and order sequencing with the general order batch method and other routing methods, respectively. The UAV's travel distance, algorithm calculation time, and warehouse completion time for each algorithm are used to compare the characteristics and performance of the algorithm. The travel distance of the UAVs indicates how evenly each algorithm distributes orders to each UAV. The algorithm calculation time represents the time consumption of decision making in an environment using mobile UAVs. Warehouse completion time is a criterion that indicates whether the proposed method speeds up order processing in a logistics environment. To confirm that the proposed order sequencing algorithm is suitable for an environment using multi-UAVs, the time taken to create a path and the length of the generated path are compared by changing the number of UAVs.

5.1. Experiment Design

The experimental environment was implemented by simulating the logistics environment of Figure 5. In the simulation, new orders arrive in a repeated period. The items of the incoming order are retrieved by UAVs based on the picker-to-part system. The UAV generates an optimal path based on the A* algorithm between shelves of items and starts from the assigned depot. In this experiment, the items are extracted from the uniform distribution of the total number of items on the assumption that they are independent. We assign the shelf a 2 $\times$ 3 space on the grid. This reflects the characteristics of the warehouse in which the items are stored. Furthermore, in the simulation environment, the algorithm ignores the time it takes to pick an item after it reaches the shelf. This reflects the assumption that picking times are the same for a given order. Therefore, as soon as each UAV reaches a shelf, it estimates its next path and moves to the next shelf.

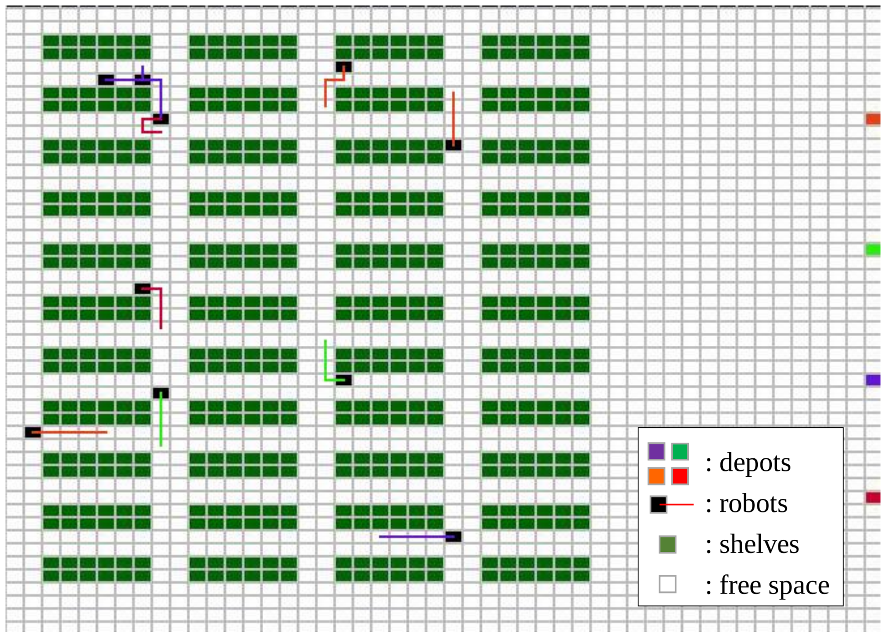

Figure 5. Logistics environment simulation in grid form. In the multiple block structure, the picking UAV is represented in a black box. The lines are robot paths with same color of the starting depot.

Algorithms compared with the proposed method are First-Come-First-Served (FCFS), which is an order batch method that is mainly used practically in the online method, and order sequencing based on the ACO. When a new order arrives, the FCFS strategy checks that more than c_{min} are waiting. Earlier works use a different assumption about the logistics environment. This assumption makes it difficult to compare our strategy with the earlier works. To evaluate the performance of our strategy, we choose FCFS strategy as baseline algorithm, which was used as a comparison algorithm in many studies [39,40]. When more orders arrive, the FCFS strategy allocates them to the UAV that is currently idle. The assigned UAV performs order sequencing and picking. If there is no idle UAV, the system assigns the orders by considering only the capacity limit of the currently picking UAV. The assigned UAV appends the order to the UAV's picking list and performs order sequencing for orders that are not picked from the picking list. In this way, order batching is performed by prioritizing the arrival time of the order using the IAA method [16] for the single picker. Both proposed k-opt-based order sequence methods and comparison methods use the ACO algorithm to determine the order sequence for an idle UAV.

We confirm performance differences with other algorithms by changing the number of arrived orders (n_γ = 2, 4, 6, 8, 10) per γ as a parameter for comparing the HC strategy performance under various conditions. γ is set to 1 second. Changing the parameters in the restriction method's parameters also discusses the effect of expired time T_e which determines the existing expired order. Between these comparative experiments, the number of shelves P is set to 88. It was assumed that a total of 200 orders had arrived online between experiments. The number of UAVs n_r is 10. In the simulation, It is assumed that the UAV's speed v_{UAV} is constant at 10 grid/s. The maximum capacity c_{max} is 8. The minimum number of orders c_{min} by the batch creation constraint is 4.

To compare the order sequencing performance of the proposed k-opt based algorithm with the ACO-based algorithm, we change the number of UAVs n_r from 1 to 10 and analyze the performance of the two algorithms. n_γ is set to 8, and the other parameters are identical as in the above experiment. The batch strategy uses the HC strategy with identical parameters. As evaluation criteria, the completion time per algorithm execution and the average moving distance of the UAV are used. The completion time per algorithm execution represents the average algorithm completion time for the strategy. By comparing the completion time per algorithm execution, the computational efficiency of the proposed algorithm is checked. Through the resultant travel distance, it is evaluated whether the proposed algorithm is suitable for the logistics environment.

We implemented the simulation in Python with multiprocessing. The simulation consists of four processes. First process generates orders periodically. According to the batch and sequence method, second process allocates orders to UAVs. To make UAVs pick the orders in picking list, third process calculates flight routes of UAVs and moves UAVs along the routes. Using PyQT package, fourth process visualizes the environments of warehouse and UAVs. In experiments, the PC is equipped with Intel Core i9-11900 CPU, 16 GB RAM. The operating system is Windows.

5.2. Experiment Result

The comparison groups of the experiment on order picking were: (1) FCFS-based order batch and ACO-based order sequencing (FCFS + ACO), (2) FCFS-based order batch and k-opt-based order sequencing (FCFS + k-opt), (3) HC strategy-based order batch and k-opt-based order sequencing (HC + k-opt). Order picking of the FCFS + ACO method shows the performance of order picking based on the basically interventionist strategy. The FCFS + k-opt algorithm guarantees real-time performance compared to the FCFS + ACO method and is used to observe the suboptimally optimized UAV's path and the effect it has on the completion time of the actual warehouse. In order to obtain a quick simulation result, the resultant data of the simulation show a scale different from the actual one. In the simulation, when the UAV moves, it moves in four directions by moving the grid one by one. It is assumed that the moving distance of the moving UAV increases by 1 m every time it moves one grid. The computation time of the algorithm means the computation time for the method to require actually in simulation. In the experiment, since the calculation time of the ACO algorithm is longer than that of k-opt, the number of times the ACO algorithm is executed is a main factor that determines the collapsed time of each algorithm.

5.2.1. The Performance of Algorithms for Order Arrival Rates and Expired Time in Simulation

In a simulation in which online orders arrive at repeatedly constant period, the completion time for each algorithm shows different results according to the order arrival rate N_γ in Figure 6. The FCFS + ACO increases the completion time as the order arrival rate increases, like the IAA method proposed by [16]. This algorithm assigns mostly idle UAVs with orders at low N_γ. Therefore, new orders are rarely assigned to picking UAVs in the pick cycle. As a result, when the order arrival rate N_γ = 2, the completion time of the FCFS + ACO algorithm increases. On the other hand, as N_γ grows, orders are also assigned to the picking UAV. From this perspective, because the relationship between the orders and new orders of the UAV is not taken into account [16], there is a negative effect on the algorithm with the increasing completion time. FCFS + k-opt is similar to the previous method, but when allocating a new incoming order to a UAV that is already picking, the k-opt algorithm is used in the order sequencing process. If many orders come in quickly, there are almost no idle UAVs, and there is a lot of intervention in the existing order picking. This intervention causes the UAV to stop frequently in order to plan a new path for order picking. As a result, when k-opt is used, most of N_γ (N_γ = 2, 4, 6, 10) shows a smaller completion time than the ACO. Since the algorithm calculation time of k-opt is faster than that of the ACO, the reduction in the calculation time of order sequencing affects the performance. In the case of the HC + k-opt, when a new order comes in, it is decided whether to allocate the other in consideration of the similarity with the picking list of UAVs in operation. This method has the effect of reducing the overall completion time because the path is formed quickly by using the k-opt for the order assigned to the UAV. In the experiment for N_γ = 10, the frequent order arrival speeds up the generation of batches with high similarity. As a result, the waiting time for orders to be grouped with orders with high similarity is reduced.

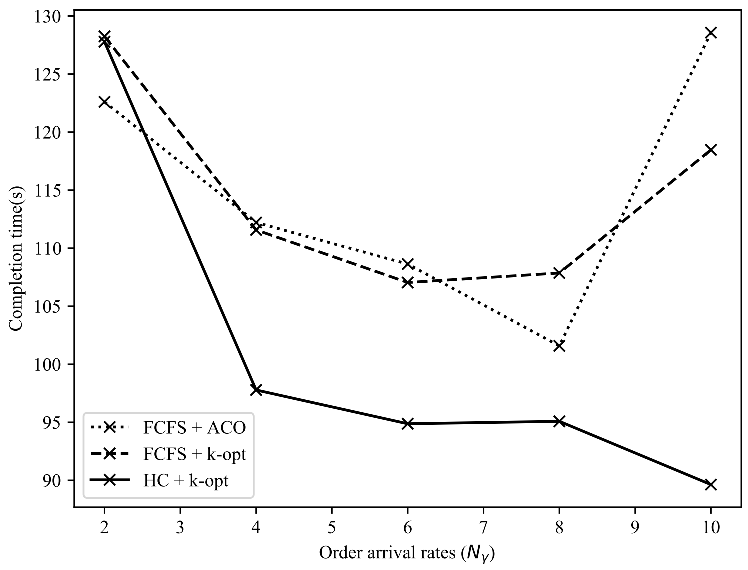

Figure 6. Comparison graph of completion time by algorithm for order arrival rate.

For online ordering, the distribution of travel distances of multi-UAVs by algorithm is expressed in the form of a box plot in Figure 7. The FCFS + ACO and the FCFS + k-opt used different order sequencing algorithms, but the UAVs to which the two algorithms are applied have similar average moving distances. Furthermore, in Figure 6, the application of k-opt with a fast operation speed affects the reduction of the completion time of the warehouse operation. The HC + k-opt method shows a box plot with a lower travel distance compared to other algorithms. In the proposed method, the overall travel distance is low and evenly distributed because the UAVs are processed in consideration of the similarity of online orders. This reduction in the moving distance of the multi-UAVs saves on the moving cost of the UAV. The saved moving cost contributes to increasing the operating time of UAVs with limited batteries in the warehouse [33].

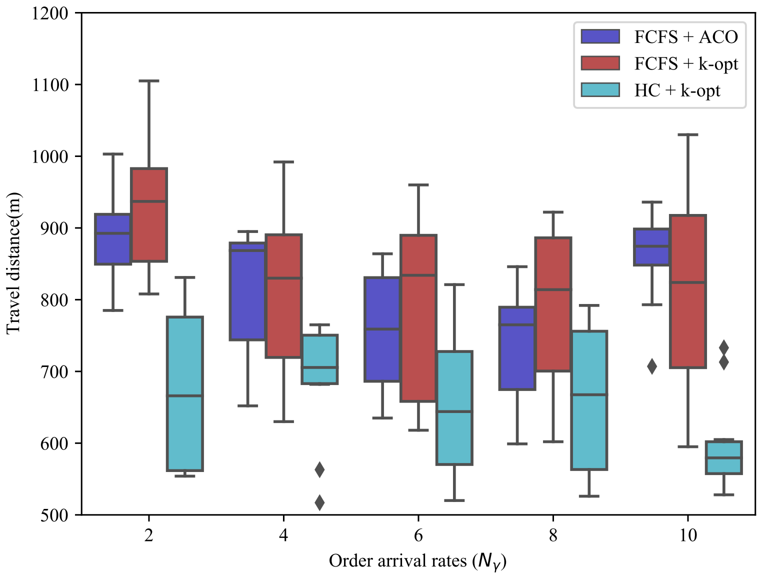

Figure 7. Comparison box plot of UAV travel distance by algorithm for order arrival rate. the rhombuses mean outliers.

The order arrival rate N_γ causes a change in the collapsed time of the UAV in Figure 8. FCFS + ACO and FCFS + k-opt show that the algorithm time increases with the increase of the order arrival rate N_γ. This is because the interference with the picking list increases as the amount of logistics increases, and the frequency of order sequencing for the modified picking list increases. The two algorithms differ in the order of sequencing interventions. The FCFS + ACO runs the ACO algorithm again whenever the picking list changes. The allocation of each order results in the addition of new items in picking lists, which means the addition of nodes in terms of the TSP problem. The overall computation time of the algorithm to solve this problem increases. On the other hand, the FCFS + k-opt uses the k-opt algorithm for the change of the picking list and shortens the computation time of the algorithm. The HC + k-opt algorithm showed similar performance to the FCFS + k-opt before N_γ = 6, but showed lower overall algorithm operation time after that. Our analysis for the result is as follows:

1. Because the HC strategy creates a new batch through waiting rather than frequently applying k-opt every time a new order arrives, the convergence time and the number of executions for the algorithm to find the optimal solution are reduced.
2. Orders added to the picking list in the HC strategy are likely to have the same item composition as the existing picking list. Since this means a decrease in the number of nodes added in the TSP algorithm, the number of algorithm executions will be reduced.

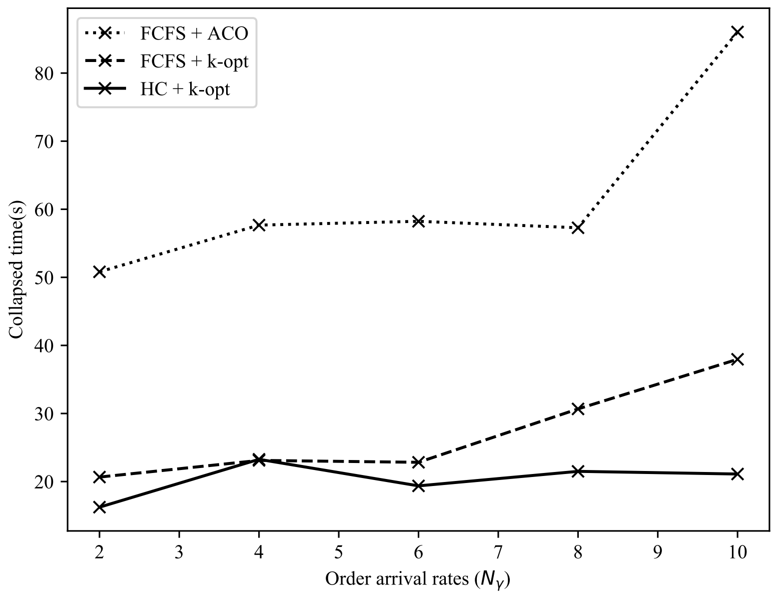

Figure 8. A comparison graph of the collapsed time for the order arrival rate. This algorithm time is the sum of the order sequencing time and the batch generation time of the order.

When the expired time T_e of the HC strategy is high, it is shown that Figure 9 affects the entire completion time of the warehouse. In these experiments, we fixed the arrival rate N_γ to 10 and gradually increased the expired time. From the experimental results, the relatively small expired time T_e = 10, 20, 50 does not have a significant effect on the overall completion time of the warehouse. On the other hand, a relatively large expired time T_e = 70, 100 makes the completion time very large. The source for this phenomenon is the remaining orders with low similarity to each other in the late stages of simulation. These orders are not assigned to UAVs unless they are expired or new orders are placed in a similar batch. As a result, the increased expired time increases the waiting time for the remaining orders.

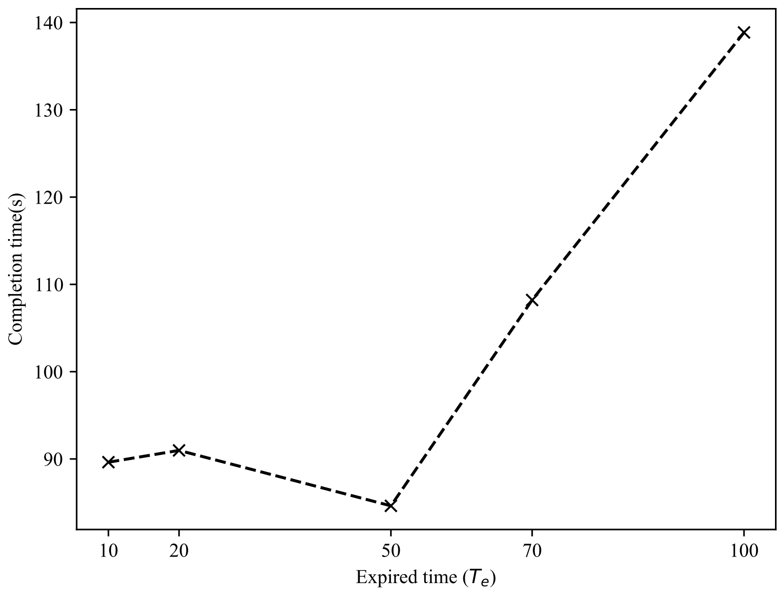

Figure 9. Completion time graph of the proposed method for expired time.

Table 2 shows detailed results of the overall performance change according to the arrival rate of the algorithm. In Table 2, Dif. represents the change rate from FCFS + ACO based on IAA to proposed HC + k-opt. When N_γ is 10, the proposed method shows the lowest completion time, median of travel distance, and collapsed time compared to the FCFS + ACO method. As a result, in the case of high order arrival, our algorithm is effective.

Table 2. Comparison of algorithm performance of the proposed method (HC + k-opt) for FCFS + ACO, FCFS + k-opt. Dif. means the change rate of the result of HC + k-opt with respect to the result of FCFS + ACO.

	N_γ	**2**	**4**	**6**	**8**	**10**
Completion Time [s]	FCFS + ACO	122.60	112.21	108.63	101.58	128.57
	FCFS + k-opt	128.25	111.57	107.04	107.86	118.47
	HC + k-opt	127.78	97.78	94.87	95.08	89.63
	Dif.	4.23%	−12.86%	−12.67%	−6.4%	−30.29%
Median of Travel Distance (n_r = 10) [m]	FCFS + ACO	892.50	868.50	759.00	765.00	874.50
	FCFS + k-opt	937.00	830.00	834.00	814.00	824.00
	HC + k-opt	666.00	705.50	644.00	667.50	579.50
	Dif.	−25.38%	−18.77%	−15.15%	−12.75%	−33.73%
Collapsed Time [s]	FCFS + ACO	50.81	57.67	58.23	57.28	86.04
	FCFS + k-opt	20.66	23.11	22.83	30.67	37.96
	HC + k-opt	16.24	23.27	19.37	21.50	21.11
	Dif.	−68.04%	−60.65%	−66.74%	−62.47%	−75.46%

5.2.2. The Performance of Algorithms for Number of UAVs

In order to check whether the order sequencing proposed in the warehouse is effective, we changed the number of UAVs, and the results were checked through the time required to calculate the algorithm and the distance traveled by the UAV. The two results can be confirmed in Figures 10 and 11, respectively. Figure 10 shows the time when an algorithm is performed once and expresses it as elapsed time. The HC strategy was used for order batching, and the ACO and proposal methods were used respectively. When the number of

UAVs increases, the proposed method shows a fast processing speed of about 0.4 s, while the comparison method is slower, with a maximum of 1.6 and a minimum of about 1 s. The proposed method eliminates the twisted edge caused by the added order using the k-opt method, so the time required to optimize the path is smaller than that of the comparison method. On the other hand, comparison methods require a new weight for the positions of all shelves, which results in a longer required time to complete the optimization.

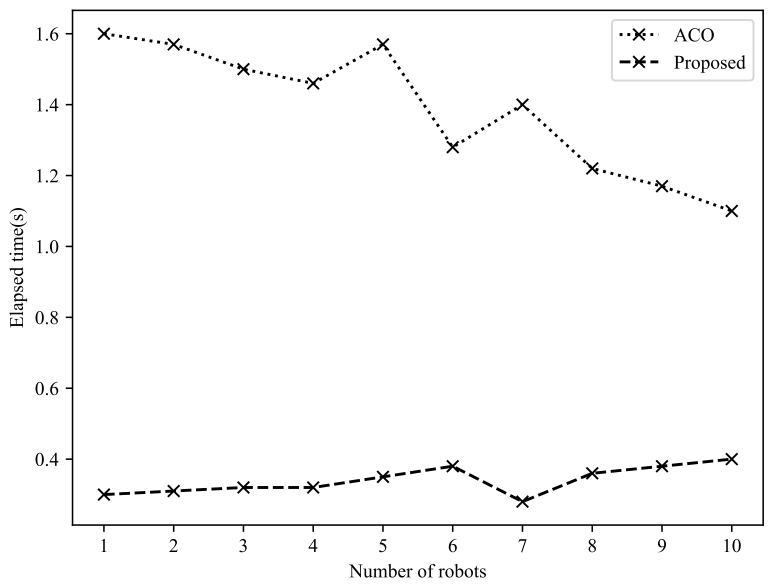

Figure 10. Elapsed time graph of the proposed method for number of UAVs.

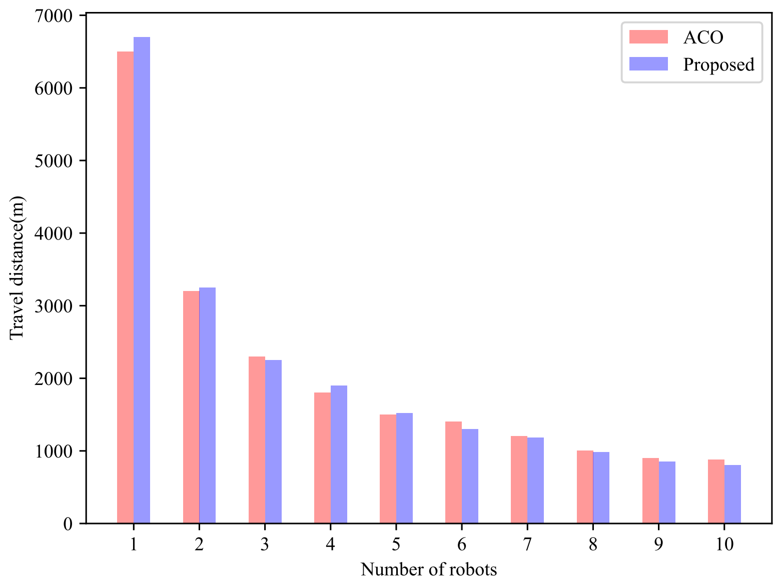

Figure 11. Travel distance graph of the proposed method for number of UAVs.

In Figure 11, the average of the distance traveled by all UAVs was used, and the comparison method and the proposed method show similar distance. When n_r is 1∼5, it can be seen that the comparison algorithm generates a slightly shorter path. In the case of using a small number of UAVs, the performance of the comparison method is slightly better. The major cause is that this case is more closer to the TSP problem rather than the DTSP problem after a certain period of time due to the limited c_{max}. On the other hand, when n_r increases from 6 to 10, the performance of the proposed method is better due to the frequent intervention. Using the proposed algorithm, there was an increase in the path of approximately 1.5%, which does not differ significantly from the optimal path obtained through the comparison algorithm. Through this result, the proposed algorithm has a faster processing speed than the comparison algorithm, and the generated path is efficient without showing a significant difference. Table 3 shows detailed results of the overall performance change according to the number of UAVs.

Table 3. Comparison of algorithm performance of the proposed order sequencing method.

	N_r	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Completion Time [s]	ACO	1.60	1.57	1.50	1.46	1.57	1.28	1.40	1.22	1.27	1.10
	Proposed	0.30	0.31	0.32	0.32	0.35	0.38	0.28	0.36	0.38	0.40
Travel Distance [m]	ACO	6512	3205	2298	1811	1480	1402	1222	998	887	868
	Proposed	6712	3254	2256	1905	1528	1311	1187	982	852	807

6. Conclusions

In this paper, we propose the order picking method with the HC strategy and k-opt-based algorithm to efficiently process orders received in real time in a logistics environment. The HC strategy is an improved method of allocating orders in consideration of multi-UAVs based on the previous interventionist strategy. The UAV in the warehouse uses different path generation algorithms depending on the UAV's picking list. When a new order is assigned to an UAV already in the aisle, a new path is created using the original path based on k-opt. The experiments were implemented in simulations of warehouses with a general layout. The proposed method is compared with the FCFS strategy for order batching, and the ACO algorithm for order sequencing. The performance of the proposed method was confirmed by changing the order arrival rates and the number of UAVs in the warehouse. By using the the HC strategy, even when a new order arrives, it reduces the convergence time and the number of executions to find the optimal solution, showing a small completion time even with an increase in order arrival rates. Furthermore, the proposed method finds the optimal path by resolving the twisted path. The proposed algorithm executes about 2.5 times faster than the previous method. Nevertheless, there was no big difference in completion time compared to the the previous TSP solving algorithm. It shows that it is more suitable for the logistics environment when using multi-UAVs that require fast work.

Our approach was created to efficiently build a logistics system using multiple UAVs. However, it is necessary to assume that starting and returning from a fixed depot and collisions between UAVs are not taken into account. Moreover, since it is a structure that handles all logistics allocation centrally, there is the disadvantage that there must be a strong connection between all UAVs and the central network. In future research, order batch through communication between UAVs will be studied to share the control of the center. In addition, the assumption of sequencing is used only when new items are added, but in order to reflect reality, we will study various situations such as removing and changing orders, and verify in not only simulation, but also actual warehouse environments.

Author Contributions: Conceptualization, C.H. and H.J.; Methodology, C.H. and H.J.; Project administration, H.L.; Software, H.J.; Supervision, J.O.; Validation, C.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A4A3033961, No.2020R1F1A1076667), and was conducted by the Research Grant of Kwangwoon University in 2022.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Almalki, F.A.; Albraikan, A.A.; Soufiene, B.O.; Ali, O. Utilizing artificial intelligence and lotus effect in an emerging intelligent drone for persevering solar panel efficiency. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 7741535. [CrossRef]
2. Takechi, H.; Aragaki, S.; Irie, M. Differentiation of river sediments fractions in UAV aerial images by convolution neural network. *Remote Sens.* **2021**, *13*, 3188. [CrossRef]
3. Padmanabhan, M.; Suresh, G.R. Coalition formation and task allocation of multiple autonomous robots. In Proceedings of the International Conference on Signal Processing, Communication and Networking (ICSCN), Chennai, India, 26–28 March 2015; pp. 1–5. [CrossRef]
4. Alsamhi, S.H.; Lee, B. Blockchain-Empowered multi-robot collaboration to fight COVID-19 and future pandemics. *IEEE Access* **2021**, *9*, 44173–44197. [CrossRef]
5. Wurman, P.R.; D'Andrea, R.; Mountz, M. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Mag.* **2008**, *29*, 9. [CrossRef]
6. Alsamhi, S.H.; Ma, O.; Ansari, M.S.; Almalki, F.A. Survey on collaborative smart drones and internet of things for improving smartness of smart cities. *IEEE Access* **2019**, *7*, 128125–128152. [CrossRef]
7. Skorobogatov, G.; Barrado, C.; Salamí, E. Multiple UAV systems: A survey. *Unmanned Syst.* **2020**, *8*, 149–169. [CrossRef]
8. Shakeri, R.; Al-Garadi, M.A.; Badawy, A.; Mohamed, A.; Khattab, T.; Al-Ali, A.K.; Harras, K.A.; Guizani, M. Design challenges of multi-UAV systems in cyber-physical applications: A comprehensive survey and future directions. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3340–3385. [CrossRef]
9. Malmborg, C.J.; Al-Tassan, K. An integrated performance model for orderpicking systems with randomized storage. *Appl. Math. Model.* **2000**, *24*, 95–111. . [CrossRef]
10. De Koster, R.; Le-Duc, T.; Roodbergen, K.J. Design and control of warehouse order picking: A literature review. *Eur. J. Oper. Res.* **2007**, *182*, 481–501. [CrossRef]
11. Tang, L.C.; Chew, E.P. Order picking systems: Batching and storage assignment strategies. *Comput. Ind. Eng.* **1997**, *33*, 817–820. [CrossRef]
12. Tompkins, J.A.; White, J.A.; Bozer, Y.A.; Tanchoco, J.M.A. *Facilities Planning*; John Wiley & Sons: Hoboken, NJ, USA, 2010.
13. ElGibreen, H.; Youcef-Toumi, K. Dynamic task allocation in an uncertain environment with heterogeneous multi-agents. *Auton. Robot.* **2019**, *43*, 1639–1664. [CrossRef]
14. Gong, Y.; De Koster, R. A polling-based dynamic order picking system for online retailers. *IIE Trans.* **2008**, *40*, 1070–1082. [CrossRef]
15. Rubrico, J.I.; Higashi, T.; Tamura, H.; Ota, J. Online rescheduling of multiple picking agents for warehouse management. *Robot. -Comput.-Integr. Manuf.* **2011**, *27*, 62–71. [CrossRef]
16. Giannikas, V.; Lu, W.; Robertson, B.; McFarlane, D. An interventionist strategy for warehouse order picking: Evidence from two case studies. *Int. J. Prod. Econ.* **2017**, *189*, 63–76. [CrossRef]
17. Ran, W.; Liu, S.; Zhang, Z. A polling-based dynamic order-picking system considering priority orders. *Complexity* **2020**, *2020*, 4595316. [CrossRef]
18. Ramanathan, R.; George, J.; Ramanathan, U. The role of logistics in e-commerce transactions: An exploratory study of customer feedback and risk. In *Supply Chain Strategies, Issues and Models*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 221–233. [CrossRef]
19. Le-Duc, T.; De Koster, R.M. Travel time estimation and order batching in a 2-block warehouse. *Eur. J. Oper. Res.* **2007**, *176*, 374–388. [CrossRef]
20. Zhang, J.; Wang, X.; Chan, F.T.; Ruan, J. On-line order batching and sequencing problem with multiple pickers: A hybrid rule-based algorithm. *Appl. Math. Model.* **2017**, *45*, 271–284. [CrossRef]
21. Van Nieuwenhuyse, I.; de Koster, R.B. Evaluating order throughput time in 2-block warehouses with time window batching. *Int. J. Prod. Econ.* **2009**, *121*, 654–664. [CrossRef]
22. Henn, S. Algorithms for on-line order batching in an order picking warehouse. *Comput. Oper. Res.* **2012**, *39*, 2549–2563. [CrossRef]
23. Stodola, P.; Michenka, K.; Nohel, J.; Rybanskỳ, M. Hybrid algorithm based on ant colony optimization and simulated annealing applied to the dynamic traveling salesman problem. *Entropy* **2020**, *22*, 884. [CrossRef]
24. Strąk, Ł.; Skinderowicz, R.; Boryczka, U.; Nowakowski, A. A self-adaptive discrete PSO algorithm with heterogeneous parameter values for dynamic TSP. *Entropy* **2019**, *21*, 738. [CrossRef]
25. Simões, A.; Costa, E. Extended virtual loser genetic algorithm for the dynamic traveling salesman problem. In Proceedings of the Annual Conference on Genetic and Evolutionary Computation, Amsterdam, The Netherlands, 6–10 July 2013; pp. 869–876.
26. Guntsch, M.; Middendorf, M. Pheromone modification strategies for ant algorithms applied to dynamic TSP. In Proceedings of the Workshops on Applications of Evolutionary Computation, Como, Italy, 18–20 April 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 213–222.
27. Mavrovouniotis, M.; Yang, S. Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. *Appl. Soft Comput.* **2013**, *13*, 4023–4037. [CrossRef]
28. Melo, L.; Pereira, F.; Costa, E. Multi-caste ant colony algorithm for the dynamic traveling salesperson problem. In Proceedings of the International Conference on Adaptive and Natural Computing Algorithms, Lausanne, Switzerland, 4–6 April 2013; pp. 179–188. [CrossRef]

29. Boryczka, U.; Strąk, Ł. A hybrid discrete particle swarm optimization with pheromone for dynamic traveling salesman problem. In Proceedings of the International Conference on Computational Collective Intelligence, Ho Chi Minh City, Vietnam, 28–30 November 2012; pp. 503–512. [CrossRef]
30. Saravanan, S.; Ramanathan, K.C.; Ramya, M.M.; Janardhanan, M.N. Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems. *Ind. Robot* **2020**, *47*, 929–942. [CrossRef]
31. Sariel-Talay, S.; Balch, T.R.; Erdogan, N. Multiple traveling robot problem: A solution based on dynamic task selection and robust execution. *IEEE/ASME Trans. Mechatron.* **2009**, *14*, 198–206. [CrossRef]
32. Nunes, E.; Manner, M.; Mitiche, H.; Gini, M. A taxonomy for task allocation problems with temporal and ordering constraints. *Robot. Auton. Syst.* **2017**, *90*, 55–70. [CrossRef]
33. Setter, T.; Egerstedt, M. Energy-Constrained Coordination of Multi-Robot Teams. *IEEE Trans. Control Syst. Technol.* **2017**, *25*, 1257–1263. [CrossRef]
34. Zhang, J.; Lu, Y.; Che, L.; Zhou, M. Moving-Distance-Minimized PSO for Mobile Robot Swarm. *IEEE Trans. Cybern.* **2022**, *52*, 9871–9881. [CrossRef]
35. Chen, F.; Wang, H.; Xie, Y.; Qi, C. An ACO-Based online routing method for multiple order pickers with congestion consideration in warehouse. *J. Intell. Manuf.* **2016**, *27*, 389–408. [CrossRef]
36. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
37. Choe, K. Aisle-Based order pick systems with batching, zoning, and sorting. In *School of Industrial and Systems Engineering*; Georgia Institute of Technology: Atlanta, GA, USA, 1990.
38. Sttzle, T. MAX-MIN ant system. *Future Gener. Comput. Syst.* **2000**, *16*, 889–914. [CrossRef]
39. Aboelfotoh, A.; Singh, M.; Suer, G. Order Batching Optimization for Warehouses with Cluster-Picking. *Procedia Manuf.* **2019**, *39*, 1464–1473. [CrossRef]
40. Fibrianto, H.Y.; Hong, S. Dynamic order batching in bucket brigade order picking systems with consecutive batch windows and non-identical pickers. *Int. J. Prod. Res.* **2019**, *57*, 6552–6568. [CrossRef]