International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 12 Issue 5 ǁ May 2024 ǁ PP. 173-179

Research of Autonomous Navigation for Mobile Robots Using Karto SLAM Algorithm Under ROS

[1]Tran Hoang Viet,[1]Truong Xuan Nghiem,[2]Nguyen Minh Huy,
[3]Truong Gia Binh, [4*]Vo Thanh Ha

[1]Department of Control and Automation, Faculty of Electrical Engineering, University of Economics-Technology for Industries, Hanoi, city Vietnam
[2]Hanoi University of Science and Technology, Hanoi, Vietnam
[3]HUS High School for Gifted Students, Hanoi
*[*4]Department of Cybernetics, Faculty of Electrical and Electronic Engineering, University of Transport and Communications, Hanoi city, Vietnam*
Corresponding Author: Vo Thanh Ha

Abstract
Researching navigation for mobile robots to follow the desired trajectory is crucial to ensure their swift and accurate movement while avoiding obstacles. Most mapping techniques rely on Simultaneous Localization and Mapping (SLAM), which involves creating a map and determining the robot's position by collecting data from sensors like Lidar and cameras. The Karto SLAM algorithm is a graph optimization method that utilizes ghosts. It optimizes the Cholesky factorization and does not require iterating to solve sparse systems. The map is represented by the mean value of the histogram, with each node denoting a point on the robot's trajectory and sensor data. When a new node is added, the map is recalculated and updated based on the node's spatial constraints. In real-world scenarios, Karto SLAM exhibits minimal error (1.03 cm), making it the preferred choice for mobile robots. The accuracy of this navigation solution is evident. The correctness of this navigation solution is demonstrated through ROS_Gazebo simultion

Keywords: ***ROS, Karto-SLAM, SLAM, Mobile Robot, Gazebo.***

Date of Submission: 06-05-2024 Date of acceptance: 18-05-2024

I. INTRODUCTION

Autonomous Robots play a crucial role in enhancing efficiency and productivity in modern industrial settings. By leveraging cutting-edge technology and artificial intelligence, these robots can perform tasks with precision and consistency, reducing errors and minimizing downtime. Their ability to navigate complex environments autonomously makes them invaluable assets in optimizing workflows and ensuring seamless operations. With the rise of digitalization and Industry 4.0, the integration of autonomous robots is set to revolutionize the way we approach manufacturing and logistics, paving the way for a more agile and competitive industrial landscape [1, 3]. Mobile robotics is a field of research in robotics and information engineering [4]. Mobile robots can be controlled by humans or fully automated with the ability to navigate their environment autonomously [5]. They are widely utilized in various sectors such as industry, commerce, military, and security [6]. Building a map of the working environment and understanding it is crucial for mobile robots to determine their position and identify obstacles. Mapping is the process through which mobile robots model their environment. With the created map, they can navigate automatically, enabling applications in areas like search and rescue and smart transportation. The simultaneous performance of mapping and positioning tasks by mobile robots is known as SLAM (Simultaneous Localization and Mapping) [7]. The Karto SLAM algorithm optimizes networks using ghosts, enhancing the Cholesky factorization process and eliminating the need for iterative methods in solving sparse systems. The map represents average histogram values, with each node indicating a specific position on the robot's path and its associated sensor data. Upon adding a new node, the map recalculates and updates while considering the node's spatial constraints. Karto SLAM demonstrates minimal inaccuracy (1.03 cm) in real-world scenarios, making it a preferred choice for mobile robots [8, 10]. Its efficiency lies in its seamless adaptation to changing environments, establishing it as a reliable solution for tasks requiring precise mapping and localization. The algorithm's robustness in handling sensor noise and uncertainties further solidifies its position as a top-tier choice in simultaneous localization and mapping [11, 14].

To assess the efficacy of the Karto SLAM algorithm, this study constructs a kinematic and dynamic model of a mobile robot with 02 active wheels, 02 passive rear wheels, and 01 Lidar sensor for position and distance determination. Additionally, an IMU sensor is utilized to ascertain the mobile robot's steering angle. The algorithm is implemented in Python and simulated using ROS_Gazebo simulation.

The paper comprises 5 primary sections. The first part outlines the significance of navigation technology for mobile robots. Subsequently, a kinematic and dynamic model for the vehicle is developed. Part 3 utilizes the kinematic model to devise the controller and regulate the speed of the DC motor. Part 4 presents the project's evaluation results through simulation. Lastly, conclusions and potential avenues for future research are discussed.

II. KINEMATIC MODEL FOR A MOBILE ROBOT

Attach a stationary frame of reference to the plane of the moving medium, as depicted in Fig. (1). The stationary frame of reference allows us to analyze the motion of the medium relative to a fixed point, providing a clearer understanding of mobile robots' behavior and characteristics. By anchoring our observations to this frame, we can track the movement of the medium over time and make meaningful comparisons between different points within the system

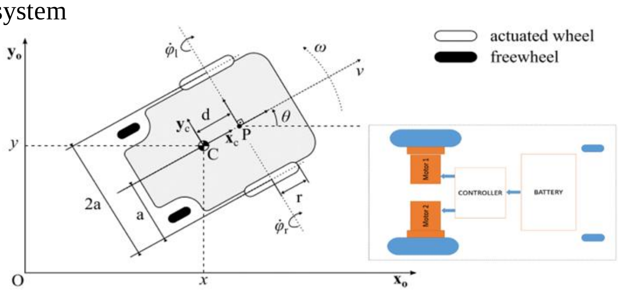


Figure 1: Kinetic relationship of mobile robots

where: *P* is Intersection of the symmetry axis with the axis of the wheels; *C* is mass center or guidance point; *d* is distance between *C* and *P*; *r* is right and left wheel radius; *2a* is distance between the actuated wheels and the symmetry axis; m_c is mass of the robot without wheels and motors; m_ω is mass of each wheel and motor assembly; m_t is total mass of the mobile robot; m_t is a moment of inertia of the mobile robot without wheels and motors about the vertical axis through P; I_c is moment of inertia of the DWMR without wheels and motors about the vertical axis through P; I_w is Moment of inertia of each wheel and motor about the wheel axis; *I* is total inertia moment of the robot; $\dot{\varphi}_r, \dot{\varphi}_l$ are angular velocity of the right and left wheels; v, ω are Angular velocity of the right and left wheels; q is linear and angular velocities of robot; θ is orientation angle.

The kinematics model equation for the mobile robot is given by Eq. (1) [1]. Eq. (1) describes the relationship between the robot's position, velocity, and acceleration, providing a crucial framework for analyzing its motion in a dynamic environment. This equation plays a fundamental role in the field of robotics, enabling researchers and engineers to design control algorithms and plan trajectories for mobile robots with precision and efficiency.

$$\dot{q} = \begin{bmatrix} \frac{r}{2}\cos(\theta) & \frac{r}{2}\mathrm{d}\cos(\theta) \\ \frac{r}{2}\sin(\theta) & \frac{r}{2}\mathrm{d}\sin(\theta) \\ \frac{r}{2a} & \frac{r}{2a} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_r \\ \dot{\varphi}_l \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{1}$$

where: *r* is right and left wheel radius; *2a* is distance between the actuated wheels and the symmetry axis; $\dot{\varphi}_r, \dot{\varphi}_l$ are angular velocity of the right and left wheels; v, ω are angular velocity of the right and left wheels;

The kinematic error model q_e of a self-propelled robot is a mathematical equation describing the deviation of the robot's position and posture, when the motion-controlled robot follows a desired trajectory ξd and is defined as following in the original coordinate system is as follows Eq. (2):

$$q_e = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = R(\theta)(q_r - q) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \quad (2)$$

The derivative of Eq. (2) combined with the kinematic equation of the mobile robot Eq. (1). The system of error function equations is as follows Eq. (3):

$$\dot{q}_e = \begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \cos(\theta_e) & 0 \\ \sin(\theta_e) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \vartheta_r \\ \omega_r \end{bmatrix} + \begin{bmatrix} -1 & y_e \\ 0 & -x_e \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \vartheta \\ \omega \end{bmatrix} \quad (3)$$

III. SLAM UNDER ROS

3.1 Light detection and ranging (Lidar) SLAM

Lidar employs laser or distance sensors for detection. Compared to cameras and other sensors like Time of Flight (ToF), lasers offer significantly higher accuracy and are utilized in fast-moving vehicles such as autonomous cars and drones. Laser sensor outputs typically consist of 2D (x, y) or 3D (x, y, z) point cloud data, enabling precise distance measurements essential for SLAM-based map construction. Vehicle position is determined by calculating movement (distance traveled). These 2D or 3D point cloud maps can be visualized as grid maps or voxel maps as shown in Fig.1.

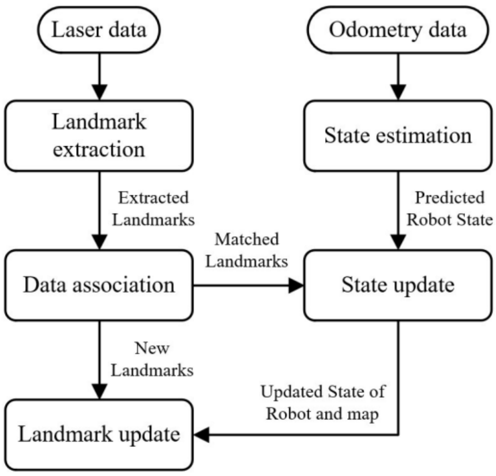


Figure 2: The process diagram of mapping with SLAM

In Fig. 2, the process of mapping with SLAM can be broken down into four steps:

• *Step 1*: The LIDAR sensor scans the surroundings, returning 360 points representing 360 degrees. Each point provides a distance value from the sensor to obstacles.

• *Step 2*: Using the SLAM algorithm, preprocessing interpolates these points onto a grid map.

• *Step 3*: Scan Matching optimizes connections between drawn and new points to minimize occupied space.

• *Step 4*: By moving the LIDAR based on previous calculations, a map of the scanned environment is generated.

3.2 Karto SLAM

Five common technologies for research and development in 2D LiDAR SLAM mapping technology are Hector SLAM, Gmapping, Karto SLAM, Core SLAM, and Lago SLAM [13]. The Karto SLAM algorithm demonstrates the smallest error (1.03 cm) in real environments, making it the algorithm of choice for the robot. The Karto SLAM algorithm employs a graph optimization approach utilizing an optimized and non-iterative Cholesky matrix to handle sparse systems. The map is represented by the mean of the histogram, with each node denoting a location on the robot's trajectory and a dataset of sensor measurements. Upon the appearance of a new node, the map undergoes calculation and updating based on the spatial constraints of the node arrow. The ROS version of Karto SLAM incorporates Spare Pose Adjustment (SPA) for scan matching and loop closure detection. The algorithm's memory requirement increases with the number of landmarks. In certain scenarios, Karto SLAM proves more efficient due to its single-point graph structure (robot pose).

The structure of the Karto SLAM algorithm is shown in Fig. (3). The Karto SLAM algorithm's structure is depicted in Fig (3). It relies on a graph-based method employing scan matching and loop closure techniques to construct a map of the environment and determine the robot's location within it. The algorithm upholds a graph model of the environment, where nodes signify robot poses and edges denote constraints among poses. Through optimizing the graph with methods like least squares optimization, Karto SLAM enhances the precision of the robot's estimated trajectory and map.

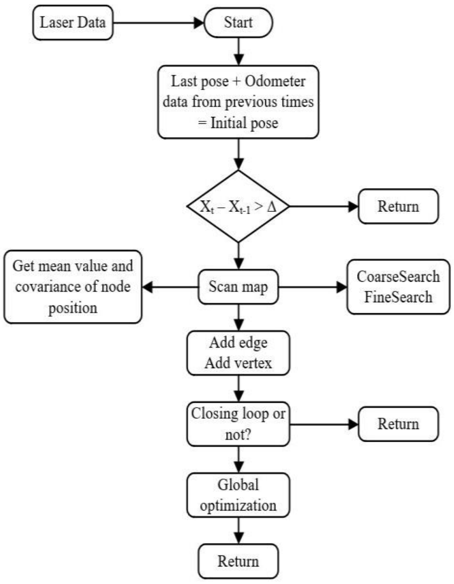


Figure 3: The structure of the Karto SLAM algorithm

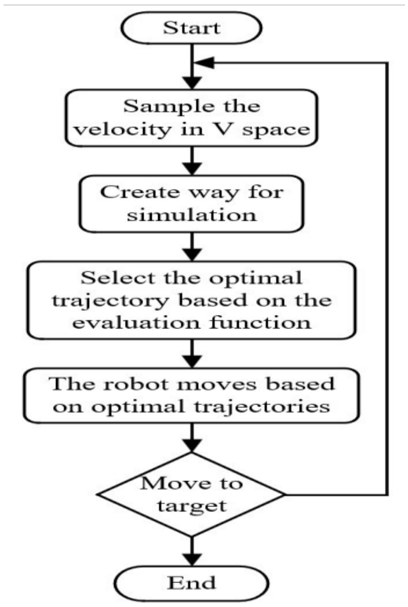


Figure 4: The DWA algorithm diagram

IV. RESULT AND DISCUSSION

In this evaluation, we aim to demonstrate the effectiveness of integrating the Karto_SLAM approach with the A* algorithm within the ROS-Gazebo environment. By leveraging the A* path-finding algorithm and the dynamic window approach (DWA) for obstacle avoidance, we anticipate achieving robust and efficient navigation capabilities for our robotic system. Through rigorous testing and analysis, we seek to showcase the enhanced mapping and localization performance that arises from this integration, ultimately highlighting the potential for seamless and reliable autonomous navigation in complex environments.

4.1 A* Algorithm

The heuristic function guides the A* algorithm's search direction. It calculates the cost of each adjacent node from the starting node, selecting the node with the least cost as the next route point. This evaluation is shown in Eq. (4).

$$f(x) = g(x) + h(x) \tag{4}$$

where:

g(x) is the weighted sum of the edges that have been passed.

h(x) is a heuristic evaluation function of the minimum cost to reach the destination from *x*.

When the function *f(x)* has a lower value, *x*'s priority is higher, therefore a minimal heap structure can be utilized to implement this priority queue.

The heuristic function is shown in Eq. (5).

$$h(x) \leq g(y) - g(x) + h(y) \tag{5}$$

4.2 Dynamic Window Approach (DWA) Algorithm

The DWA algorithm avoids foreign objects used in the ROS operating system to build a local plan. The DWA algorithm diagram is shown in Fig. 4.

The DWA algorithm diagram illustrates the DWA for robot navigation. It demonstrates how the robot determines its permissible velocity and angular velocity by considering the dynamic window, obstacle information, and goal position. This diagram aids in comprehending how the robot strategizes its path and circumvents obstacles in real-time scenarios.

Initially, the robot operates not in the x, y coordinate system but within a search space defined by the maximum achievable speed x and angular velocity ω derived from the objective function, while taking into account the robot's orientation, velocity, and obstacles. Fig. 5 depicts the velocity search space and dynamic window, while Fig. 6 showcases the mobile robot's longitudinal velocity *v* and angular velocity.

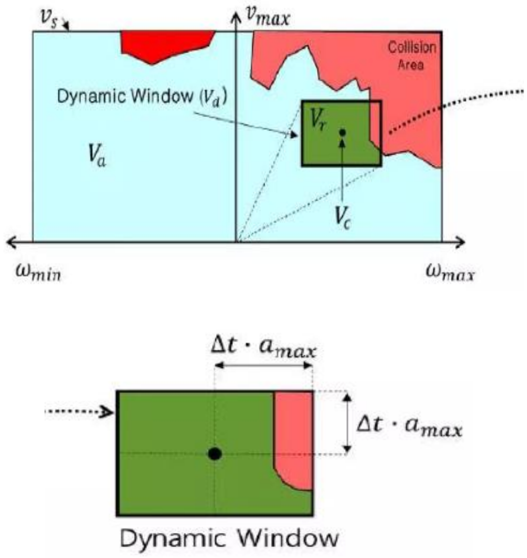


Figure 5: The robot's orientation, velocity, and obstacles

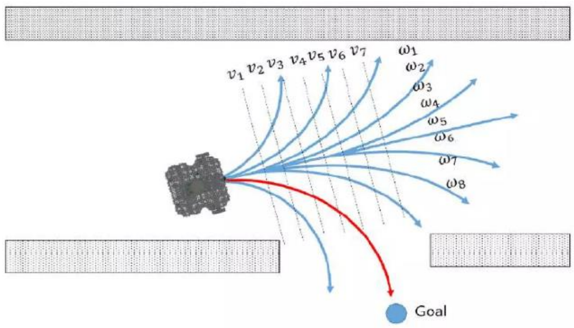


Figure 6: the mobile robot's longitudinal velocity *v* and angular velocity

The vicinity surrounding the robot is categorized into three zones:

- The first zone is a rectangle precisely encompassing the robot's dimensions, known as the *base_footprint*, allowing the robot to rotate freely within it. As the robot moves, this area shifts to indicate the robot's occupied space on the map.

- The second zone is a larger rectangle sized by the user to indicate potential collision risks.

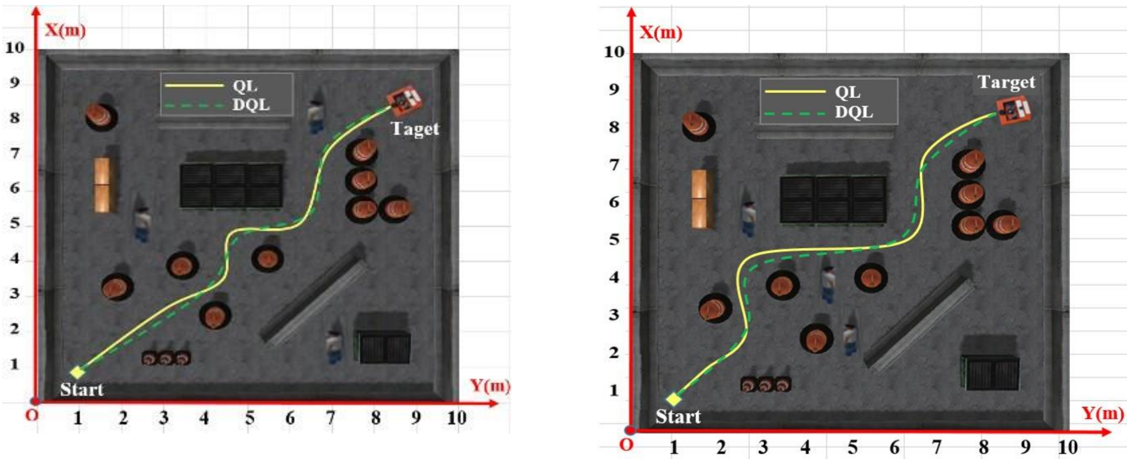
- The third zone lies beyond the collision risk area but remains within the Lidar sensor's field of view.

When an obstacle emerges in the third zone, the robot pauses to eliminate the obstacle from the map. This behavior is particularly relevant when a person swiftly traverses the robot's field of view. If the individual halts, the robot rotates in place to alter its course until the path is clear. If the obstruction persists, the robot attempts obstacle removal in the second zone through a rotational scan of the map. If the robot remains unable to navigate around the obstacle, it deems itself stuck and reports an error indicating the inability to reach the destination.

4.3 Simulation Results

The control robot system created a scenario in ROS-Gazebo to simulate a virtual factory, connecting the virtual environment with the real world. Various obstacles were set up to test the Karto_SLAM combined with the A* algorithm in ROS-Gazebo. It is the environment that includes walls, stationary blocks, moving individuals, targets, and mobile robots. The robot had to navigate to the goal while avoiding both stationary and moving obstacles, as shown in Fig 7.

Figure 7: The working environment of mobile robot in ROS_Gazebo



(a) Case 1 (b) Case 2

Figure 8: Depicts a realistic environment for route planning for a mobile robot

Fig. 8 depicts a realistic environment for route planning for a mobile robot, with a workspace of 12m x12m and a minimum distance between blocks of 0.6 m. The robot begins at location (1, 1) for all tests and moves to the goal (11, 11). Table 1 summarizes for 2 simulation case.

Table 1. Simulation results on ROS-Gazebo

No	Distance (m)	Run time(s)
Case 1	17.758	12,314
Case 2	18.416	14.637

In all 2 cases, simulation results show promising results for the Karto_SLAM combined with the A* algorithm in ROS-Gazebo for mobile robots, especially regarding time. In addition, the simulation results in Table 1 show that the Karto_SLAM combined with the A* algorithm in ROS-Gazebo ensures the mobile robot plans the optimal path. At the same time, the computation time to establish the optimal rotation and motion is much better. In summary, the results demonstrate the superiority of the Karto_SLAM combined with the A* algorithm in ROS-Gazebo in orbiting planning for mobile robots. Not only does it excel in terms of efficiency and speed, but it also guarantees optimal path planning for the mobile robot.

Furthermore, the robustness of the Karto_SLAM combined with the A* algorithm in ROS-Gazebo was evident in various simulated scenarios, showcasing its adaptability and reliability. The seamless integration of these technologies not only streamlines the navigation process but also enhances the overall performance of mobile robots in dynamic environments. As advancements in robotics continue to evolve, the utilization of such cutting-edge solutions will undoubtedly play a pivotal role in shaping the future of autonomous system

V. CONCLUSION

This research has developed a 2-wheel drive adaptive mobile robot positioning and mapping system using ROS. The Karto SLAM algorithm was chosen to create 2D maps following an efficiency comparison of various 2D laser SLAM algorithms. Global path optimization for robot planning is carried out using the A* algorithm. Local path planning for real-time obstacle avoidance is achieved through the DWA algorithm. Consequently, once the target point is set, the robot can autonomously navigate to it successfully. The robot's localization accuracy was evaluated through extensive testing in various environments, showcasing robust performance even in challenging scenarios with dynamic obstacles. Additionally, the system's modular design allows for easy integration of additional sensors or algorithms, making it adaptable to different applications and environments. Overall, this research represents a significant advancement in autonomous mobile robot technology, demonstrating the effectiveness of using ROS for developing complex robotic systems.

REFERENCES

[1]. Volos CK, Kyprianidis IM, Stouboulos I N. A chaotic path planning generator for autonomous mobile robots. Robots Auton Syst, 2012, 60: 651–656.

[2]. Chaari I, Koubaa A, Trigui S, et al. SmartPATH: an efficient hybrid ACO-GA algorithm for solving the global path planning problem of mobile robots. Int J Adv Robot Syst, 2014, 11: 399-412.

[3]. MS Gharajeh and HB Jond, "An intelligent approach for autonomous mobile robot’s path planning based on adaptive neuro-fuzzy inference system," Ain Shams Eng. J. May 2021, doi: 10.1016/j.asej.2021.05.005.

[4]. C. Zhang, L. Zhou, Y. Li, and Y. Fan, "A dynamic path planning method for social robots in the home environment," Electronics, vol. 9, no. 7, p. 1173, Jul. 2020

[5]. X. Yingqi, S. Wei, Z. Wen, L. Jingqiao, L. Qinhui, and S. Han, "A real-time' dynamic path planning method combining artificial potential field method and biased target RRT algorithm," J. Phys., Conf. Ser, vol. 1905, no. first, May 2021, Art. no. 012015.

[6]. .B. Yang, J. Yan, Z. Cai, Z. Ding, D. Li, Y. Cao, and L. Guo, "A novel heuristic emergency path planning method based on vector grid map," ISPRS Int. J. Geo-Inf, vol. 10, no. 6, p. 370, May 2021.

[7]. S. Xiao, X. Tan, and J. Wang, "A simulated annealing algorithm and grid map-based UAV coverage path planning method for 3D reconstruction," Electronics vol. 10, no. 7, p. 853, Apr. 2021.

[8]. T. Lin, "A path planning method for mobile robot based on A and antcolony algorithms," J. Innov. Soc. Sci. Res.vol. 7, no. 1, pp. 157-162, 2020.

[9]. Jianming Guo ; Liang Liu ; Qing Liu ; Yongyu Qu , “ An Improvement of D* Algorithm for Mobile Robot Path Planning in Partial Unknown Environment ” Print ISBN: 978-0-7695-3804-4, DOI: 10.1109/ICICTA.2009.561

[10]. Lai, X. , Wu, D. , Wu, D. , Li, JH and Yu, H. (2023), "Enhanced DWA algorithm for local path planning of mobile robot", Industrial Robot , Vol. 50 No. 1, pp. 186 194. https://doi.org/10.1108/IR-05-2022-0130

[11]. C. Zong, X. Han, D. Zhang, Y. Liu, W. Zhao, and M. Sun, "Research on local path planning based on improved RRT algorithm," Proc. Inst. Mech. Eng, vol. 235, no. 8, pp. 2086–2100, Mar. 2021.

[12]. Tsai CC, Huang HC, Chan C K. Parallel elite genetic algo rithm and its application to global path planning for autonomous robot navigation. IEEE Trans Ind Electron, 2011, 58: 4813–4821.

[13]. Saska M, Macaˇs M, Pˇreuˇcil L, et al. Robot path planning using particle swarm optimization of Ferguson splines. Print: Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). New York: IEEE Press, 2006. 833–839.

[14]. Raja P, Pugazhenthi S. On-line path planning for mobile robots in dynamic environments. Neural Netw World, 2012, 22: 67–83