2022 年臺灣國際科學展覽會
優勝作品專輯

作品編號	**190047**
參展科別	電腦科學與資訊工程
作品名稱	**A.N.T.s: Algorithm for Navigating Traffic System in Automated Warehouses**
得獎獎項	

國　　家	**Indonesia**
就讀學校	**Cita Hati Christian Senior School West Campus**
指導教師	**Kuncoro Kohar, S.T., B.Eng., M.Sc.**
作者姓名	**Audrey Cherilyn Ilham**

關鍵詞　**Automated warehouse system、heuristic algorithm、pathfinding**

作者照片

ABSTRACT

According to CNN Indonesia 2020, the demand for e-Commerce in Indonesia has nearly doubled during this pandemic. This surge in demand calls for a time-efficient method for warehouse order-picking. One approach to achieve that goal is by incorporating automation in their warehouse systems. Globally, the market of warehouse robotics is expected to reach 12.6 billion USD by 2027 (Data Bridge Market Research, 2020).

In this research, the warehouse system studied would utilize AMR (Autonomous Mobile Robots) to lift and deliver movable shelf units to the packing station where workers are at. This research designed a heuristic algorithm called A.N.T.s (Algorithm for Navigating Traffic System) to conduct task assigning and pathfinding for AMR in the automated warehouse. The warehouse layout was drawn as a two-dimensional map in grids. When an order is placed, A.N.T.s would assign the task to a robot that would require the least amount of time to reach the target shelf. A.N.T.s then conducted pathfinding heuristically using Manhattan Distance. A.N.T.s would help the robot to navigate its way to the target shelf unit, lift the shelf and bring it to the designated packing station. A.N.T.s algorithm was tested in various warehouse layouts and with a varying number of AMRs. Comparison against the commonly used Djikstra's algorithm was also conducted (Shaikh and Dhale, 2013). Results show that the proposed A.N.T.s algorithm could execute 100 orders in a 27x23 layout with five robots 9.96 times faster than Dijkstra with no collisions. The algorithm is also shown to be able to help assign tasks to robots and help them find short paths to navigate their ways to the shelf units and packing stations. A.N.T.s could navigate traffic to avoid deadlocks and collisions in the warehouse with the aid of lanes and directions.

1. INTRODUCTION

1.1. Research Background

The demand for e-Commerce has seen a significant increase, nearly doubling during this pandemic in Indonesia [3]. As lockdown restrictions are stricken and shopping malls become harder to visit, consumers have resorted to the alternative e-Commerce.

With a click of a button, consumers would expect their orders to be delivered right at their doorstep as quickly as possible. A high rise in demand calls for a time-efficient and cost-effective method for warehouse order-picking. Order-picking is the process of collecting the goods and items needed to fulfill a customer's order. The efficiency of warehouse operations rely heavily on their order-picking process. Order-picking is generally the most time-consuming process in a warehouse and accounts for 70% of the time and 55% of the cost of warehouse activity [2]. Hence, optimizing order-picking should be a priority in increasing the efficiency and decreasing the costs of warehouse logistics.

In conventional warehouses, items are collected by workers who go around the warehouse by foot or forklift to fetch said items on the shelves and take them back to the packaging stations. This method is very time-consuming and energy demanding as workers must walk around large warehouses to look for items scattered in various locations. Another common method is to use conveyor belts which are costly and take up a lot of space. With the advancement of automation, robots are starting to be integrated in logistic systems. Two methods of automation may be applied:

1) Pick and place robots moving to the designated fixed shelves to pick up the items and bring the items to the packing station.

2) Autonomous mobile robots (AMR) moving to the designated movable shelves to lift the shelf and bring the shelf to the packing station.

In the former, path planning of robots can be done using the Traveling Salesman Problem (TSP) or the Vehicle Routing Problem (VRP) [11]. However, this first method comes with an obstacle as items may come in various sizes, shapes, and textures, which leads to the complex development of the robots' detection and grasping ability. If the items were to be stored in boxes, there would be limitations on how many boxes a robot can grab at a time.

The latter method, with movable shelves, would be explored in this research. autonomous mobile robots (AMRs) are a type of automated guided vehicle that may be deployed without the need for any supporting infrastructure or human intervention [1].

1.2. Automated Warehouse Concept

This research proposes an automated warehouse order-picking approach that utilizes autonomous mobile robots that navigate their ways through the warehouse to lift, transport, and place movable shelf units from one location in the warehouse to another. Therefore, workers are not required to move around the warehouse. Only a few workers are needed and they are positioned at the packing station.

The goods and items are placed and stored in movable shelf units throughout the warehouse. When an order is placed, the control system would assign the task to a robot that would require the least amount of time to reach the target shelf. The control system would instruct the robot to move to the target shelf unit where the item is located. The robot would then slide under the target shelf, lift it up and move along the shortest path calculated by the algorithm to reach the designated packing station. At the packing station, the workers would retrieve individual items from the shelf units. After the required items are retrieved by the workers, the robot would transport the shelf unit back to its original location.

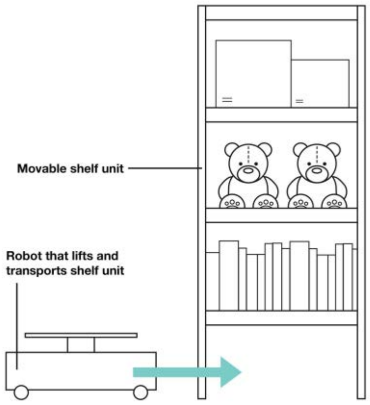


Figure 1. Robot sliding under a moveable shelf unit to lift it up

2. RESEARCH GOAL

- Develop an algorithm that is able to handle task assigning and pathfinding for multiple robots simultaneously and to avoid deadlocks, collisions, and obstacles in the warehouse.
- Evaluate algorithm's performance against other algorithm.
- Test the algorithm performance through simulations in various layouts and with varying numbers of robots.

3. RESEARCH METHODOLOGY

3.1. Warehouse Environment Layout

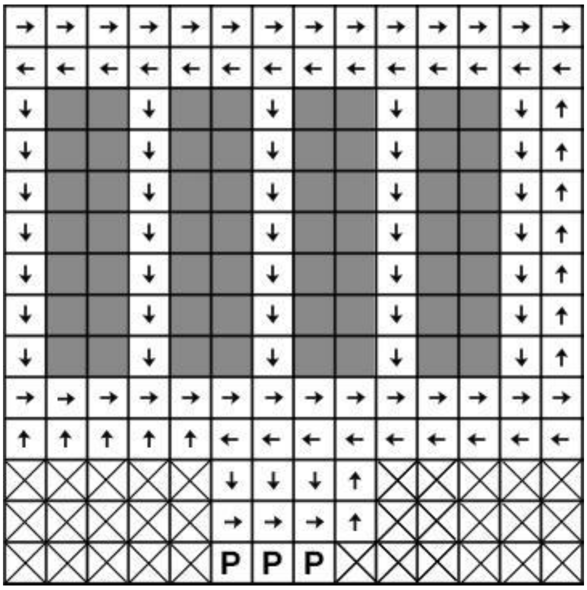


Figure 2. Sample of warehouse environment layout

Figure 2 shows a warehouse environment layout studied in this research. The above layout is drawn in a 14x14 grid comprising 56 movable shelf units and three packing stations. To avoid collisions and deadlocks, each road is specified to a single direction.

Table 1. Warehouse environment layout legend

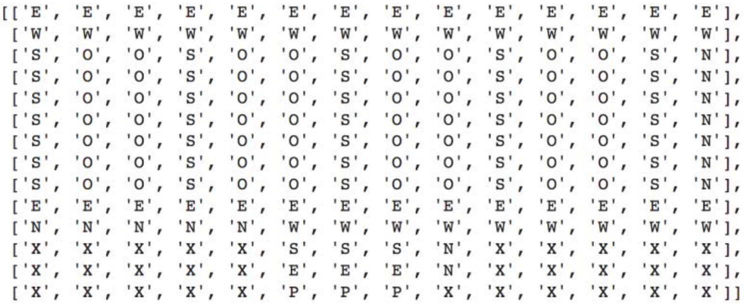
Icon	Name	Description
■	Shelf unit	The shelf unit that stores goods and items. It would be picked up and transported by robots to the packing station. There can be any number of shelf units in the warehouse.
←	Road	The path robots can move on. Each road is specified for one direction. Robots may move to adjacent roads to switch lanes.
P	Packing station	Where warehouse workers would pick up and pack items from the shelf units delivered to them. There can be any

		number of packing stations in the warehouse.

A control system would manage all the robots. This control system would assign each robot a shelf unit they would need to pick up and deliver and the path they need to follow. To avoid collisions and to aid in task assigning and pathfinding, all robots would report their location to the control system everytime it moves a grid by scanning codes on the road. In addition, each robot would be equipped with sensors to detect other obstacles in their way.

3.2. Mapping

The warehouse environment layout would be stored as a two-dimensional map as shown in Figure 3.



```
[['E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E'],
 ['W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'O', 'O', 'S', 'N'],
 ['E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E', 'E'],
 ['N', 'N', 'N', 'N', 'N', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W', 'W'],
 ['X', 'X', 'X', 'X', 'X', 'S', 'S', 'S', 'N', 'X', 'X', 'X', 'X', 'X'],
 ['X', 'X', 'X', 'X', 'X', 'E', 'E', 'E', 'N', 'X', 'X', 'X', 'X', 'X'],
 ['X', 'X', 'X', 'X', 'X', 'P', 'P', 'P', 'X', 'X', 'X', 'X', 'X', 'X']]
```

Figure 3. 2D map of warehouse environment layout

Figure 3 would act as an unchangeable reference. A different map would store the same layout with additions of each robots' position, and would be updated every time the robots moved.

3.3. Proposed Algorithm

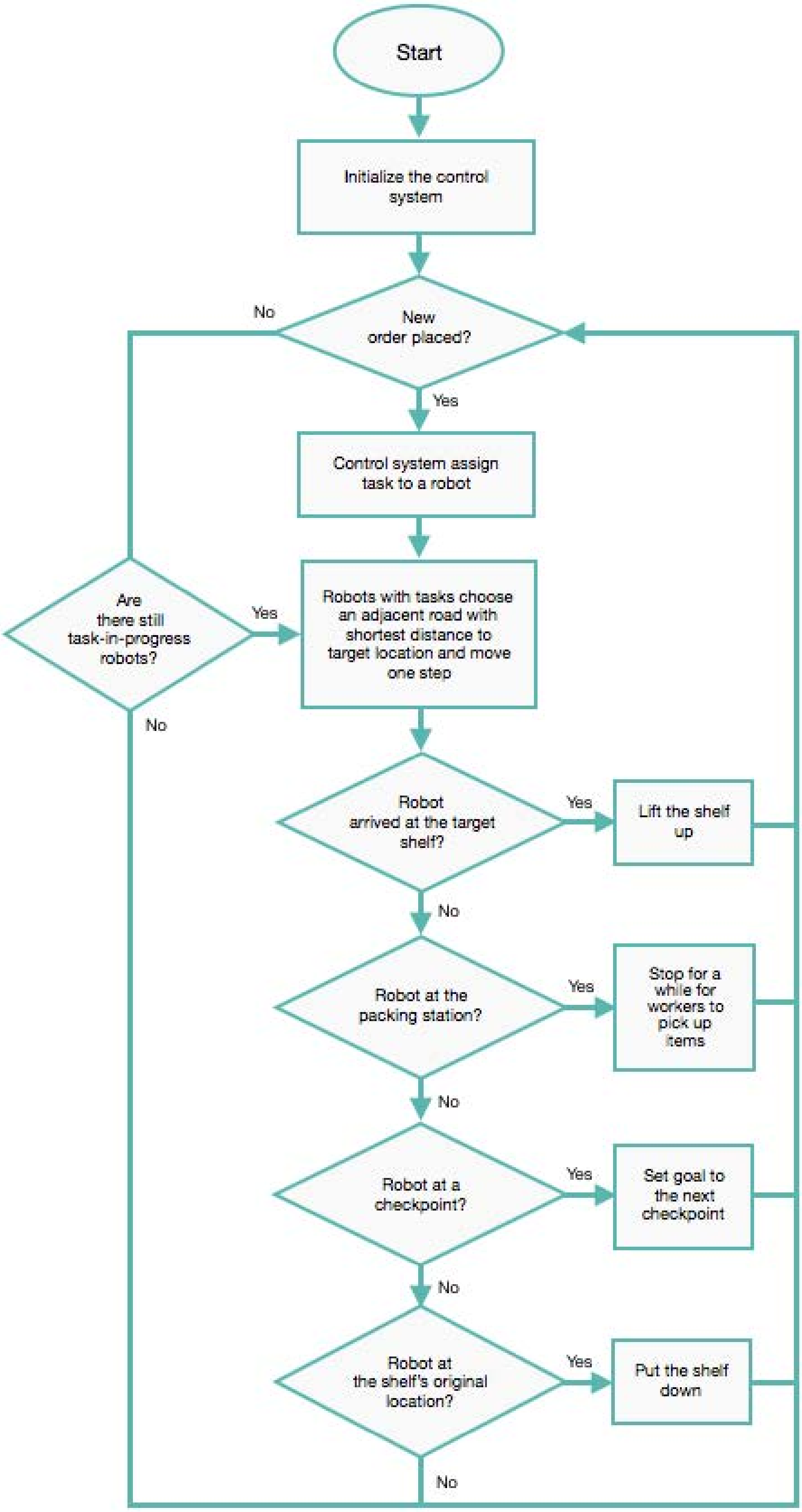

Start
Initialize the control system
New order placed?
No
Yes
Control system assign task to a robot
Are there still task-in-progress robots?
Yes
No
Robots with tasks choose an adjacent road with shortest distance to target location and move one step
Robot arrived at the target shelf?
Yes
Lift the shelf up
No
Robot at the packing station?
Yes
Stop for a while for workers to pick up items
No
Robot at a checkpoint?
Yes
Set goal to the next checkpoint
No
Robot at the shelf's original location?
Yes
Put the shelf down
No

Diagram 1. A.N.T.s outline

Step 1: Task assigning to robots

The first task of the control system is to assign delivery tasks to robots as efficiently as possible. When receiving a new order, the control system must assign it to a robot that requires the least amount of time to reach the targeted shelf unit.

Distance between a robot's origin and the targeted shelf unit of the new task is calculated using Manhattan distance (M) as follows:

$$M = |x_{origin} - x_{target}| + |y_{origin} - y_{target}|$$

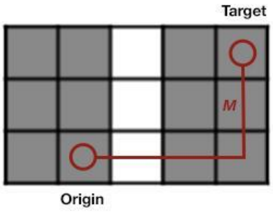


Figure 4. Manhattan Distance from a robot's origin

There are three modes of robots that can be assigned a new task:

1) idle robots,
2) robots carrying out another task, and
3) robots already carrying the target shelf because the shelf was requested by another task.

Idle robots can move to the designated shelf unit immediately when assigned a new task. Therefore, the total time needed to complete the task would be the time needed for it to move to the targeted shelf unit. Since a task ends after returning the shelf unit, idle robots are parked under a shelf unit. Therefore, distance from an idle robot's origin to a new targeted shelf can be defined as follows:

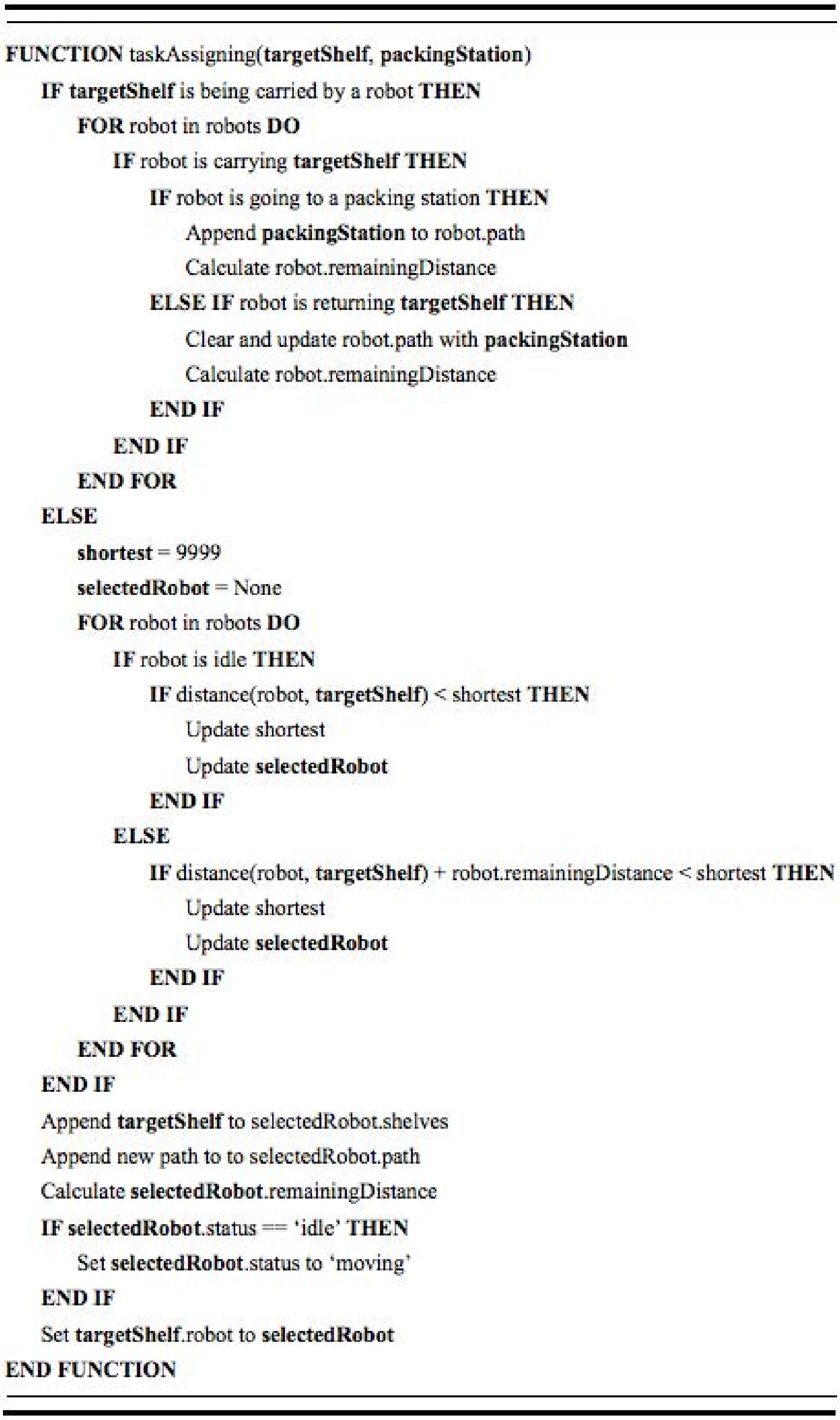
$$total\ distance = M$$

On the other hand, robots carrying out another task must finish its initial task before executing a new task given. In other words, the total distance needed to complete the task would be the remaining distance needed for it to complete its initial task plus the distance between the final position of the initial task and the position of the new task. In this case total distance would be defined as:

$$total\ distance = D + M$$
$$D = remaining\ distance\ from\ initial\ task$$

Robots already carrying the target shelf because the shelf was requested by another task would be prioritized over the two other modes of robot. If the robot has not been to the first task's packing station, the new task's packing station would be added to the robot's route. If the robot has already been to the first task's packing station, the robot's task would be interrupted and it must immediately reroute to the new task's packing station.

For optimum task completion, all robots in the warehouse are evaluated. The robot with the shortest completion distance would be selected.



```
FUNCTION taskAssigning(targetShelf, packingStation)
    IF targetShelf is being carried by a robot THEN
        FOR robot in robots DO
            IF robot is carrying targetShelf THEN
                IF robot is going to a packing station THEN
                    Append packingStation to robot.path
                    Calculate robot.remainingDistance
                ELSE IF robot is returning targetShelf THEN
                    Clear and update robot.path with packingStation
                    Calculate robot.remainingDistance
                END IF
            END IF
        END FOR
    ELSE
        shortest = 9999
        selectedRobot = None
        FOR robot in robots DO
            IF robot is idle THEN
                IF distance(robot, targetShelf) < shortest THEN
                    Update shortest
                    Update selectedRobot
                END IF
            ELSE
                IF distance(robot, targetShelf) + robot.remainingDistance < shortest THEN
                    Update shortest
                    Update selectedRobot
                END IF
            END IF
        END FOR
    END IF
    Append targetShelf to selectedRobot.shelves
    Append new path to to selectedRobot.path
    Calculate selectedRobot.remainingDistance
    IF selectedRobot.status == 'idle' THEN
        Set selectedRobot.status to 'moving'
    END IF
    Set targetShelf.robot to selectedRobot
END FUNCTION
```

Figure 5. Task assigning

Step 2: Pathfinding

After the robot moves to the starting position of the task (the target shelf unit), the next step is to find a collision-free short path for the robot to move to the packing station and then return the shelf unit to its original location. Pathfinding would be conducted heuristically.

Due to roads being one-way, the warehouse layout was added checkpoints classified into different colors as shown in Figure 6.

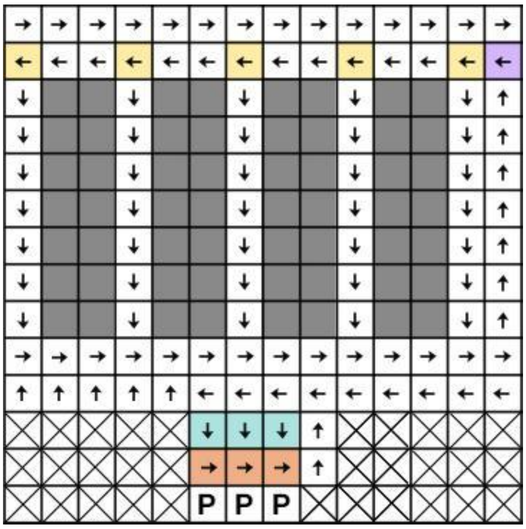


Figure 6. Sample warehouse environment layout with checkpoints

Each checkpoint will act as the robot's 'goal'. The goal will loop as shown in Figure 7.

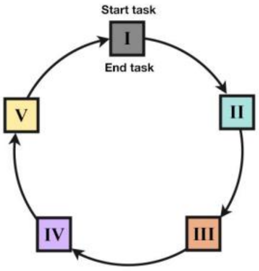
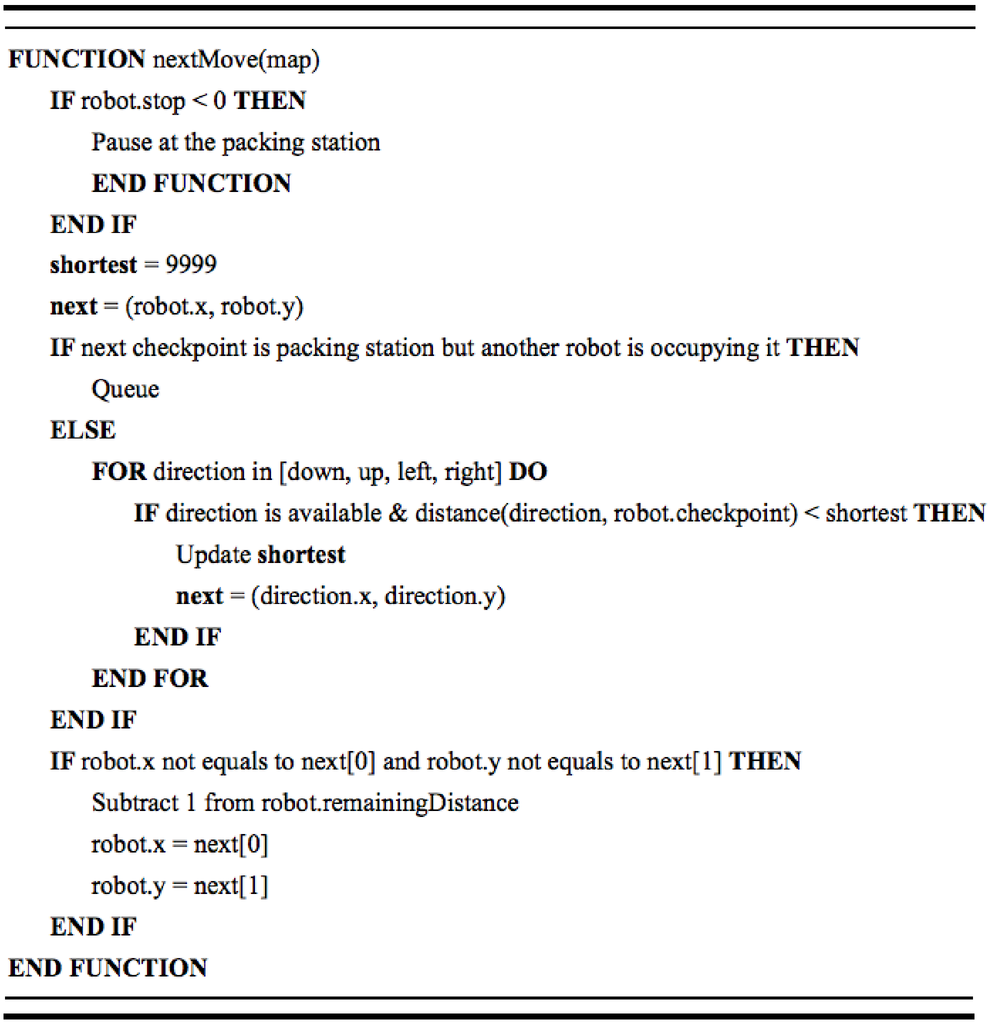


Figure 7. Checkpoints order

After picking up a shelf unit, the robot would need to pass one of the blue and orange checkpoints depending on which packing station their target is. The blue checkpoint acts as a queuing spot. It must then pass through the purple checkpoint and one of the yellow checkpoints depending on which aisle the shelf unit's original location is at. After that, it may return the shelf unit to its original location, completing its given task.

Every time the robot moves one grid, it will check its four adjacent grids and decide which grid to move to next with the following criteria:

1) No other robot is about to move there.
2) May move under a shelf if it is not carrying a shelf.
3) The grid must have a direction that does not collide with its movement.
4) The grid must have the shortest Manhattan Distance to the current active checkpoint.
5) Queue (stop) at the blue checkpoint if another robot is at its targeted packing station.
6) Prioritize moving under a shelf.



```
FUNCTION nextMove(map)
    IF robot.stop < 0 THEN
        Pause at the packing station
        END FUNCTION
    END IF
    shortest = 9999
    next = (robot.x, robot.y)
    IF next checkpoint is packing station but another robot is occupying it THEN
        Queue
    ELSE
        FOR direction in [down, up, left, right] DO
            IF direction is available & distance(direction, robot.checkpoint) < shortest THEN
                Update shortest
                next = (direction.x, direction.y)
            END IF
        END FOR
    END IF
    IF robot.x not equals to next[0] and robot.y not equals to next[1] THEN
        Subtract 1 from robot.remainingDistance
        robot.x = next[0]
        robot.y = next[1]
    END IF
END FUNCTION
```

Figure 8. Choosing a robot's next move

3.4. Simulation Testing Method

The algorithm was implemented with Python 3.0 to be tested. Simulations were visualized using Pygame and ran in Jupyter Notebook. Four types of simulations were conducted:

3.4.1. Task Assigning Simulation

In each of the simulations in this section, two robots (A and B) would be situated in the warehouse.

Table 2. Task assigning simulation cases

No.	Case	Orders Placed
1	Multiple robots	Shelf 03 to Packing Station 1 at timestamp 0; Shelf 16 to Packing Station 0 at timestamp 0

2	Shortest distance robot	Shelf 03 to Packing Station 1 at timestamp 0; Shelf 21 to Packing Station 2 at timestamp 48
3	New order on shelf already on the way to packing station	Shelf 03 to Packing Station 1 at timestamp 0; Shelf 03 to Packing Station 0 at timestamp 10
4	New order on shelf already on the way returning	Shelf 03 to Packing Station 1 at timestamp 0; Shelf 03 to Packing Station 2 at timestamp 38

3.4.2. Pathfinding Simulation

Six robots are dispatched to the warehouse with randomly generated orders placed at the beginning of each of the 10 rounds.

3.4.3. Comparing A.N.T.s to Dijkstra's algorithm in Various Layouts

In the test, Dijkstra's algorithm finds the shortest path from the shelf unit to the packing station and back by exploring all nodes. Roads are stored as nodes with equally weighted edges in an adjacency list graph. Both Djikstra and A.N.T.s were applied in three warehouse layouts. For each trial in each layout, five robots were dispatched and 20 randomly generated orders were placed. The same set of orders were given to the algorithms for completion time comparison.

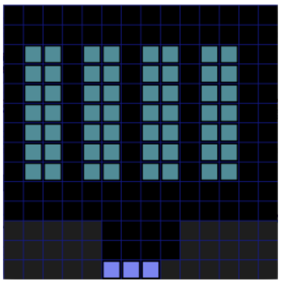


Figure 9. 14x14 layout

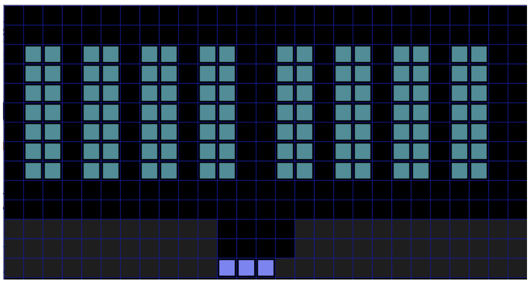


Figure 10. 27x14 layout

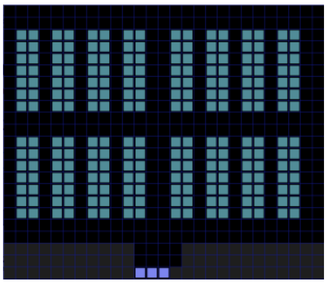


Figure 11. 27x23 layout

3.4.4. Number of Robots vs. Number of Steps

Three different sets of 100 orders were generated randomly. Each set of orders was executed five times each time with a different number of robots (2, 4, 6, 8, and 10) on a 14x14 layout (Figure 9). The number of steps needed to complete all orders in each set was evaluated. The simulation was also conducted on a 27x14 layout (Figure 10) with 4, 8, 12, 16, and 20 robots.

4. RESULTS AND ANALYSIS

Dynamic movement of the warehouse system may be viewed in this Google Drive link (https://drive.google.com/file/d/1ItaUoXza7_YtYlI3DjiVepzAc2_rD1m1/view?usp=sharing).

4.1. Task Assigning Simulation

Case 1: Multiple robots

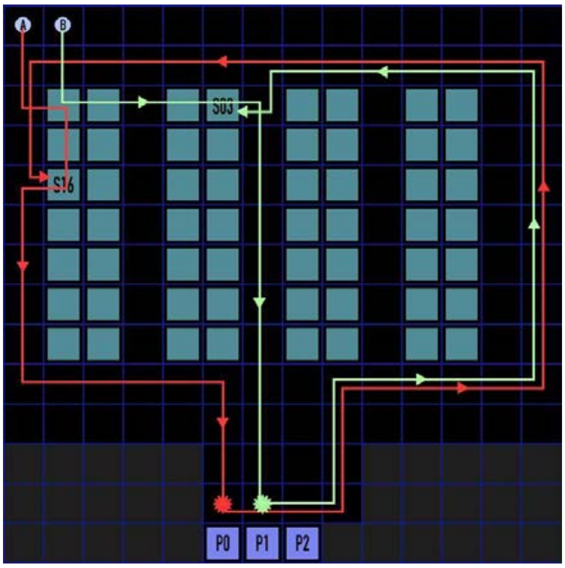


Figure 9. Task assigning simulation Case 1 result

A.N.T.s was able to split the tasks according to the shortest distance of the robots to the shelf units (Robot A to Shelf 16 and Robot B to Shelf 03).

Case 2: Shortest distance robot

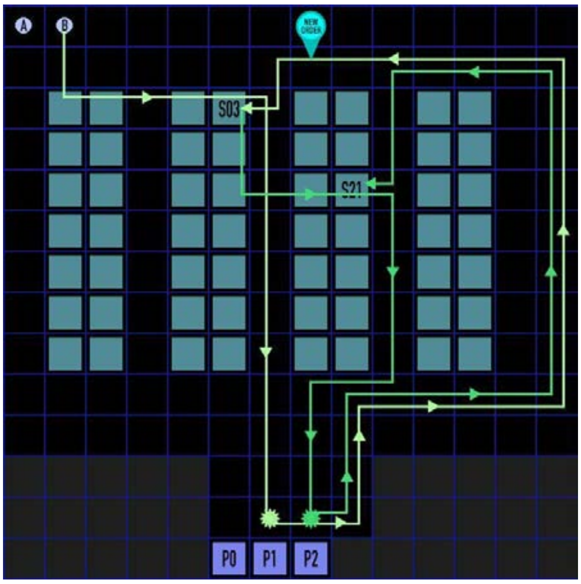


Figure 10. Task assigning simulation Case 2 result

A.N.T.s chose to assign the Shelf 21 task to Robot B instead of the idle Robot A. This is because it takes less distance for Robot B to complete its Shelf 03 task and go to Shelf 21 than for Robot A to go to Shelf 21.

Case 3: New order on shelf already on the way to packing station

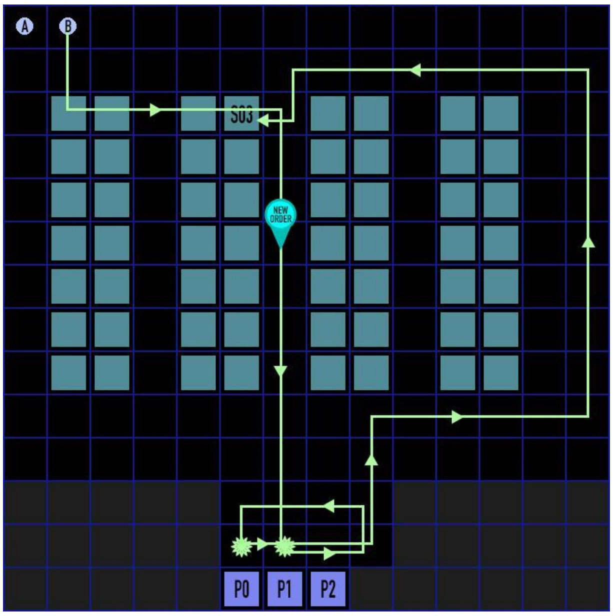


Figure 11. Task assigning simulation Case 3 result

A.N.T.s appended the new packing station (P0) to Robot B's path when a new order came in with the same shelf (S03) and still prioritized the first order (P1) over the new one.

Case 4: New order on shelf already on the way returning

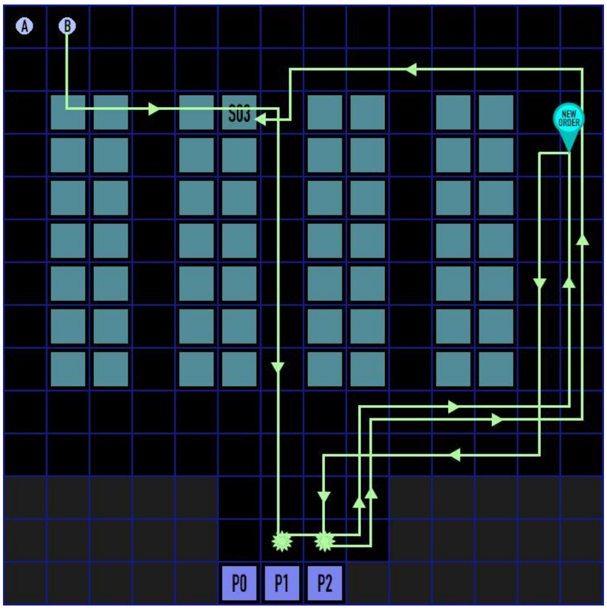


Figure 12. Task assigning simulation Case 4 result

A.N.T.s rerouted Robot B's path to the new packing station (P2) when a new order came in with the same shelf (S03).

4.2. Pathfinding Simulation

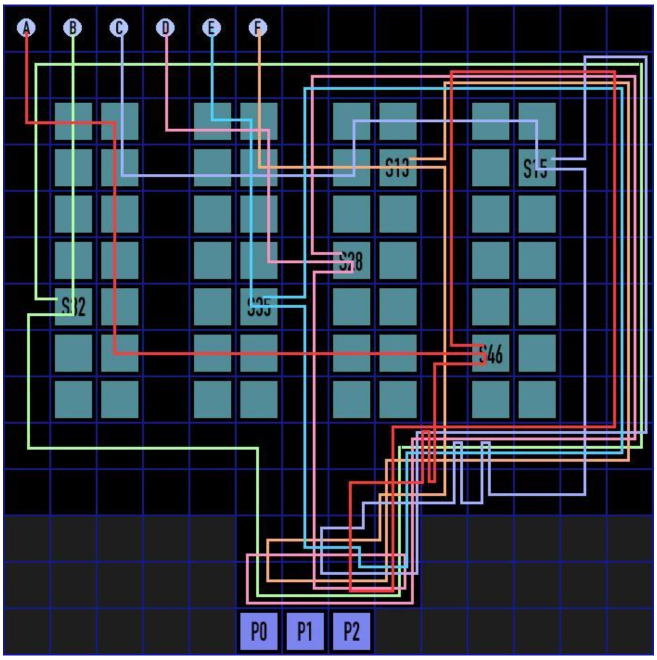


Figure 13. Pathfinding simulation Round 1 result

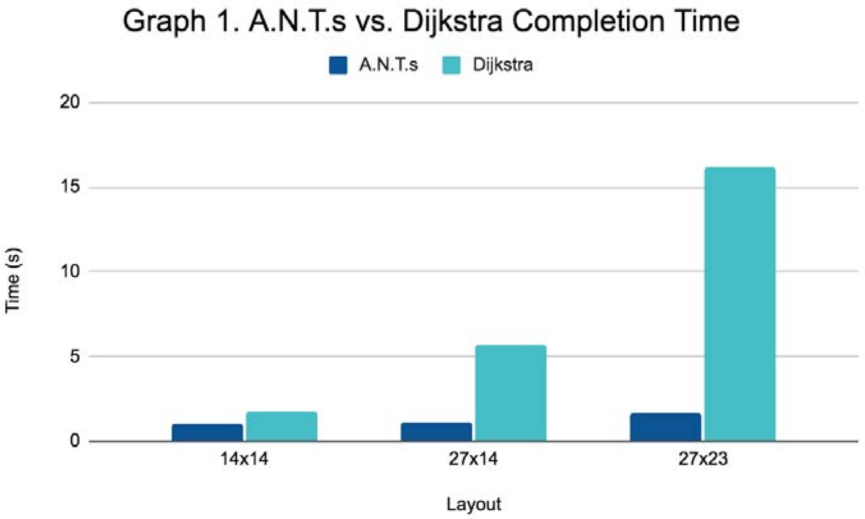
In all 10 rounds (120 orders), the A.N.T.s algorithm is able to assign the six robots paths with no collision. When the designated packing station was still occupied by another robot, the robot queued behind. When another robot is in its path, the robot would change lanes to avoid collision.

4.3. Comparing A.N.T.s to Dijkstra in Various Layouts

In each trial below, five robots were dispatched and 20 randomly generated orders were placed. Both algorithms were given the same set of orders to be compared.

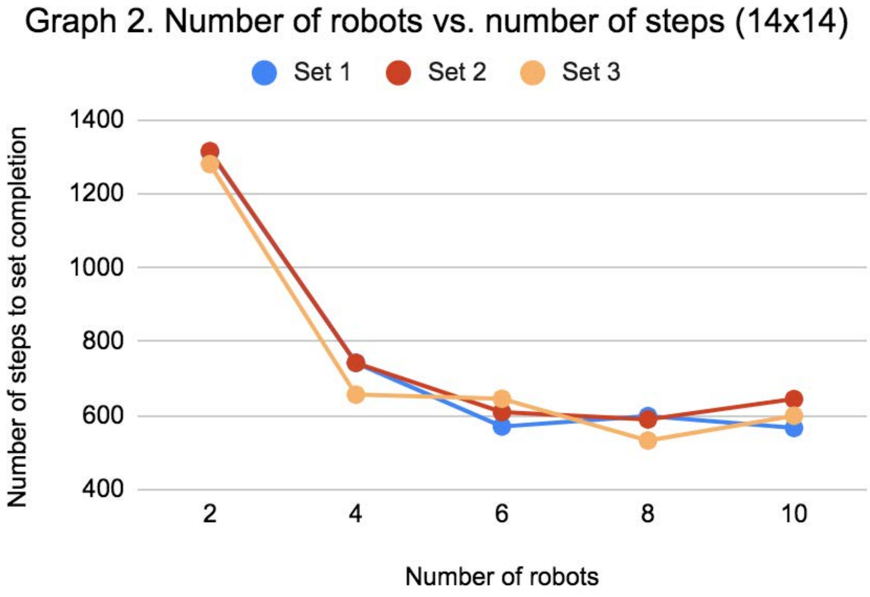
Table 3. A.N.T.s vs Dijkstra's algorithm trial results

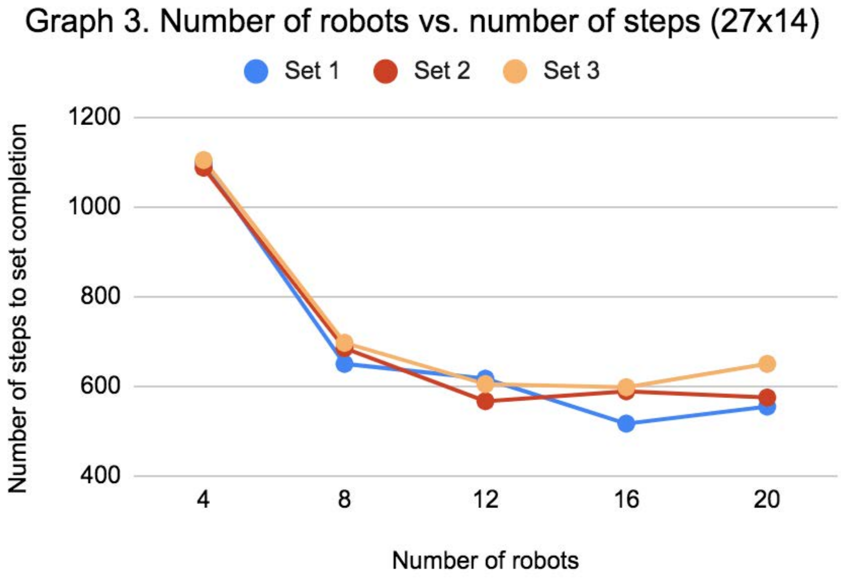
Layout	Time (s)							
	A.N.T.s				Dijkstra			
	Trial 1	Trial 2	Trial 3	Average	Trial 1	Trial 2	Trial 3	Average
14x14	0.94071	1.12735	0.87699	**0.98168**	1.64350	1.74688	1.89869	**1.76302**
27x14	1.17626	1.08328	1.10478	**1.12144**	5.85844	5.23951	5.96506	**5.68767**
27x23	1.55772	1.51237	1.81233	**1.62747**	17.96567	15.98652	14.65906	**16.20375**



Results show that A.N.T.s have shorter computation time in all three layouts. In some scenarios, Dijkstra was seen to route shorter paths for robots than A.N.T.s. However, as Dijkstra evaluates all nodes, an increase in warehouse size shows a significant increase in computation time by Dijkstra. The increases correspond to its time complexity O(V^2). A.N.T.s is seen to be 1.80 times faster in the 14x14 layout and 9.96 times faster in the 27x23 layout than Dijkstra. Therefore, A.N.T.s is shown to be more applicable for task assigning and pathfinding in warehouses as warehouse expansions would not significantly impact the performance of A.N.T.s.

4.4. Number of Robots vs. Time Simulation





Both graphs show a negative exponential trend. More robots dispatched may not necessarily decrease order set completion time. Results show that in a 14x14 and 27x14 grid warehouse layout, for 100 orders placed at the start of the simulation, a number of robots greater than 6 and 12 respectively resulted in no significant change in the total time taken. This is because the more robots dispatched, the greater the chance for robots to be in each other's path. Consequently, additional waiting time would be incurred.

5. CONCLUSION

A.N.T.s is shown to be able to help a simulated automated warehouse system handle task assigning and pathfinding for multiple robots simultaneously while avoiding deadlocks and collisions in the warehouse. As compared to Dijkstra's algorithm, A.N.T.s did not show significant increases in computation time when the warehouse layout increased in size. As seen from the simulation with varying numbers of robots, additions of robots show a negative exponential trend in completion time. The initial addition of robots was able to reduce order-picking process time significantly. However, once it reaches a certain point, an additional number of robots may not be beneficial in reducing completion time as an increase in concentration results in greater chances for robots to be in each other's paths. Consequently, additional waiting time would be incurred.

For future work, A.N.T.s would be compared to other warehouse order-picking methods for efficiency tests and would be further evaluated against other heuristic methods. In addition, robot charging stations would be added to the route.

The limitation of this research is that robots must move at constant speeds one grid at a time. Therefore if implemented in a warehouse, robot hardwares must be able to configure speed depending on the varying weight of the shelf units as well.

REFERENCES

[1] Banker, S., 2019. The Autonomous Mobile Robot Market Is Taking Off Like A Rocket Ship. [online] Forbes. Available at: <https://www.forbes.com/sites/stevebanker/2019/03/11/the-autonomous-mobile-robot-market-is-taking-off-like-a-rocket-ship/?sh=7a4000dc1603> [Accessed 9 September 2021].

[2] Bartholdi JJ, Hackman ST., 2011. Warehouse and distribution science, the supply chain and logistics institute.

[3] CNN Indonesia. 2020. Transaksi e-Commerce Naik Nyaris Dua Kali Lipat saat Pandemi. [online] Available at: <https://www.cnnindonesia.com/ekonomi/20201021193353-92-561232/transaksi-e-commerce-naik-nyaris-dua-kali-lipat-saat-pandemi> [Accessed 5 September 2021].

[4] Databridgemarketresearch.com. n.d. Warehouse Robotics Market – Global Industry Trends and Forecast to 2027 | Data Bridge Market Research. [online] Available at: <https://www.databridgemarketresearch.com/reports/global-warehouse-robotics-market> [Accessed 9 September 2021].

[5] Ferguson, D., Likhachev, M. and Stentz, A., 2005, June. A guide to heuristic-based path planning. In Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS) (pp. 9-18).

[6] Habib, H., Waseem, S. and Ghafoor, A., 2019. Development and Implementation of Enhanced Shortest Path Algorithm for Navigation of Mobile Robot (MRWA).

[7] Javaid, Adeel. 2013. Understanding Dijkstra Algorithm. SSRN Electronic Journal. 10.2139/ssrn.2340905.

[8] Kumar, N. and Kumar, C., 2018. Development of collision free path planning algorithm for warehouse mobile robot. Procedia Computer Science, 133, pp.456-463.

[9] Liu, H., Liu, W., Zhang, M. and Chen, A., 2019. Algorithm of Path Planning Based on Time Window for Multiple Mobile Robots in Warehousing System.

[10] Shaikh, E.A. and Dhale, A., 2013. AGV path planning and obstacle avoidance using Dijkstra's algorithm. International Journal of Application or Innovation in Engineering & Management (IJAIEM), 2(6), pp.77-83.

[11] Shetty, N., Sah, B. and Chung, S., 2020. Route optimization for warehouse order picking operations via vehicle routing and simulation. SN Applied Sciences, 2(2).

[12] Vällfors, L. and Hernqvist, F., 2019. Robot Collaboration Techniques in Automated Warehouses.

【評語】190047

This project designed a heuristic algorithm called A.N.T.s (Algorithm for Navigating Traffic System) to conduct task assigning and pathfinding for AMR in the automated warehouse. Comparison against the commonly used Djikstra's algorithm was also conducted (Shaikh and Dhale, 2013). Results show that the proposed A.N.T.s algorithm could execute 100 orders in a 27x23 layout with five robots 9.96 times faster than Dijkstra with no collisions. Overall, this is a very nice work. However, a very important assumption was that the robot moving at a constant speed, regardless of the payload. If differences in payload, moving speed and other environmental constraints are considered, this can be a very good project.