Journal of Robotics and Control (JRC)
Volume 6, Issue 4, 2025
ISSN: 2715-5072, DOI: 10.18196/jrc.v6i4.26660

Multiple Targets Path Planning for Document Delivering Mobile Robot in Dynamic Environments

Van-Phong Vu [1*], Đinh-Hieu Nguyen [2], Thanh-Trung Nguyen [3], Minh-Duc Tran [4]
[1, 2, 3, 4] Department of Automatic Control, Ho Chi Minh City University of Technology and Education, Ho Chi Minh City, Vietnam
Email: [1] phongvv@hcmute.edu.vn, [2] 20151369@student.hcmute.edu.vn, [3] 20124091@student.hcmute.edu.vn
*Corresponding Author

***Abstract*—This paper proposes a method to control and make path planning for a delivering mobile robot that operates in a dynamic environment. The working environment is an office building that will appear both static obstacles and dynamic obstacles (such as such as people or other mobile robots). The mobile robot is designed to carry documents to multiple targets that are determined by users. Users can call the mobile robot and input the information of the documents and targets that need to be delivered via the website. The working environment map will be established by using LiDAR and SLAM technology. The path plaining is executed in two steps. Firstly, the ant colony algorithm (ACO) is employed to solve the indoor traveling salesman problem (ITSP), the TSP for indoor application, for determining the globally optimal moving schedule to multiple targets in this paper. Then, the shortest moving path between point to points for the delivering mobile robot is determined by using the Dijkstra algorithm. The shortest moving path for the delivering mobile robot is determined by using the Dijkstra algorithm. The ant colony algorithm (ACO) is employed to solve the inner traveling salesman problem (ITSP) to determine the optimal moving schedule to multiple targets in this paper. The dynamic window approach (DWA) methodology is applied to assist mobile robots in avoiding static and dynamic obstacles. In addition, the adaptive monte Carlo localization (AMCL) is used for positioning the mobile robot on the map. Finally, the simulation in MATLAB and Gazebo environment as well as the experiments, are presented to prove the superior success of the delivering mobile robot.**

Keywords*—Mobile Robot; Inner Traveling Salesman (ITSP); Ant Colony Algorithm (ACO); Path Planning; SLAM; Dijkstra; DWA; AMCL.

I. INTRODUCTION

Recently, mobile robots have received a great deal of attention from researchers because of their wide applications in practices such as smart agriculture [1], service for elderly people [2], and in smart factories [3]. There have been many studies focused on designing and controlling mobile robots in the past few years [4]-[14]. For example, the finite-time control law combined with the Backstepping algorithm was proposed to control the wheeled mobile robot to track the predefined trajectories in paper [4]. The optimal path planning based on the improved A* has been applied for the fire-fighting mobile robot in the ware-house to obtain the shortest moving path [7]. The rescue mobile robot was designed and controlled on the basis of the SLAM technique for completing the rescue mission [8]. The navigation for the mobile robot based on Machine learning algorithm was investigated in [13]-[14].

Additionally, normally, the mobile robot will work in an unknown environment, hence, working map construction for mobile robots plays a very important role in controlling the mobile robot. There exist several methods to obtain the maps of the working environment, such as Rao-Blackwellized particle filters (RBPF) method was proposed in [15], and the method based on SLAM and LIDAR was studied in [8], [16], [17]. There exist the drawbacks of the SLAM and LiDAR such as Map corruption in crowded or dynamic environments, sensor noise, map inconsistency due to moving obstacles, or computational latency. However, SLAM and LiDAR provide a robust localization and mapping solution for real-time autonomous navigation. The combination of SLAM and LiDAR increases the accurate, drift-minimized, and adaptive navigation even in partially dynamic environments. Due to this advantage of SLAM and LiDAR, they are employed to build the maps of the working environment.

Besides, the path planning for the mobile robot to determine the shortest moving route has received attention from researchers. Many studies have concentrated on the path planning of the mobile robot in the past few years [7], [8], [12], [17]-[25]. For instance, the traditional Jump Point Search (JPS) method was proposed for the mobile robot to determine the optimal path [20]. Another method to find the shortest moving path of the mobile robot, so-called “Dijkstra Algorithm”, was investigated in [20]-[23]. The Dijkstra algorithm can select the optimal path from a given node to other nodes in the graph based on a min-priority queue data structure. The improved RRT* and the modified A* methodologies were employed to obtain the optimal path of the mobile robot in papers [24] and [25], respectively. However, these methods are applied for finding the shortest moving path of the single-target scenarios but do not scale well or lack efficiency when generalized to multi-target settings.

Furthermore, normally, the mobile robots always operate in complex environments with many obstacles. For this reason, obstacle avoidance is a pressing issue when controlling mobile robots. Many articles have been investigated for avoiding the obstacles based on the dynamics window approach (DWA) method in recent years [24], [26]-[29]. For example, in paper [27], the DWA method was applied to control the mobile robot for avoiding collision with the obstacles while moving. The improved DWA algorithm was developed for the mobile robot with a narrow channel in the paper [28], and the DWA combination with a neural network was proposed for the mobile robot to avoid dynamic obstacles [29]. Apart from obstacle avoidance, the localization for the mobile robot in the working map is also a critical task. The monte Carlo localization (MCL) and the adaptive monte Carlo localization (AMCL) were proposed in [30] and [31]-[43] to determine the position and direction of the mobile robot in the map. The AMCL method can track the pose and the direction of the mobile robot by employing the particle filter and probabilistic localization system.

It should be noted that, up to now, there exist many studies paying attention to implementing and controlling the mobile robot. However, based on our best knowledge, almost studies focus on finding the optimal path between point to point, and there are very few works dealing with path planning for multiple targets. In this paper, the document-delivering mobile robot is designed and controlled to deliver the document to multiple places. It means that the mobile robot has to travel to the multiple goals one time with the shortest moving path. To deal with this requirement, the indoor travelling salesman problem (ITSP) principle will be applied to find the optimal routine of the document delivering mobile robot. To find the solution of ITSP, the Ant Colony Optimization (ACO) [44] has been applied. ITSP refers to a TSP for indoor applications, where a mobile robot must determine the optimal visiting order of a set of delivery waypoints confined within a bounded operational space (e.g., a single floor of a building). The ACO algorithm has been studied in plenty of articles to find the best routine of ITSP [45]-[57]. In this work, the ACO algorithm combined with the Dijkstra algorithm is also applied for determining the best route with the shortest moving distance.

It is noted that in this paper, the considered scenario is that the mobile robot is requested to travel to multiple targets and deliver exactly documents. Thus, the methods: Jump Point Search (JPS) method [19], Improved A*[24], Dijkstra [20]-[23], and RRT*[25] is difficult to apply for finding the optimal moving path with multiple targets. Due this reason, in this work, the combination between ACO and Dijkstra algorithms is employed to determine the optimal moving path of the document delivering mobile robot in which The ACO is employed for determining the global optimal moving schedule (order of the targets that the mobile robot need to travel with the shortest movement), the Dijkstra's algorithm is used for finding the local optimal path planning (the shortest traveling path between point to point);

With the above-mentioned discussion, the main contributions of this paper are emphasized in the following aspects.

- A mobile robot is designed and implemented to carry documents and deliver them to the exact address. The user can call the delivering mobile robot to an arbitrary room, then put the documents in the stack with a lock and input the information of the documents, including: the name of the document, the stack number that contains the document, the delivering address, and the receiver's email address. When the delivering mobile robot arrives, the receiver has to input the password that was sent via email to open the lock and pick up the correct documents. The delivering mobile robot will assist in reducing the delivery time, increasing the delivery accuracy, and ensuring the safety of the documents.
- The working map is established by applying SLAM technology and signals from Lidar. The adaptive monte Carlo localization (AMCL) is used to localize the pose and direction of the mobile robot in the map. In addition, the DWA algorithm is also employed for both static and dynamic obstacle avoidance.
- The delivering mobile robot has to travel to multiple targets for delivering documents, hence, the ACO and Dijkstra algorithms are applied for finding the best route with the shortest moving road.

The rest of this paper is organized as follows. The system model and problem description are presented in section II. The proposed method, including map construction, AMCL, DWA, ACO, and Dijkstra's algorithm, is shown in section III. Section IV shows the simulation and experimental results. Finally, the conclusion is drawn in section V.

II. System Model and Problem Description

A. System Model

The document-delivering mobile robot is designed and implemented and shown in Fig. 1. The whole mobile robot is controlled by a mini IPC, and the control algorithm is embedded in an Arduino Mega 3 board that connects to the mini IPC via UART protocol. A user interface is designed on the Tablet that allows the user to interact with the mobile robot.

The dynamic model of the mobile robot is described in Fig. 2, and the mathematical model of the mobile robot is represented in (1).

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{1}$$

where $q = [x \quad y \quad \theta]^T$, (x, y) is the position of the Mobile Robot, θ is the rotation angle of the mobile robot. $v = \frac{v_R + v_L}{2}$ is the velocity of the mobile robot, v_R and v_L are the velocities of the left and right wheels, respectively. ω is the speed angle of the mobile robot.

The updated position and direction of the mobile robot at $t + \Delta t$ is calculated in the following formula:

$$\begin{bmatrix} x(t+\Delta t) \\ y(t+\Delta t) \\ \theta(t+\Delta t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} + \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \Delta t \tag{2}$$

in which

$$x(t) = \int_0^t v(t)\cos[\theta(t)]\,dt \tag{3}$$

$$y(t) = \int_0^t v(t)\, sin[\theta(t)]\, dt \tag{4}$$

$$\theta(t) = \int_0^t \omega(t)\, dt \tag{5}$$

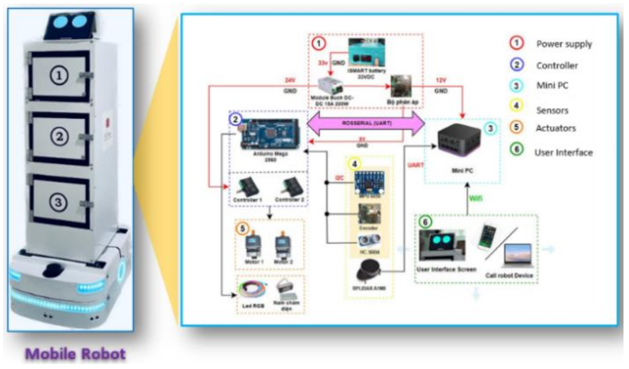


Fig. 1. Structure of the delivering mobile robot

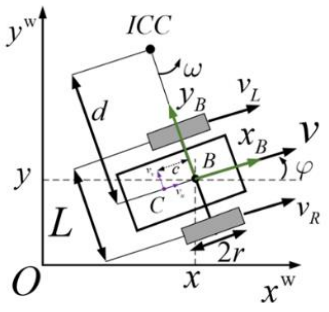


Fig. 2. The dynamics model of the mobile robot

B. Problem Description

Suppose that the documents need to be delivered to several rooms. Currently, the documents are almost delivered to the rooms by the staff. Unfortunately, delivering documents by hand is time-consuming, and sometimes the document is delivered to the incorrect places. In order to overcome this difficulty, in this paper, a mobile robot is designed for delivering the document to multiple places, automatically and exactly. First of all, the mobile robot will create the map of the working environment by using SLAM technology and RPLIDAR. When users want to use the delivering mobile robot, the call is executed by accessing a website and inputting the information of the document and the receiver's information. Then the mobile robot will transport the documents to the rooms that are assigned by users. The path planning method based on Dijkstra's algorithm is applied to determine the shortest path to travel from the targets. During traveling, the mobile robot will encounter not only static obstacles but also dynamic obstacles. Hence, to avoid these obstacles, the DWA algorithm is employed to control the mobile robot. In addition, in case the documents need to be delivered to many rooms, the mobile robot has to travel to each room once. The arising question is how to determine the moving order of the rooms with the shortest traveling distance, that is the Inner Traveling Salesman Problem. To deal with this issue, the ACO will be applied to solve the ITSP for determining the optimal path of the mobile robot. When the mobile robot arrives at the target, for the sake of security reasons, receivers have to enter the password that has been sent via email to open the stack and pick up the document. The operating procedure of the document-delivering mobile robot is illustrated in Fig. 3.

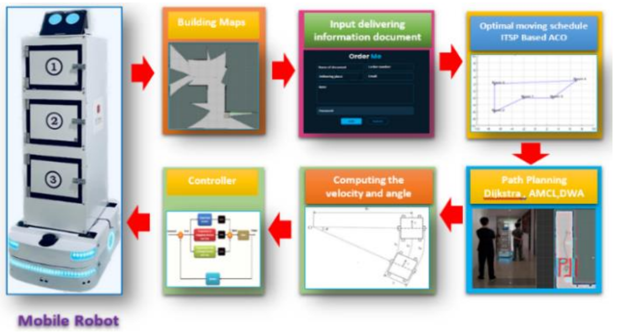

Mobile Robot
Building Maps
Input delivering information document
Optimal moving schedule ITSP Based ACO
Controller
Computing the velocity and angle
Path Planning Dijkstra_AMCL_DWA

Fig. 3. The working procedure of the document delivering mobile robot

III. Proposed Method

A. Constructing the Map of the Working Environment

In this paper, a 2D map working environment will be constructed for a mobile robot to be able to move and transport the document automatically in the working environment. To create the working map, the *slam_gmapping pakeage* of *ROS* is employed to collect the data from Lidar Rplidar A1M8, encoder, IMU sensor then carry out the construction of the Maps. The scanned data from RPLIDAR is exported under the ROS topics, usually topic /scan and the nodes of ROS can subscribe to this topic to process scanned data. The procedure for creating the maps is presented in Fig. 4 and the obtained map is shown in Fig. 5.

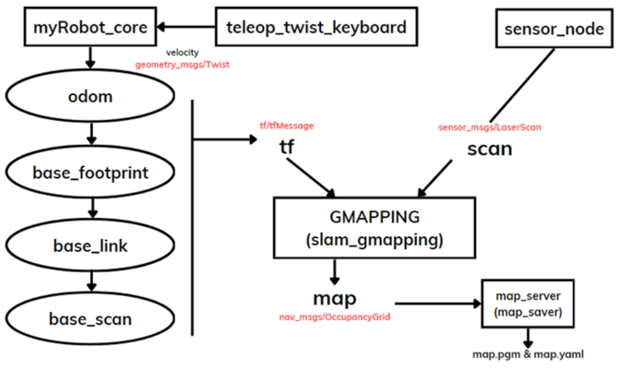


Fig. 4. Mapping construction by Slam-gmapping package of ROS

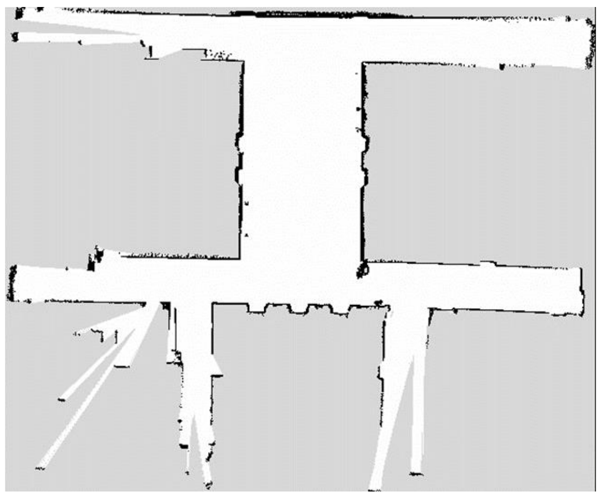


Fig. 5. The obtained working map

Remark 1: The SLAM technology is applied to construct the working map. However, there exist the drawbacks of SLAM technology such as it does not inherently handle moving obstacles or map drift. To overcome these challenges, the data Lidar Rplidar A1M8 is used to distinguish between static and dynamic obstacles during operation.

B. Dijkstra Algorithm

Dijkstra's algorithm is applied to the Occupancy grid map by traversing all grid cells, starting from the mobile robot's initial position. The algorithm iteratively considers unvisited neighboring grid cells and adds them to the list of considered paths. This search will be executed on the entire map from the initial position until the desired target position is reached. Dijkstra's algorithm guarantees to find the shortest path on the occupancy grid map from the initial position to the target position.

Let us define the initial position and target position to be s and t, respectively. Q is the optimal set of notes. The total of notes in the diagram is N. n is the number of notes; $1 \leq n \leq N$. The distance from the initial notes to the p note is computed in the following.

$$d = \sum_i^n d_{i,i+1}\,;\; i \in Q \qquad (6)$$

where $d_{i,i+1}$ is the distance from the note *i* to the adjacent *i+1*. Dijkstra's algorithm is presented as follows.

```
Dijkstra Algorithm
1: Algorithm Dijkstra ($G, s, t$):
2: Input:  Diagraph $G$, initial position and target position ($s$, $t$)
3: Output: The shortest path from s to t and its distance
4: Initialization:
   • Assign the distance from current c to s to zero; distance from current c to other note v to infinite: d(v)← ∞
   • Initialize the optimal path set of notes to the initial position s: Q = {s}.
while the target t has not been reached
5: Update the distance:
   • The distance from current note c to other adjacent notes is updated by:
       $d(v) = \min\,(d(v), d(c) + W(c,v))$
     In which $W(c,v)$ is the weight of edge $(c,v)$
6: Assign the note that has the shortest distance to be the current note.
7: Put this note into the optimal set of note Q
   // Repeat the steps 5-7 until the target t is reached.
 Return the shortest distance  $d(v)$
```

C. Adaptive Monte Carlo Localization Algorithm

Monte Carlo localization (MCL) [31] is a popular method to estimate the position and orientation of the mobile robot in a given map. In this work, the state of document delivering mobile robot is $x_t = (x, y, \theta)$ that includes position (x, y) and the orientation θ. In the MCL algorithm, each possible state of the mobile robot is represented by a particle filter. The estimated state of the mobile robot at time *t* will be based on the previous state $\boldsymbol{x_{t-1}}$, actuation command $\boldsymbol{u_t}$, and the information from sensors $\boldsymbol{z_t}$. The MCL is successful in solving both the local and global localization problems. Unfortunately, this method fails to localize the mobile robot when the robot kidnapping, or global localization failures. To deal with this challenge, an adaptive monte Carlo localization (AMCL) is improved by adding random particles to the particle sets. The algorithm of the AMCL is shown as follows.

```
Algorithm Adaptive_MCL [30]
1: Algorithm Adaptive_MCL ($X_{t-1}, u_t, z_t, m$):
2:      Static $w_{slow}, w_{fast}$
3:      $\overline{X_t} = X_t = \emptyset$
4:      for $m = 1$ to $M$ do
5:          $x_t^{[m]}$= sample_motion_model($u_t, x_{t-1}^{[m]}$)
6:          $w_t^{[m]}$= measurement_model($z_t, x_t^{[m]}, m$)
7:          $\overline{X_t} = \overline{X_t} + \langle x_t^{[m]}, w_t^{[m]} \rangle$
8:          $w_{avg} = w_{avg} + \frac{1}{M} w_t^{[m]}$
9:      end for
10:     $w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow})$
11:     $w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast})$
12:     for $m = 1$ to $M$ do
13:     with probability max(0.0,1.0 $- w_{fast}/w_{slow}$) do
14:     add a random pose to $X_t$
15:     else
16:     draw $i$ with probability ∞ $w_t^{[i]}$
17:     add $x_t^{[i]}$ to $X_t$
18:     end with
19:     end for
20:     return $X_t$
```

D. Dynamic Window Approach

Dynamic Window Approach (DWA) [25] is the method for online obstacle avoidance of the mobile robot (Fig. 6). DWA consists of two main steps:

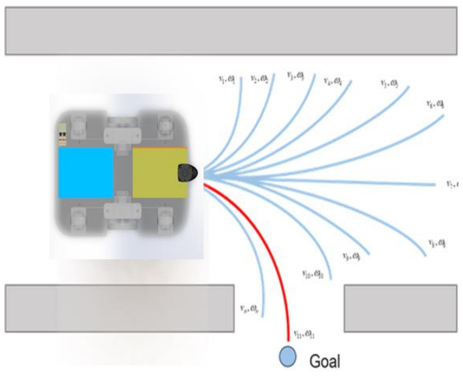


Fig. 6. Dynamics window approach algorithm

- Step 1: Generating the valid search space of velocity. In this step, the possible velocities of the robot are determined based on kinetic constraints, i.e., the minimum and maximum range of velocities that the mobile robot can achieve in a short period of time. On the basis of the data obtained from LiDAR, the velocities that cause to collisions with obstacles in the working environment are eliminated. As a result, the search space for finding reasonable velocities, including safe and achievable velocities, is identified.

- Step 2: Determining the optimal velocity in this valid search space. The feasible velocities will be evaluated based on safety to avoid collisions, the fastest speed to approach the target, and stability of the mobile robot movement.

The valid search space is determined by following procedures:

- *Circular trajectories*: Circular trajectories are determined by pairs of translational velocity (v) and rotational speed (ω).
- *Admissible velocities*: A pair of velocities (v, ω) is feasible if the mobile robot can stop before colliding with an obstacle. It means that the mobile robot can avoid the obstacles and stop safely in the available space (Fig. 7). The admissible velocities are calculated by

$$V_a = \left\{v, \omega \middle| v \leq \sqrt{2 \cdot dist(v,\omega) \cdot \dot{v}_b} \wedge \omega \leq \sqrt{2 \cdot dist(v,\omega) \cdot \dot{\omega}_b}\right\} \quad (7)$$

in which V_a admissible velocity**,** $dist(v, \omega)$ is the smallest distance between the mobile robot and the obstacle without collision. $\dot{v}_b, \dot{\omega}_b$ are the maximum translational and rotational accelerators of the mobile robot that will cause to collision with obstacles.

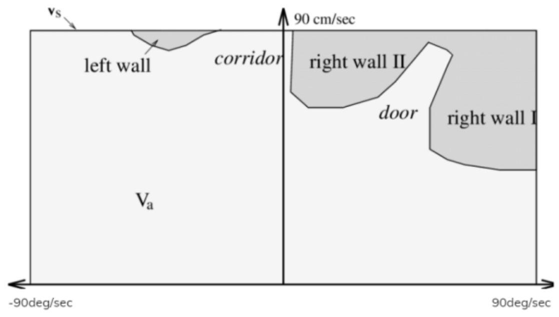


Fig. 7. Admissible velocity [25]

- *Dynamic window****:*** Dynamic windows limit the admissible velocities so that the robot can achieve in a short period of time, with the robot's limited acceleration (Fig. 8).

$$V_d = \{(v, \omega)| v \in [v_a - \dot{v} \cdot t, v_a + \dot{v} \cdot t] \wedge \omega[\omega_a - \dot{\omega} \cdot t, \omega_a + \dot{\omega} \cdot t]\} \quad (8)$$

And then the search space is determined by

$$V_r = V_s \cap V_a \cap V_d \quad (9)$$

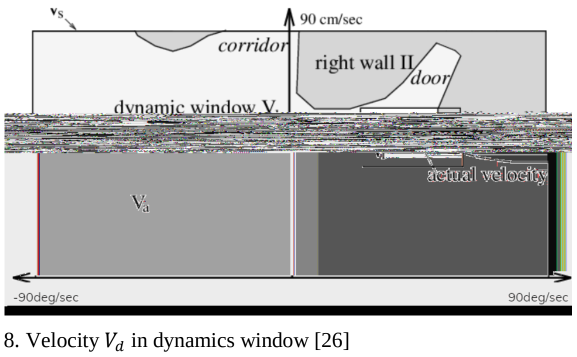


Fig. 8. Velocity V_d in dynamics window [26]

- *Optimization****:*** The objective function

$$G(v, \omega) = \sigma\big(\alpha \cdot heading(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot velocity(v, \omega)\big) \quad (10)$$

where $heading(v, \omega) = 180^o - \theta$ is the angle between the robot's current direction of movement and the direction of the target point that the robot is trying to aim for. $dist(v, \omega)$: is the distance to the nearest obstacle that cuts through the curve. If no obstacles exist on the curve, this value will be set as a large constant. $velocity(v, \omega)$: is the moving forward velocity of the mobile robot and supports fast movements.

E. ITSP-based Ant Colony Optimization

The Inner Travelling Salesman Problem is a famous and conventional issue that needs to determine the optimal moving schedule to travel a list of cities. To find the solution of ITSP, in this article, the Ant Colony Optimization methodology [34] is employed. The Ant Colony Optimization (ACO) algorithm is an optimization method based on the foraging behavior of the ant colonies in the wild (see Fig. 9). ACO has many advantages when applied to optimization problems, especially those with a graphical structure such as TSP (Traveling Salesman Problem). The ACO algorithm can be easily extended and applied to a variety of optimal problem types just by changing the representation and updating the pheromone trace. The probability of path selection of each ant is computed in (11).

$$p_{ij}^k = \frac{(\tau_{ij})^\alpha (\eta)^\beta}{\sum_{l \in N_i^k} (\tau_{il})^\alpha (\eta_{il})^\beta}, j \in N_i^k \quad (11)$$

where p_{ij}^k is the probability of the kth ant selecting the path(i, j), $\tau_{ij}$is the level of pheromone of path (i, j), η_{ij} is the heuristic information, N_i^k is the adjacent station of node *i* that the kth ant has not been passed through. α and β are parameters to adjust $\tau_{ij}$and η_{ij}, respectively.

The paths that the ants have traveled will have stronger pheromone levels than the paths that the ants have not traveled. Therefore, the level of pheromone will be updated in (12):

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} + \Delta\tau_{ij}^k \quad (12)$$

In which ρ the evaporation factor of pheromone, c^k is the moving distance of the ant, $\Delta\tau_{ij}^k = \frac{1}{c^k}$, If the ant travels to the path (i, j), otherwise $\Delta\tau_{ij} = 0$. The ACO algorithm for finding the optimal solution of ITSP is presented as follows.

```
ACO algorithm for Solving ITSP
1: Define number of iterations, number of ant, and initialize parameters $\alpha, \beta, \rho$
2: For a number of iterations
3:   Initialize the start city for each ant.
4:   Construct solutions of ant: find a tour of each ant based on a probability formula (7) to select the next city.
8:   Update the pheromone level based on formula (8).
9:   Evaluate the best solution.
10: End for
```

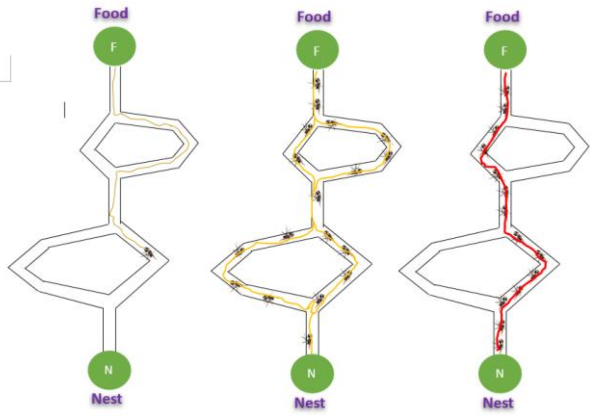


Fig. 9. Ant colony optimization principle

F. User Interface

To communicate between users and document delivering mobile robot, a user interface is built under a website. Additionally, an app is also built and deployed on a tablet to help the receiver get the right document. Suppose that the document delivering mobile robot is located at arbitrary position, user will access the user interface call the document delivering mobile robot (see Fig. 10). When the mobile robot goes to the user position, user will put the documents into the stacks and input information of document such as: name of document, stack number, receiver's email and room number that need to deliver the document (see Fig. 11). To ensure the document is delivered exactly, the information including password of the document will be sent to receiver (see Fig. 12). When mobile robot approach to the target, the user has to input the password to open the stack and receive the document (see Fig. 13).

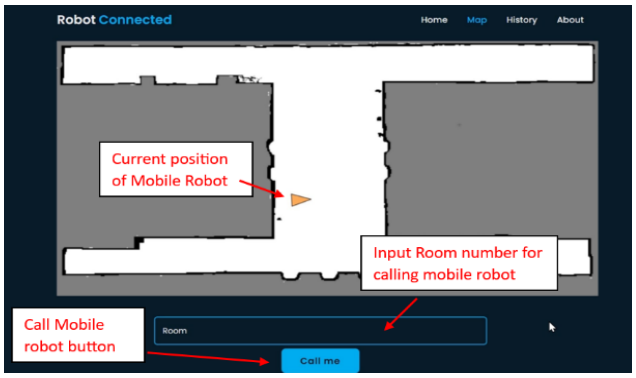

Robot Connected
Home
Map
History
About
Current position of Mobile Robot
Input Room number for calling mobile robot
Call Mobile robot button
Room
Call me

Fig. 10. Calling mobile robot interface

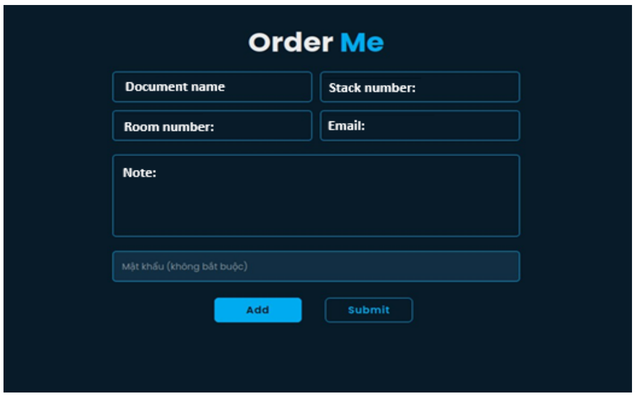
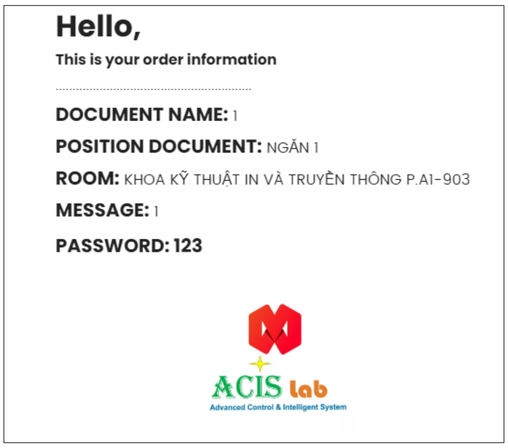


Fig. 11. Input document information interface



Hello,

This is your order information

DOCUMENT NAME: 1

POSITION DOCUMENT: NGĂN 1

ROOM: KHOA KỸ THUẬT IN VÀ TRUYỀN THÔNG P.A1-903

MESSAGE: 1

PASSWORD: 123

ACIS lab Advanced Control & Intelligent System

Fig. 12. Sending massage to receiver via email

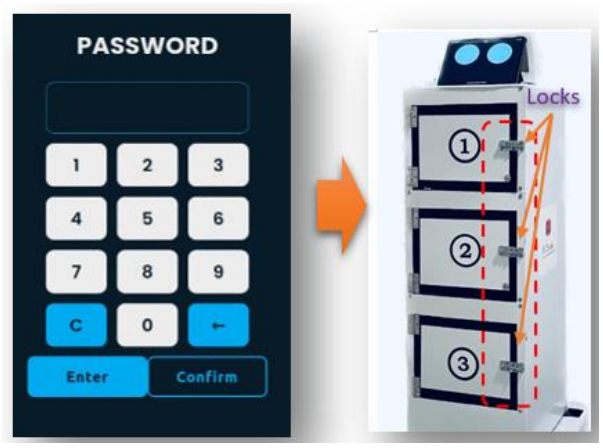


Fig. 13. Input password to open stack and receive document

IV. Simulation and Experiment Results

A. Simulation Results

1) Building the Maps in Gazebo

In this section, the Gazebo software is employed to simulate the SLAM technique to obtain the working map. The layout is created in Gazebo, which is shown in Fig. 14, and the mapping process is illustrated in Fig. 15. After complete scanning, the working map is presented in Fig. 16.

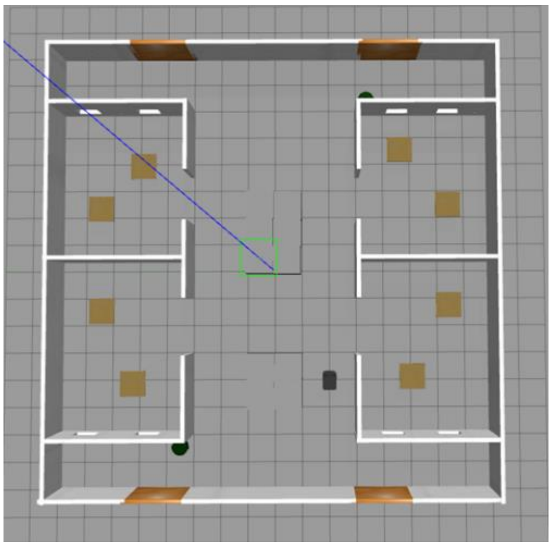


Fig. 14. Layout in Gazebo

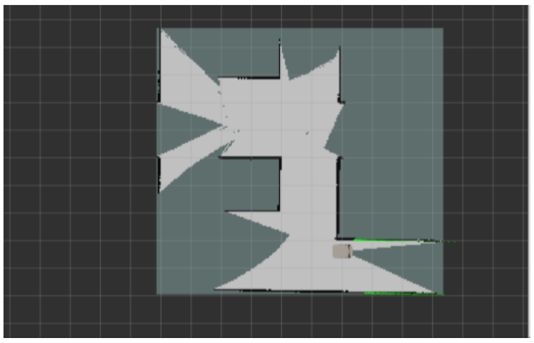


Fig. 15. Map scanning

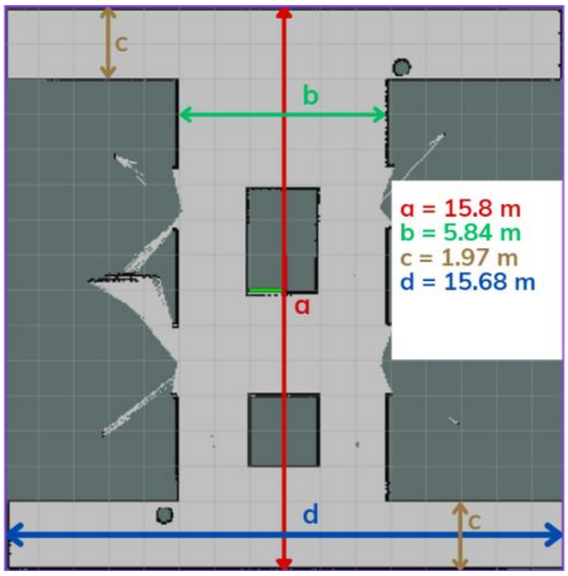


Fig. 16. Obtained map in Gezabo

2) *Simulation ACO Algorithm for ITSP*

In this section, the ACO algorithm is applied to obtain the optimal moving path for the document delivering mobile robot. The algorithm is simulated in MATLAB with the parameters illustrated in Table I.

TABLE I. PARAMETERS OF ACO ALGORITHM

Parameters	Descriptions	Value
m_{iter}	Number of iterations	100
m	Number of ants	100
α	The adjustment coefficient for the effect of τ_{ij}	1
β	The adjustment coefficient for the effect of η_{ij}	3
ρ	The pheromone evaporation rate	0.1

a) *Simulation in MATLAB*

In this part, there five cases will be simulated in MATLAB. Case 1 will simulate for 5 rooms; case 2 will simulate 10 rooms; case 3 will simulate for 15 rooms; case 4 will simulate for 20 rooms; and case 5 will simulate for 30 rooms that need to deliver the documents. The ACO is employed to find the optimal moving path. The genetic algorithm (GA) [58] is also simulated for the perpose of comparison with ACO method.

- Case 1: The positions of the rooms are shown in Table II. Applying the ACO, the optimal moving path is "Room 1→ Room 2→Room 3→Room 4→Room 5→ Room 1" that results in Fig. 16 with the shortest distance is 3.841 m.
- Case 2: Simulation with 10 targets
- Case 3: Simulation with 15 targets
- Case 4: Simulation with 20 targets
- Case 5: Simulation with 30 targets

The simulation results are shown in Fig. 17 to Fig. 26. To prove the effectiveness of the proposed method, the ACO algorithm is compared to the GA [58] for determining the optimal path. The moving path obtained by GA algorithm are shown in Fig. 17, Fig. 19, Fig. 21, Fig. 23, Fig. 25 and the moving path determined by ACO algorithm are demonstrated in Fig. 18, Fig. 20, Fig. 22, Fig. 24, Fig. 26. The traveling cost of ACO and GA algorithms with 5, 10, 15, 20, and 30 targets are provided in Table III. From the simulation results in Fig. 17 to Fig. 26, and data in Table III, it is obvious that with 5 targets, two methods obtained the same traveling cost, however, when the number of targets increase to 10, 15, 20 and 30, the traveling cost of ACO method is shorter than GA algorithm. Hence, it can conclude that the ACO method employed in this work is better than GA algorithm for finding the optimal moving schedule.

TABLE II. POSITION F THE ROOMS

Rooms	*x (m)*	*y (m)*	*z (m)*
Room 1	-2.0	-2.0	0.0
Room 2	-7.0	-6.0	0.0
Room 3	-7.0	2.0	0.0
Room 4	7.0	3.0	0.0
Room 5	3.0	-2.0	0.0

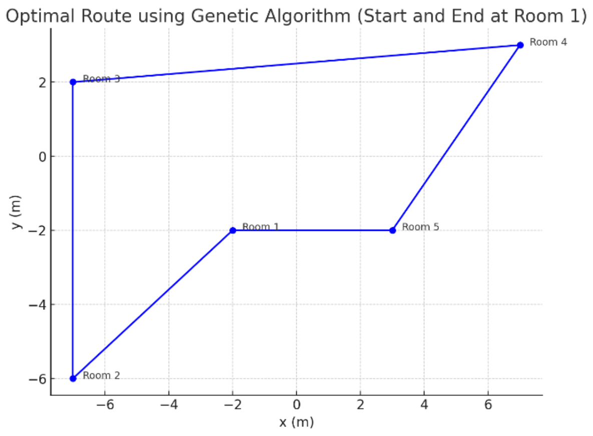


Fig. 17. Optimal path planning calculated by the GA algorithm

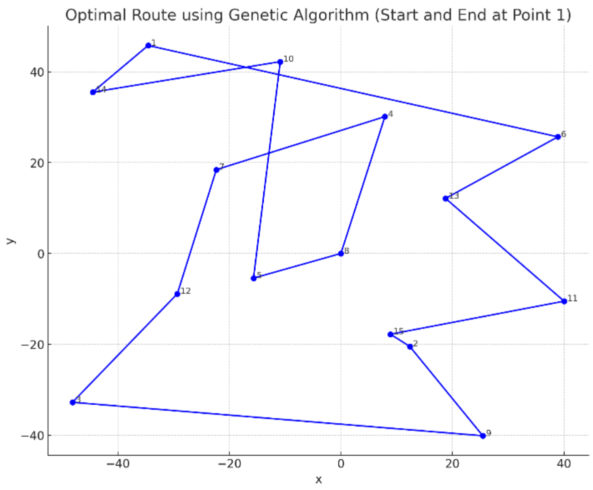


Fig. 18. Optimal path planning calculated by the ACO algorithm

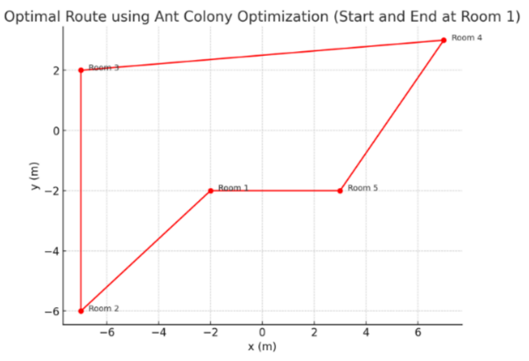


Fig. 19. Optimal Path Planning calculated by the GA algorithm-10 targets

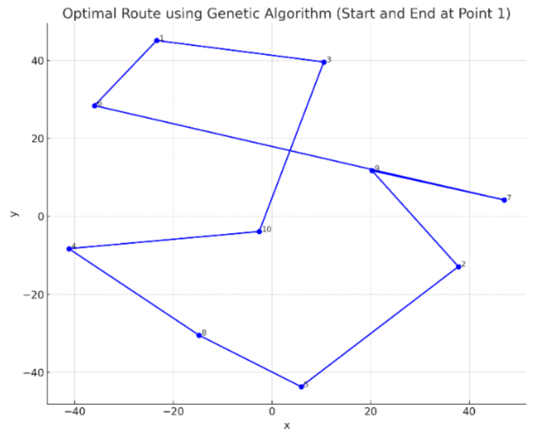


Fig. 20. Optimal Path Planning calculated by the ACO algorithm-10 targets

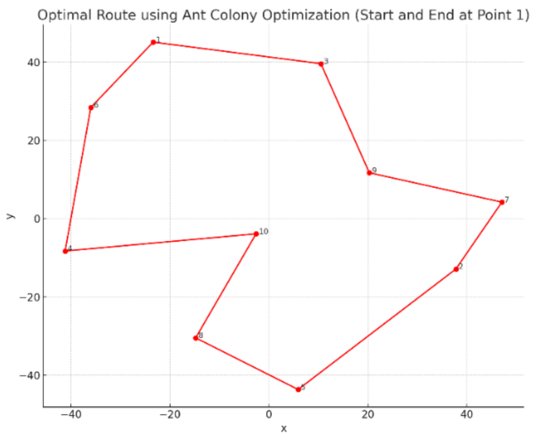


Fig. 21. Optimal Path Planning calculated by the GA algorithm-15 targets

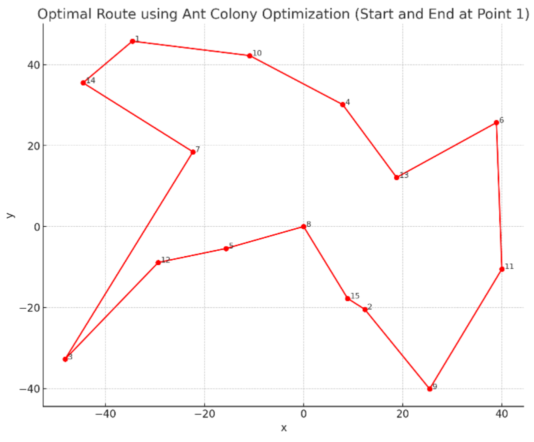


Fig. 22. Optimal Path Planning calculated by the ACO algorithm-15 targets

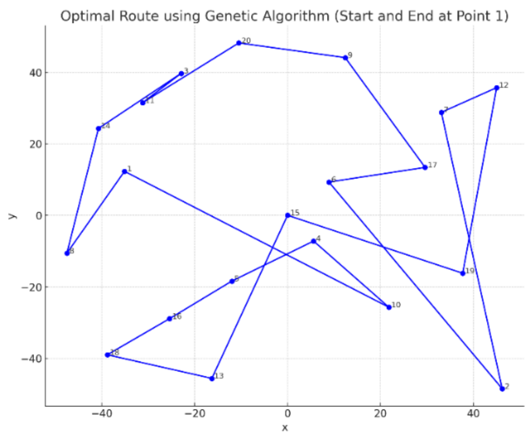


Fig. 23. Optimal Path Planning calculated by the GA algorithm-20 targets

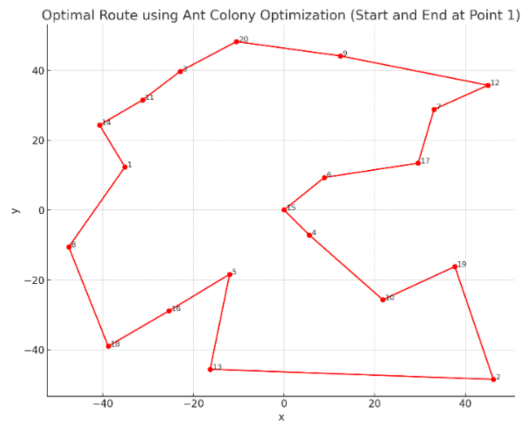


Fig. 24. Optimal Path Planning calculated by the ACO algorithm-20 targets

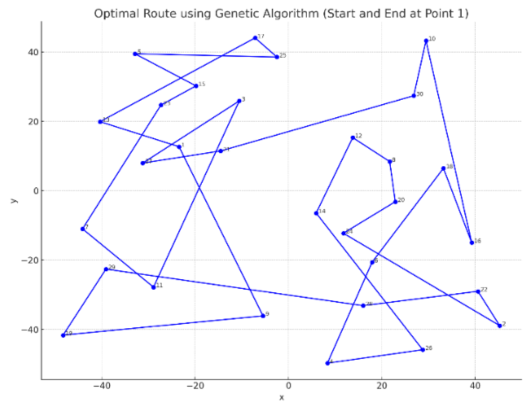


Fig. 25. Optimal Path Planning calculated by the GA algorithm-30 targets

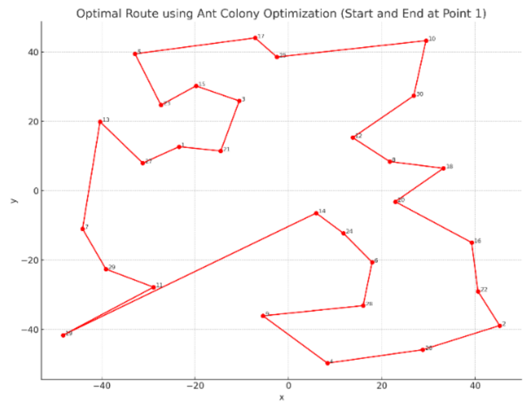


Fig. 26. Optimal Path Planning calculated by the ACO algorithm-30 targets

TABLE III. COMPARISON BETWEEN ACO AND GA

Number of targets	Traveling cost- GA	Traveling Cost - ACO
5	39.84	39.84
10	387.5	306.2
15	500.56	369.32
20	676.8	437.6
30	866.25	519.19

b) Simulation in Gazebo

In this section, the ACO algorithm is also simulated in the Gazebo software. The layout is shown in Fig. 27, and the positions of the rooms are illustrated in Table IV. After employing the ACO algorithm, the optimal path is determined by following order: Home→ Room 2→ Room 1→Room 7→ Room 4→ Room 3→Home that can be seen in Fig. 28 and Fig. 29 demonstrates that mobile robot can follow the optimal path that is computed by ACO. From simulation results in Fig. 29, it is seen that in the Gazebo simulation environment, the delivering mobile robot can move exactly to the rooms in order according to the optimal path.

TABLE IV. POSITION OF THE ROOMS IN GAZEBO

Rooms	*x (m)*	*y (m)*	*z (m)*
Home	-4.0	-2.0	0.0
Room 1	-7.0	-6.0	0.0
Room 2	-2.0	-2.0	0.0
Room 3	3.0	-2.0	0.0
Room 4	7.0	3.0	0.0
Room 5	3.0	2.0	0.0
Room 6	-2.0	2.0	0.0
Room 7	-7.0	2.0	0.0

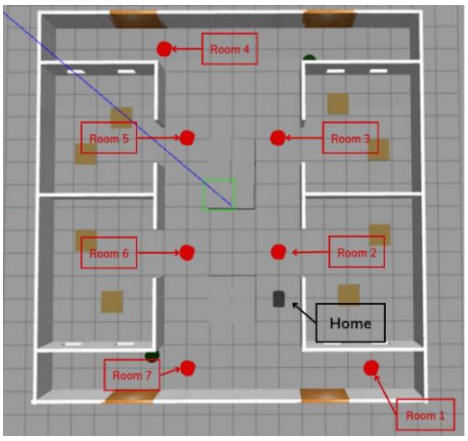


Fig. 27. Layout in Gazebo

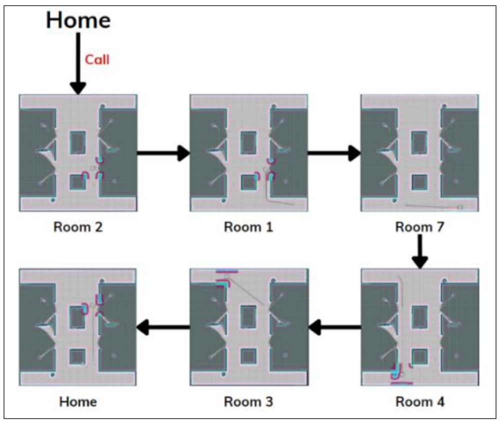


Fig. 28. Optimal moving schedule of mobile robot in Gezabo

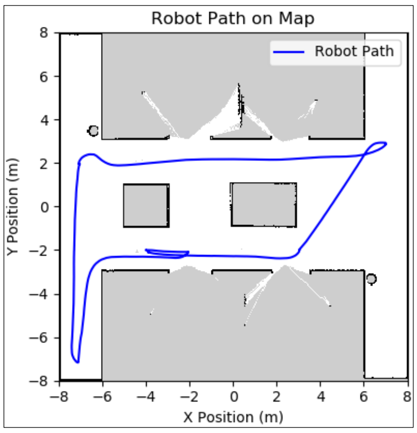


Fig. 29. Mobile robot moving path in Gezabo

B. Experimental Results

1) Control Document Delivering Mobile Robot

In this paper, the PID control is employed to control the mobile robots. Two PID controllers are designed to control the left and right wheels of the mobile robot. The parameters of the controllers are determined by the trial-and-error method as follows:

$$\begin{cases} K_{P(right)} = 1.28 \\ K_{I(right)} = 3.73 \end{cases}, \quad \begin{cases} K_{P(left)} = 1.11 \\ K_{I(left)} = 3.77 \end{cases}$$

To prove the success of the PID controller, there are 5 different velocities have been tested and shown in Fig. 30 and Fig. 31. From these experimental results, it is obvious that the speeds of both left and right wheels can track the reference velocities.

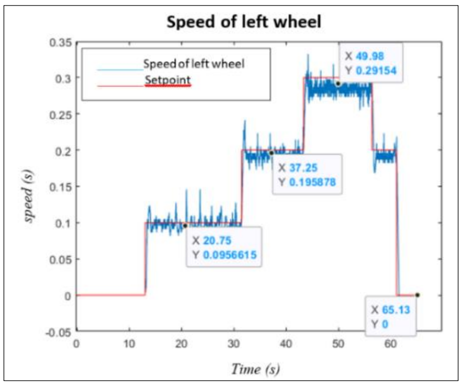


Fig. 30. Speed of left wheel

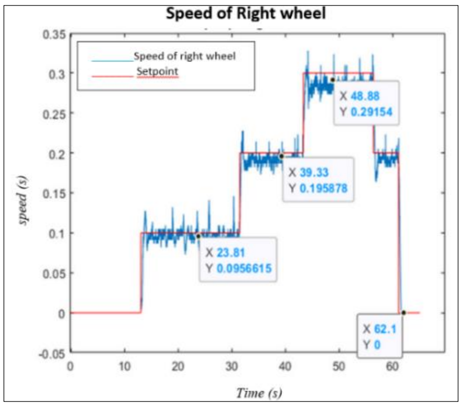


Fig. 31. Speed of right wheel

2) *Map Construction*

In this paper, a map of 9th floor in the main building of Ho Chi Minh City University of Technology and Education is constructed (see the Fig. 32). Mobile robot scans the working map via LiRDA in Fig. 33 and the obtained map is presented in Fig. 34. From the results in Table V, it is easily seen that the obtained map has similar size with real maps and the accuracy is more than 98%.

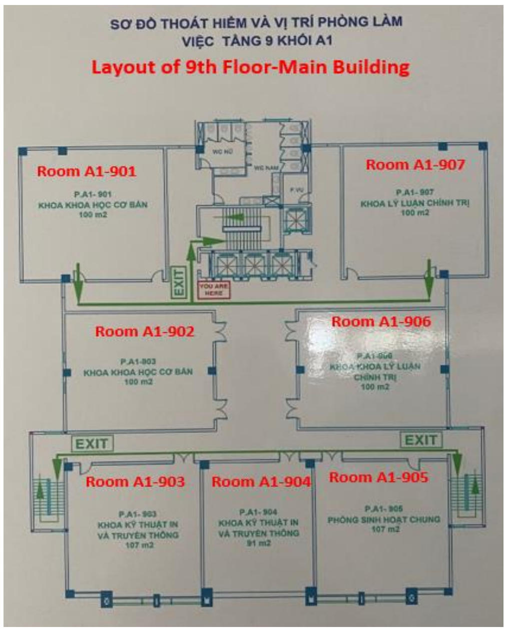


Fig. 32. Map of 9th floor in the main building

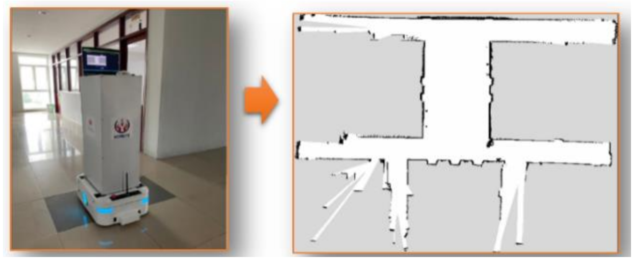


Fig. 33. Scanning the working environment map of 9th floor

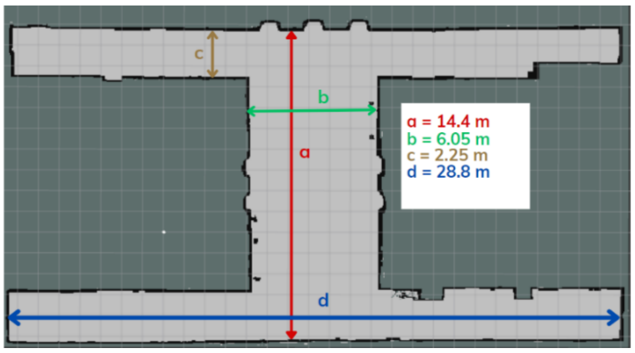


Fig. 34. Obtained working environment map of 9th floor

TABLE V. THE OBTAINED LENGTH OF THE MAP AND ACCURACY

	Values in the obtained map (m)	Real values (m)	Accuracy (%)
a	14.4	14.2	98.61
b	6.05	6	99.2
c	2.25	2.23	99.1
d	28.8	28.4	98.61

3) *Obstacle Avoidance*

For obstacle avoidance, the DWA is applied to control the delivering mobile robot. The experiments are carried out for both static and dynamic obstacles. The experimental results are illustrated in Fig. 35 and Fig. 36. In Fig. 35, two static obstacles located on the moving path of the mobile robot, and it is seen that the mobile robot can smoothly avoid these two obstacles. The moving obstacle is considered in the experiment (see Fig. 36). It is obvious that the delivering mobile robot can avoid the dynamic obstacle smoothly.

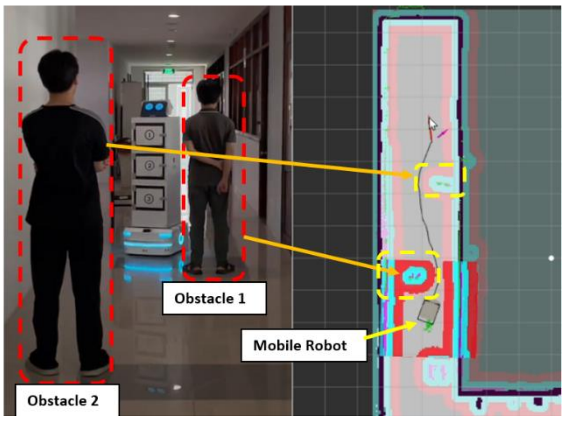


Fig. 35. Experiment for static obstacle avoidance

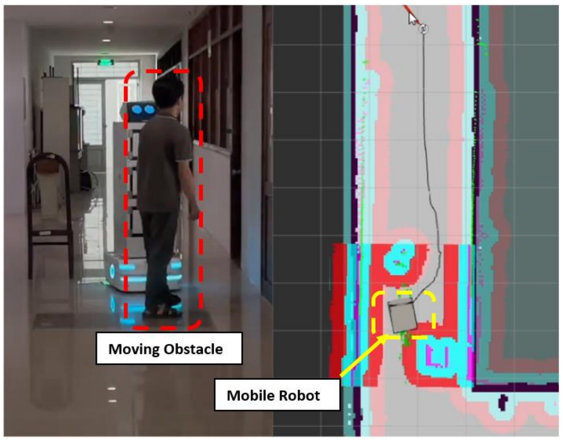


Fig. 36. Experiment for dynamic obstacle avoidance

4) *ACO Algorithm for ITSP*

In this section, ACO algorithm is employed to find the optimal path for the mobile robot in 9th floor. The 9th floor of the main building has seven rooms from Room A1-901 to Room A1-907 that have the positions in Table VI. Suppose that the documents need to deliver to the Room A1-901, Room A1-902, Room A1-903, and Room A1-906. After solving ACO in MALTAB, the optimal path is shown in Fig. 37 and the order of moving room is shown in Fig. 38.

The experimental results are shown in Fig. 39 to Fig. 44. From these results, it is seen that the ACO algorithm can determine the shortest moving schedule for the delivering mobile robot, and the mobile robot can follow the optimal schedule exactly in order from Initial position ➔ Room A1-901➔ Room A1-902➔ Room A1-903➔ Room A1-906➔ Initial position (see Fig. 44). The experimental results can be seen in the video via the following link: *https://www.youtube.com/watch?v=FNIPWwnnlPM.*

TABLE VI. POSITION OF THE ROOM IN 9TH FLOOR

Rooms	*x (m)*	*y (m)*	*z (m)*
Room A1-901	-1.0	-8.5	0.0
Room A1-902	-5.0	-2.0	0.0
Room A1-903	-2.0	4.0	0.0
Room A1-904	-6.0	4.0	0.0
Room A1-905	-12.0	4.0	0.0
Room A1-906	-9.0	-2.5	0.0
Room A1-907	-14	-8.5	0.0

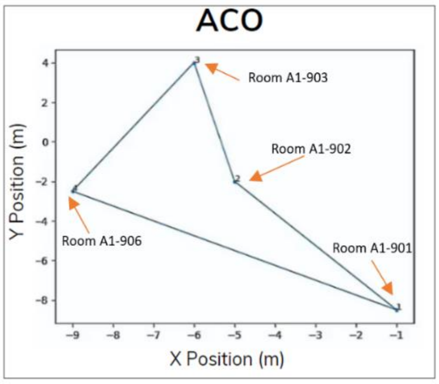


Fig. 37. Solution of ACO for ITSP in MATLAB

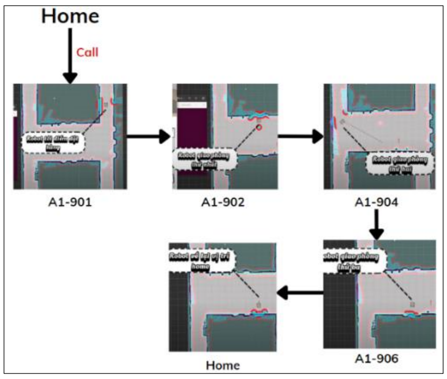


Fig. 38. Optimal path of Mobile Robot

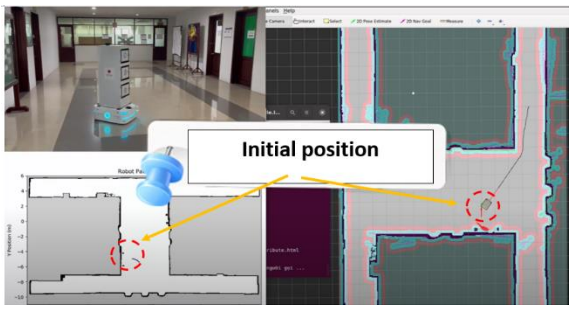


Fig. 39. Initial position of the delivering mobile robot

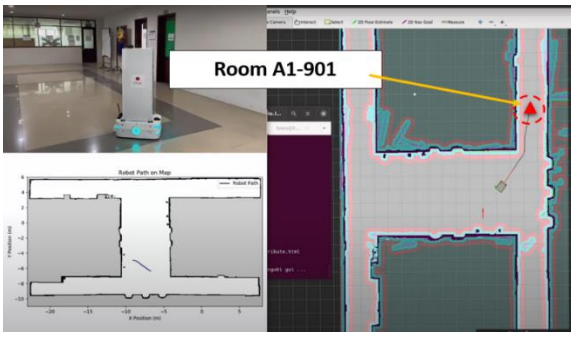


Fig. 40. Delivering mobile robot moving to Room A1-901

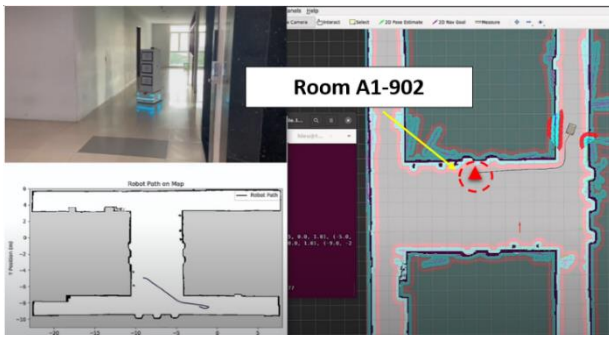


Fig. 41. Delivering mobile robot moving to Room A1-902

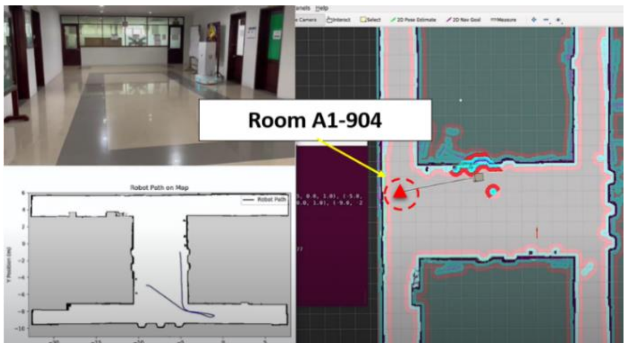


Fig. 42. Delivering mobile robot moving to Room A1-904

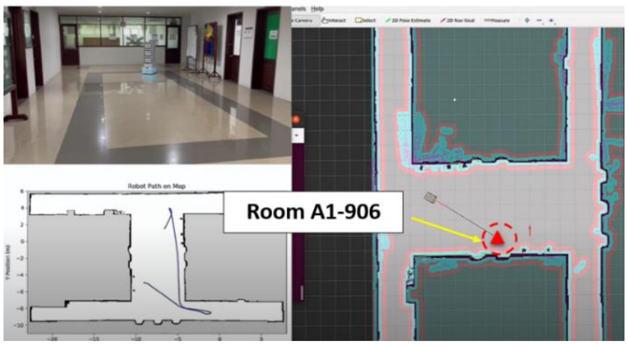


Fig. 43. Delivering mobile robot moving to Room A1-906

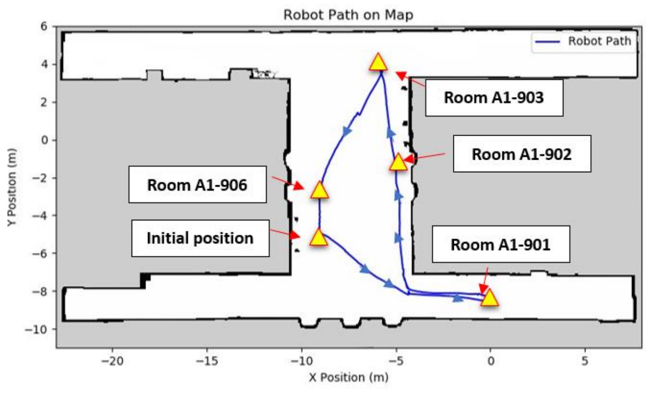


Fig. 44. Optimal moving path of the delivering mobile robot on map

V. CONCLUSION

A delivering mobile robot is designed and built in this study. The ACML and DWA algorithms are proposed to localize the mobile robot in working maps and avoid both static and dynamic obstacles. The optimal path with the shortest traveling distance is determined by applying ACO and Dijkstra's algorithm. The PID controller is used to control the left and right wheels and let the delivery mobile robot travel to multiple targets. The simulation results in MATLAB and Gazebo, and the experimental results show that the proposed method is successful in controlling the delivering mobile robot. Using delivering mobile robot to replace the staff will help to save delivery time, reduce incorrect deliveries, and guarantee the security of the documents. Additionally, with the wide range applications of AI, in the future work, the performance of the mobile robot can be improved by integrating AI/machine learning for path planning and obstacle avoidance. Besides, in this paper, he parameters of ACO and PID controller are obtained by trial-and-error methods, thus, these parameters can be determined optimally by integrating the optimization methods to improve the performance of the system.

Acknowledgment

This work is funded by Ho Chi Minh City University of Technology and Education, Vietnam, under the project with grant number T2025-224.

References

[1] R. R. Shamshiri and I. A. Hameed, *Mobile Robots for Digital Farming*. CRC Press, 2024, doi: 10.1201/9781003306283.

[2] P. Asgharian, A. M. Panchea, and F. Ferland, "A Review on the Use of Mobile Service Robots in Elderly Care," *Robotics*, vol. 11, no. 6, pp. 1–27, 2022, doi: 10.3390/robotics11060127.

[3] R. Hercik, R. Byrtus, R. Jaros, and J. Koziorek, "Implementation of Autonomous Mobile Robot in SmartFactory," *Applied Sciences (Switzerland)*, vol. 12, no. 17, p. 8912, 2022, doi: 10.3390/app12178912.

[4] J. Zhang, Q. Gong, Y. Zhang, and J. Wang, "Finite-time global trajectory tracking control for uncertain wheeled mobile robots," *IEEE Access*, vol. 8, pp. 187808–187813, 2020, doi: 10.1109/ACCESS.2020.3030633.

[5] E. Slawinski, D. Santiago, and V. Mut, "Dual Coordination for Bilateral Teleoperation of a Mobile Robot with Time Varying Delay," *IEEE Latin America Transactions*, vol. 18, no. 10, pp. 1777–1784, 2020, doi: 10.1109/TLA.2020.9387669.

[6] H. Yang, S. Wang, Z. Zuo, and P. Li, "Trajectory tracking for a wheeled mobile robot with an omnidirectional wheel on uneven ground," *IET Control Theory and Applications*, vol. 14, no. 7, pp. 921–929, 2020, doi: 10.1049/iet-cta.2019.1074.

[7] Y. T. Liu, R. Z. Sun, T. Y. Zhang, X. N. Zhang, L. Li, and G. Q. Shi, "Warehouse-Oriented Optimal Path Planning for Autonomous Mobile Fire-Fighting Robots," *Security and Communication Networks*, vol. 2020, p. 13, 2020, doi: 10.1155/2020/6371814.

[8] X. Zhang, J. Lai, D. Xu, H. Li, and M. Fu, "2D Lidar-Based SLAM and Path Planning for Indoor Rescue Using Mobile Robots," *Journal of Advanced Transportation*, vol. 2020, p. 14, 2020, doi: 10.1155/2020/8867937.

[9] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021, doi: 10.26599/TST.2021.9010012.

[10] J. Wang, B. Ma, and K. Yan, "Mobile Robot Circumnavigating an Unknown Target Using Only Range Rate Measurement," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 2, pp. 509–513, 2022, doi: 10.1109/TCSII.2021.3082195.

[11] F. Sun, Y. Chen, Y. Wu, L. Li, and X. Ren, "Motion Planning and Cooperative Manipulation for Mobile Robots With Dual Arms," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 6, pp. 1345–1356, 2022, doi: 10.1109/TETCI.2022.3146387.

[12] Z. Wang, P. Li, Q. Li, Z. Wang, and Z. Li, "Motion Planning Method for Car-Like Autonomous Mobile Robots in Dynamic Obstacle Environments," *IEEE Access*, vol. 11, pp. 137387–137400, 2023, doi: 10.1109/ACCESS.2023.3339539.

[13] H. Cao *et al.*, "Safe Reinforcement Learning-Based Motion Planning for Functional Mobile Robots Suffering Uncontrollable Mobile Robots," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 5, pp. 4346–4363, 2024, doi: 10.1109/TITS.2023.3330183.

[14] S. Hong and D. Park, "ML-Based Fast and Precise Embedded Rack Detection Software for Docking and Transport of Autonomous Mobile Robots Using 2-D LiDAR," *IEEE Embedded Systems Letters*, vol. 16, no. 4, pp. 401–404, 2024, doi: 10.1109/LES.2024.3442927.

[15] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007, doi: 10.1109/TRO.2006.889486.

[16] J. Qiao, J. Guo, and Y. Li, "Simultaneous localization and mapping (SLAM)-based robot localization and navigation algorithm," *Applied Water Science*, vol. 14, no. 7, pp. 1–8, 2024, doi: 10.1007/s13201-024-02183-6.

[17] J. Zhong, D. Kong, Y. Wei, X. Hu, and Y. Yang, "Efficiency-optimized path planning algorithm for car-like mobile robots in bilateral constraint corridor environments," *Robotics and Autonomous Systems*, vol. 186, p. 104923, 2025, doi: 10.1016/j.robot.2025.104923.

[18] J. C. Trujillo, R. Munguia, J. C. Albarrán, and M. Arteaga, "Control and monocular visual SLAM of nonholonomic mobile robots," *European Journal of Control*, vol. 82, p. 101171, 2025, doi: 10.1016/j.ejcon.2024.101171.

[19] Z. An, C. Li, Y. Han, and M. Niu, "Improved Bidirectional JPS Algorithm for Mobile Robot Path Planning in Complex Environments," *Computers, Materials and Continua*, vol. 83, no. 1, pp. 1347–1366, 2025, doi: 10.32604/cmc.2025.059037.

[20] Yujin and G. Xiaoxue, "Optimal Route Planning of Parking Lot Based on Dijkstra Algorithm," in *Proceedings - 2017 International Conference on Robots and Intelligent System, ICRIS 2017*, pp. 221–224, 2017, doi: 10.1109/ICRIS.2017.62.

[21] S. Kim, H. Jin, M. Seo, and D. Har, "Optimal Path Planning of Automated Guided Vehicle using Dijkstra Algorithm under Dynamic Conditions," in *2019 7th International Conference on Robot Intelligence Technology and Applications, RiTA 2019*, pp. 231–236, 2019, doi: 10.1109/RITAPP.2019.8932804.

[22] H. Afrisal, A. Syakur, M. A. Riyadi, G. K. Nugraha, A. A. Nanda, and I. Setiawan, "Experimental Analysis on Path Planning Strategy using Dijkstra Algorithm and Follow the Carrot Method for Indoor Mobile Robot Navigation," in *Proceedings - International Conference on Informatics and Computational Sciences*, pp. 193–198, 2021, doi: 10.1109/ICICoS53627.2021.9651862.

[23] D. D. Zhu and J. Q. Sun, "A New Algorithm Based on Dijkstra for Vehicle Path Planning Considering Intersection Attribute," *IEEE Access*, vol. 9, pp. 19761–19775, 2021, doi: 10.1109/ACCESS.2021.3053169.

[24] Y. Su, J. Xin, and C. Sun, "Dynamic Path Planning for Mobile Robots Based on Improved RRT∗ and DWA Algorithms," *IEEE Transactions on Industrial Electronics*, 2025, doi: 10.1109/TIE.2025.3546349.

[25] Y. Luo, S. Lin, Y. Wang, and K. Liang, "Optimized Path Planning for Autonomous Guided Vehicle Through Fusion of Improved A* and Dynamic Window Approach," *IEEE Access*, vol. 13, pp. 68577–68586, 2025, doi: 10.1109/ACCESS.2025.3561005.

[26] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997, doi: 10.1109/100.580977.

[27] Y. Li *et al.*, "A Mobile Robot Path Planning Algorithm Based on Improved A∗ Algorithm and Dynamic Window Approach," *IEEE Access*, vol. 10, pp. 57736–57747, 2022, doi: 10.1109/ACCESS.2022.3179397.

[28] X. Wu, K. Tu, and J. Jiang, "Study on Improved Dynamic Window Approach for Mobile Robot Under Narrow Channel," in *2023 5th International Conference on Robotics, Intelligent Control and Artificial Intelligence, RICAI 2023*, pp. 139–142, 2023, doi: 10.1109/RICAI60863.2023.10489376.

[29] M. Dobrevski and D. Skocaj, "Dynamic Adaptive Dynamic Window Approach," *IEEE Transactions on Robotics*, vol. 40, pp. 3068–3081, 2024, doi: 10.1109/TRO.2024.3400932.

[30] S. Thrun, *Probabilistic robotics*, vol. 45, no. 3. MIT Press, 2002, doi: 10.1145/504729.504754.

[31] W. Xiaoyu, L. Caihong, S. Li, Z. Ning, and F. U. Hao, "On adaptive monte carlo localization algorithm for the mobile robot based on ROS,"

in *Chinese Control Conference, CCC*, vol. 2018, pp. 5207–5212, 2018, doi: 10.23919/ChiCC.2018.8482698.

[32] G. Puga, N. Espinosa, M. Hidalgo, O. García, and I. Paunovic, "Beluga: A Modern Monte Carlo Localization Package for ROS and ROS 2," in *Springer Proceedings in Advanced Robotics*, pp. 126–130, 2024, doi: 10.1007/978-3-031-76424-0_23.

[33] H. Zhu and Q. Luo, "Indoor Localization of Mobile Robots Based on the Fusion of an Improved AMCL Algorithm and a Collision Algorithm," *IEEE Access*, vol. 12, pp. 67199–67208, 2024, doi: 10.1109/ACCESS.2024.3399192.

[34] M. A. Chung and C. W. Lin, "An Improved Localization of Mobile Robotic System Based on AMCL Algorithm," *IEEE Sensors Journal*, vol. 22, no. 1, pp. 900–908, 2022, doi: 10.1109/JSEN.2021.3126605.

[35] L. Yu, M. Li, and G. Pan, "Indoor Localization Based on Fusion of AprilTag and Adaptive Monte Carlo," in *IEEE Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2021*, pp. 464–468, 2021, doi: 10.1109/ITNEC52019.2021.9587205.

[36] Y. Li, L. Jiang, B. Tang, Y. Guo, B. Lei, and H. Liu, "Adaptive Monte Carlo Localization in Unstructured Environment via the Dimension Chain of Semantic Corners," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 7, pp. 9714–9724, 2024, doi: 10.1109/TII.2024.3385836.

[37] M. Ibrahim, M. Salem, I. Hossain, A. Sarbabidya, and S. K. Saha, "Autonomous robot navigation with SLAM, AMCL, and real-time user interaction for task management and identity verification," in *2024 IEEE 3rd International Conference on Robotics, Automation, Artificial-Intelligence and Internet-of-Things, RAAICON 2024 - Proceedings*, pp. 252–257, 2024, doi: 10.1109/RAAICON64172.2024.10928347.

[38] W. Huang and J. Yuan, "Improvements based on Adaptive Monte Carlo Localization," in *2024 5th International Conference on Information Science, Parallel and Distributed Systems, ISPDS 2024*, pp. 103–106, 2024, doi: 10.1109/ISPDS62779.2024.10667543.

[39] A. De Mello Gai, S. Bevilacqua, A. R. Cukla, and D. F. T. Gamarra, "Evaluation on IMU and odometry sensor fusion for a Turtlebot robot using AMCL on ROS framework.," in *Proceedings - 2023 Latin American Robotics Symposium, 2023 Brazilian Symposium on Robotics, and 2023 Workshop of Robotics in Education, LARS/SBR/WRE 2023*, pp. 637–642, 2023, doi: 10.1109/LARS/SBR/WRE59448.2023.10332977.

[40] L. P. N. Matias, T. C. Santos, D. F. Wolf, and J. R. Souza, "Path Planning and Autonomous Navigation using AMCL and AD," in *Proceedings - 12th LARS Latin American Robotics Symposium and 3rd SBR Brazilian Robotics Symposium, LARS-SBR 2015 - Part of the Robotics Conferences 2015*, pp. 320–324, 2016, doi: 10.1109/LARS-SBR.2015.31.

[41] G. Shang, G. Gu, F. Jiang, and X. Hu, "Research on AMCL Algorithm Coupled with DWA Algorithm in Logistics Scenarios," in *2024 IEEE International Workshop on Radio Frequency and Antenna Technologies, iWRF and AT 2024*, pp. 453–458, 2024, doi: 10.1109/iWRFAT61200.2024.10594703.

[42] J. Weber, M. Heimbach, and M. Schmidt, "Experimental Validation of NDT-AMCL: a Precise and Reliable Localizer for Mobile Robots in Human Crowds Using Normal Distribution Transforms," in *Proceedings - 2024 8th IEEE International Conference on Robotic Computing, IRC 2024*, pp. 162–169, 2024, doi: 10.1109/IRC63610.2024.00035.

[43] M. Dorigo, M. Birattari, and T. Stützle, "Ant colony optimization artificial ants as a computational intelligence technique," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, Nov. 2006, doi: 10.1109/CI-M.2006.248054.

[44] M. Breaban, R. Necula, D. Lucanu, and D. Stamate, "Joint Decision Making in Ant Colony Systems for Solving the Multiple Traveling Salesman Problem," *Procedia Computer Science*, vol. 225, pp. 3498–3507, 2023, doi: 10.1016/j.procs.2023.10.345.

[45] M. Liu *et al.*, "A Slime Mold-Ant Colony Fusion Algorithm for Solving Traveling Salesman Problem," *IEEE Access*, vol. 8, pp. 202508–202521, 2020, doi: 10.1109/ACCESS.2020.3035584.

[46] B. Song, S. Tang, and Y. Li, "A new path planning strategy integrating improved ACO and DWA algorithms for mobile robots in dynamic environments," *Mathematical Biosciences and Engineering*, vol. 21, no. 2, pp. 2189–2211, 2024, doi: 10.3934/mbe.2024096.

[47] X. Zeng, Q. Song, S. Yao, Z. Tian, and Q. Liu, "Traveling Salesman Problems with Replenishment Arcs and Improved Ant Colony Algorithms," *IEEE Access*, vol. 9, pp. 101042–101051, 2021, doi: 10.1109/ACCESS.2021.3093295.

[48] M. Wang, T. Ma, G. Li, X. Zhai, and S. Qiao, "Ant Colony Optimization with an Improved Pheromone Model for Solving MTSP with Capacity and Time Window Constraint," *IEEE Access*, vol. 8, pp. 106872–106879, 2020, doi: 10.1109/ACCESS.2020.3000501.

[49] Y. Liu, S. Guo, S. Tang, J. Song, and J. Zhang, "Path Planning for Robots Based on Adaptive Dual-Layer Ant Colony Optimization Algorithm and Adaptive Dynamic Window Approach," *IEEE Sensors Journal*, vol. 25, no. 11, pp. 19694–19708, 2025, doi: 10.1109/JSEN.2025.3557437.

[50] G. Li, C. Liu, L. Wu, and W. Xiao, "A mixing algorithm of ACO and ABC for solving path planning of mobile robot," *Applied Soft Computing*, vol. 148, p. 110868, 2023, doi: 10.1016/j.asoc.2023.110868.

[51] F. Sui, X. Tang, Z. Dong, X. Gan, P. Luo, and J. Sun, "ACO+PSO+A*: A bi-layer hybrid algorithm for multi-task path planning of an AUV," *Computers and Industrial Engineering*, vol. 175, p. 108905, 2023, doi: 10.1016/j.cie.2022.108905.

[52] K. A. Athira, R. Yalavarthia, T. Saisandeepa, K. S. S. Harshitha, A. Shaa, and D. J. Udayan, "ACO-DTSP Algorithm: Optimizing UAV Swarm Routes with Workload Constraints," in *Procedia Computer Science*, vol. 235, pp. 163–172, 2024, doi: 10.1016/j.procs.2024.04.019.

[53] E. De Kuyffer, W. Joseph, L. Martens, and T. De Pessemier, "Travel route and formation optimization for flocks of drones in package delivery by using an ACO based V-Shape algorithm," *Results in Engineering*, vol. 24, p. 103627, 2024, doi: 10.1016/j.rineng.2024.103627.

[54] D. Li, L. Wang, J. Cai, K. Ma, and T. Tan, "Research on Terminal Distance Index-Based Multi-Step Ant Colony Optimization for Mobile Robot Path Planning," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 4, pp. 2321–2337, 2023, doi: 10.1109/TASE.2022.3212428.

[55] V. P. Vu, T. G. T. Nguyen, X. S. Nguyen, V. T. Le, and D. H. Pham, "ITSP/IMTSP-Based Path Planning for Multiple-Mobile Robot System," *IEEE Access*, vol. 12, pp. 99183–99200, 2024, doi: 10.1109/ACCESS.2024.3427798.

[56] X. Yao, X. Shi, X. Zhang, and X. Mei, "An Ant Colony Optimization Parameter Tuning Method Based on Uniform Design for Path Planning of Mobile Robots," in *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications, AEECA 2022*, pp. 1183–1191, 2022, doi: 10.1109/AEECA55500.2022.9918906.

[57] K. Tomitagawa, A. Anuntachai, S. Chotipant, O. Wongwirat, and S. Kuchii, "Performance Measurement of Energy Optimal Path Finding for Waste Collection Robot Using ACO Algorithm," *IEEE Access*, vol. 10, pp. 117261–117272, 2022, doi: 10.1109/ACCESS.2022.3219416.

[58] H. J. Kaleybar, M. Davoodi, M. Brenna, and D. Zaninelli, "Applications of Genetic Algorithm and Its Variants in Rail Vehicle Systems: A Bibliometric Analysis and Comprehensive Review," *IEEE Access*, vol. 11, pp. 68972–68993, 2023, doi: 10.1109/ACCESS.2023.3292790.