A Mobile Robot with a Manipulator to Alleviate the Shortage of Health Workers in Hospitals in COVID 19

Zichang Zhou

Department of Mechanical, Materials and Manufacturing Engineering, Faculty of Engineering, Nottingham University, United Kingdom

sqyzz2@nottingham.ac.uk

Abstract. COVID-19 has brought an unprecedented burden to hospitals. Hospitals are facing many problems, such as understaffing and infected employees. But this has also promoted the development of hospital robots to a certain extent. This paper aimed to develop a robot that can move around the hospital and pick and send stuff to patients. The robot is mainly divided into 2 two parts: the moving part and the grasping part. The moving part includes map construction, localization, and navigation with gmapping, A*, and DWA algorithms that would stimulate on the ROS platform. For the grasping part, the forward and inverse kinematics of the robot arm and PD control would be analyzed using MATLAB with the Robotics System Toolbox, Simulink, and Simscape Multibody to simulate and control the robot arm. Using this approach, the robot can reach the target position and avoid dynamic obstacles. In addition, the robot arm can pick up and drop the ball with the designed trajectory.

Keywords: Gmapping; A* Algorithm; DWA; Kinematic; PD.

1. Introduction

Nowadays, COVID-19 is still a primary health problem in the world, although various vaccines have been invented. Hospitals where the forefront of the fight against the epidemic still meet problems like insufficient nurses, which can overload medical workers, and patients cannot receive timely and effective health care due to a lack of staff [1]. In addition, health care workers in direct contact roles have a high risk of being infected or even dying if they have prolonged contact with these patients. As the WHO estimated, from January 2020 to May 2020, about 115500 medical workers died from COVID-19 [2]. This was a vast number that hospitals and governments should pay attention to and minimize damage in future epidemics. One of the most direct and effective ways is to reduce the contact time between medical staff and patients. To achieve this, robots are a potential option to take the place of some easy work of the medical workers, like delivering stuff or collecting patient health data. Nowadays, compared with the industrial and logistics scenarios, although the technological development level of this kind of robot has matured, few robots have been applied to the hospital environment. The main reasons include the system, the technical requirements being more stringent in the hospital environment, and the patients' distrust of robots [3]. However, in this Covid 19 period, the promotion of hospital robots has been accelerated. Some robots like Aethon's TUG autonomous mobile robot have already been used to transport some medical products and sensitive materials. In addition, a company named Xenex has created a robot that can disinfect.

This paper aims to design robots that can take some simple kinds of stuff for patients in hospital beds. The robot can walk around the ward according to the proposed work algorithm while using the robot arms to catch simple stuff. The Turtlebot with OpenMANIPULATOR was used as a robot model, and the software ROS, gazebo, and Matlab was used to test algorithms and prove the result. According to the functions of robots, the essay was mainly divided into two parts: the moving part and the grasping part.

2. Methods

2.1 Moving System

The moving system can contain three main parts: map construction, localization, and navigation. The SLAM algorithms are usually used to construct the terrain map and locate the robot's location. The pathing planning occupies a crucial position in the moving system, and depending on the environmental conditions, it can be divided into global path planning and local path planning. Global path planning is normally used in the static environment, and location path planning is used in the dynamic environment.

2.2 SLAM

Considering the application environment is in hospitals, as it is the indoor environment and the flat ground. 2D SALM is competent for this task and reduces the calculation burden of the hardware. The laser sensor was chosen to measure the distance and angles from obstacles. In addition, compared with visual SLAM, it has higher accuracy and a larger detected field, and it doesn't depend on the light [4], so it can work in poorly light areas in the hospital at night.

As the 2D SLAM and laser sensor, the Gmapping algorithm was used as the SLAM algorithm, it is based on the Rao-Blackwellised Particle Filter and optimized particle number and particle degradation issues in RBPF.

The RBPF algorithm decomposes the estimation of robot pose information and the analysis of environmental features into two separate parts, pose estimation and environment estimation, which can reduce the vector's dimension and improve the operation's efficiency. The formula is below

$$p(X_{1:k}|Y_{1:k}, U_{0:k}, x(0)) = p(X_{1:k}|Y_{1:k}, U_{0:k}, x(0))p(m|X_{0:k}, Y_{0:k}) \quad (1)$$

where x is the robot pose information, and m is the environment features.

However, there are two problems with this algorithm. One problem is that to obtain the posterior probability distribution, numerous particles are required and the other is the problem of particle scarcity caused by particle degradation. To solve this, the gmapping algorithm changes the importance probability density function, using the latest sensor observation information to constrain the model and reduce particle numbers. The improved formula is the following.

$$\omega_i(k) \propto \omega_i(k-1)p(y(k)|x_i(k-1)) \quad (2)$$

Another improvement is adaptive resampling, which means setting a threshold for the effective number of particles $Qeff$. When the numbers are below the threshold, resampling is performed, so it can effectively reduce the number of resampling, thereby slowing down the degradation of particles.

2.3 Path Planning

Path planning is crucial for the automatic mobile robot. It is performed to find the possible path from the start point to the target point in the raster map, and the path should be a short distance, smooth, no collision and other constraints. The strategy here was the A* algorithm was used to find the optimal path for global path planning and dynamic window approaching for local path planning to manage the real-time obstacles.

2.3.1 A* Algorithm

To meet the requirement static environment, the A* algorithm was used. It is a heuristic search algorithm that can find a near-optimal path from the start point to the target point. It is similar to the Dijkstra algorithm but reduces the number of possible paths to improve the computer calculation efficiency. In addition, the A* algorithm is based on grid maps, which is suit for the Gmapping algorithm as it generates grid maps.

Its primary work principle is estimating the cost of non-obstacle nodes around the state point and building a priority queue depending on the cost of nodes, for each iteration. the lowest cost node is selected from the priority queue as the next start node. The function was shown below.

$$f(n) = g(n) + h(n) \tag{3}$$

where, g(n) is the actual cost from the initial node to node n, and h(n) is the estimated cost from the target node to node n.

the heuristic function h(n) has different functions depending on the number of directions that can move, this paper discussed the robot was two-wheel and could only move in four directions, so the Manhattan distance is used. The expression was written below

$$h(n) = D \times (|x(n) - x(goal)| + |(y(n) - y(goal)|) \tag{4}$$

where, D is moving cost between two nearby nodes.

2.3.2 DWA Algorithm

Global path planning A* algorithm can be used in the static environment, but in the actual hospital application scenario, there are many moving obstacles, like people, that may block a planned path by the A* algorithm. To ensure the robots can arrive at the target position, another algorithm that avoids obstacles is required to meet the dynamic environment. Therefore, dynamic window approaching was chosen for location path planning which can fine-tune the path to prevent collisions [5].

DWA algorithm is a velocity-based planning algorithm. There are two steps based on its working process. The first step is to search the speed simple range with restrictions of limited speed and limited acceleration. The formulas were shown below. Equation 5 showed the dynamics of the robot itself. Equation 6 showed that the robots would not collide with the nearest obstacle in the maximum deceleration under the current speed. Equation 7 showed that acceleration and deceleration changes have upper and lower limits, so the speed in the next state has limits. The final velocity range can be expressed as the intersection of the above three formula velocities [6].

$$V_s = \{(v, w) | v \in [v_{min}, v_{max}] \cap w[w_{min}, w_{max}]\} \tag{5}$$

$$V_a = \left\{(v, w) \middle| v \leq \sqrt{2 \times dist(v, w) \times a_v} \cap w \leq \sqrt{2 \times dist(v, w) \times a_w}\right\} \tag{6}$$

$$V_d = \{(v, w) | v \in [v_a - \dot{v} \times t, v_a + \dot{v} \times t] \cap w \in [w_a - \dot{w} \times t, w_a + \dot{w} \times t]\} \tag{7}$$

$$V_f = V_s \cap V_a \cap V_d \tag{8}$$

The second step is to choose the optimal speed, the computers can stimulate all the possible trajectories with different velocities, using the equation below to evaluate trajectories under the three criteria target heading, distance of obstacles, and velocity

$$G(v, w) = G(v, w) = \sigma(\alpha \times heading(v, w) + \beta \times (dist(v, w) + \gamma \times velocity(v, w)) \tag{9}$$

Where σ is used to smooth the weighted sum and let the result have more clearance from obstacles.

2.4 Simulation

The gazebo platform was applied to build the hospital inpatient wards and make the simulation environment more realistic. A moving person was added with the action function in Gazebo. With the help of the rviz, the map can be constructed by moving the robot around the hospital with the gmapping algorithm. The white area indicated the completed scanned area, and the grey area meant an unknown environment.

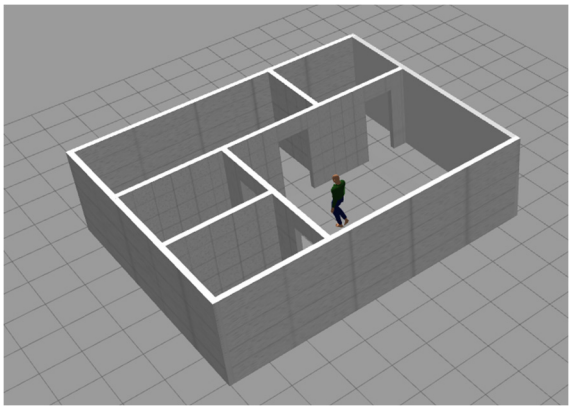


Figure 1. Building hospital environment

Figure 2. Generated map by Gmapping

It can be found that although the map was built, some part of the map it wasn't clear, and the wall shape was distorted from the original environment. This is caused by the pose estimation accuracy, and a more accurate pose sensor can solve it.

For path planning, the global and the local path planning algorithms were saved in the same package and ran at the same time and the static map built by SLAM was written into the map service to show map information. The right figures showed the selected path by the A* algorithms and the left figure showed that when the robot detected the real-time robot, the path was locally changed. In the simulation, when the robot was close to the obstacles, although on the map the path was safe, but the robot would collide due to the robot's irregular-shaped and inertia. therefore, the warning area of the edge and some speed limits were set the fix the problem.

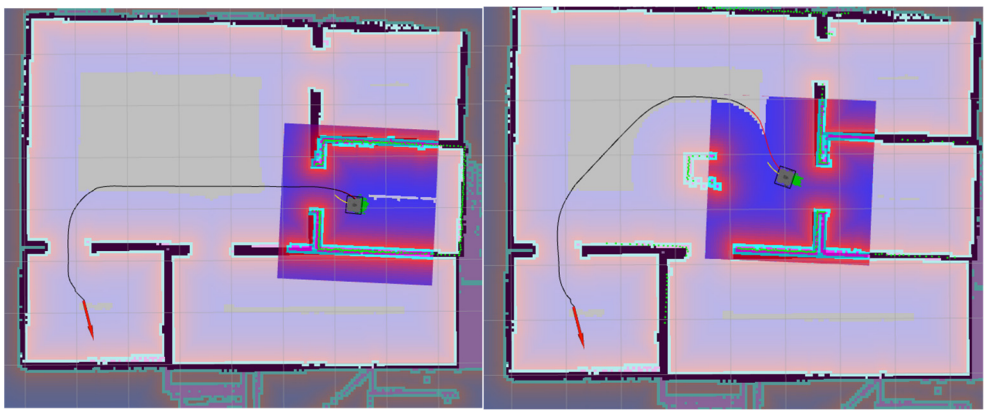


Figure 3. Path planning with dynamic obstacle

The digraph below showed the communication architecture in Gazebo. The move_base package includes global and local path planning. It is the core of the navigation function, based on data and known map information. It plans the optimal path and controls the speed. The amcl node worked with the /scan and /tf defines the global location of the moving robot and the map_serve package provides the map information.

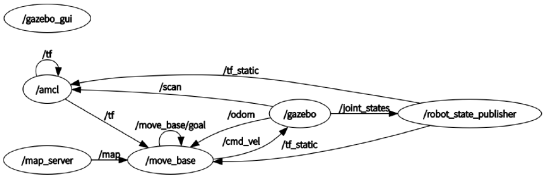


Figure 4. Communication architecture in Gazebo

2.5 Grasping System

In this paper, the chosen robot arm was OpenManipulator-X, it is an open source, and 4-degree-of-freedom manipulator and figure 7 showed the detailed information of the manipulator.

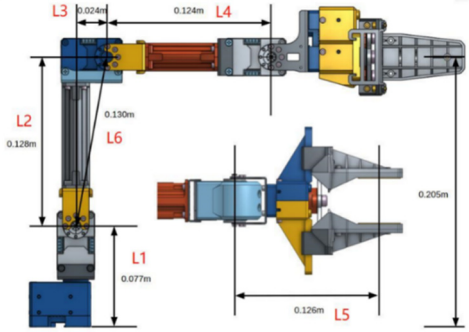


Figure 5. OpenManipulator-X model [7]

2.5.1 Kinematic Model

The forward kinematic and inverse kinematic of the robot arm would be analyzed. The forward kinematic used joint displacement and angle to obtain the position of end effector, while the inverse kinematic used known end-effector position to reverse computation of the joint displacement and the angle requirement.

The Denavit-Hartenberg (DH) described the relationship between the two relative joints. With the parameters of angle and offset of joints, and length and twist angle of links, the manipulator configuration can be fixed [8].

Table 1. The DH parameters of the OpenMANIPULATOR-X

Joint	a_i(degree)	α_i(m)	d_i(m)	θ_i(degree)
1	0	90	0.77	θ_1
2	0.13	0	0	$\theta_2 - \theta_0$
3	0.124	0	0	$\theta_3 + \theta_0$
4	0.126	0	0	θ_4

Where θ_0 is the offset angle between the L2 and L6

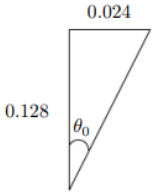


Figure 6. The θ_0 position

$$\theta_0 = \sin^{-1}\left(\frac{L_3}{L_6}\right) = 10.64° \tag{10}$$

2.5.1.1 Forward Kinematic

On the base of the DH parameters, the transformation matrix can be written. Since in this case, the robot manipulator was 4 Dof, the transformation matrices related to the initial frame and the end effector can be equated.

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & \alpha_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\sin\alpha_i\cos\theta_i & \alpha_i\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

$$T_4^0 = T_1^0 \times T_2^1 \times T_3^2 \times T_4^3 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12}$$

Where , (nx, ny, nz),(ox, oy, oz) and (ax, ay, az) are the initial frame orientation vector and the value p_x, p_y, p_z are the coordinates of the end-effector.

2.5.1.2 Inverse Kinematic

The geometry method was used in this paper to solve the inverse kinematics of the manipulator. As the position of the end-effector was known. It was easy to calculate θ_1 in the X-Y plane.

$$\theta_1 = atan2(p_y, p_x) \tag{13}$$

In order to reduce the calculation complex, the 3D coordinate was simplified to a 2D coordinate by merging the X, Y axis into a new axis r, The Pythagorean equation was used blow.

$$p_r = \sqrt{{p_x}^2 + {p_y}^2} \tag{14}$$

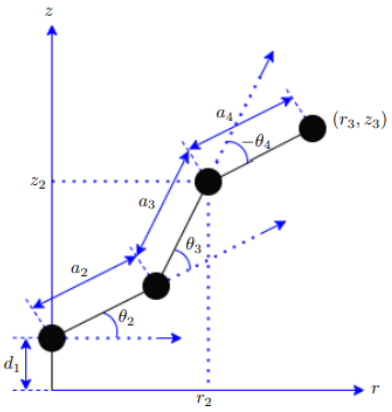


Figure 7. The r-z plane for the manipulator. [9]

Figure 9 showed the r-z plane for the robot arm and the rotation angles of the manipulator θ_2 θ_3 θ_4 were limited at the first and fourth quadrants (-90°,90°). With the given pitch angle ϕ, the θ_3 can be figured out by the law of cosines

$$r_2 = r_3 - a_4\cos\phi \tag{15}$$

$$z_2 = z_3 - a_4\sin\phi \tag{16}$$

$$\theta_3 = \pm \cos^{-1}\left(\frac{{r_2}^2+{z_2}^2-({a_2}^2+{a_3}^2)}{2a_2a_3}\right) \tag{17}$$

The θ_2 can be calculated by the following equation

$$r_2 = a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3) = \cos\theta_2\,(a_2 + a_3\cos\theta_3) - \sin\theta_2\,(a_3\sin\theta_3) \tag{18}$$

$$z_2 = a_2\sin\theta_2 + a_3\sin(\theta_2+\theta_3) = \cos\theta_2\,(a_3\sin\theta_3) + \sin\theta_2\,(a_2 + a_3\cos\theta_3) \tag{19}$$

$$\cos\theta_2 = \frac{(a_2+a_3\cos\theta_3){r_2}+(a_3\sin\theta_3){z_2}}{{r_2}^2+{z_2}^2} \tag{20}$$

$$\sin\theta_2 = \frac{(a_2+a_3\cos\theta_3){z_2}+(a_3\sin\theta_3){r_2}}{{r_2}^2+{z_2}^2} \tag{21}$$

$$\theta_2 = atan2(\sin\theta_2, \cos\theta_2) \tag{22}$$

After the calculated θ_2 and θ_3, it is easy to obtain the value of θ_4

$$\theta_4 = \phi - \theta_2 - \theta_3 \tag{23}$$

2.5.2 PD Control with Online Gravity Compensation

For N-joint robot arms, the dynamic model in the Lagrangian form can be expressed as

$$M(\theta)\ddot{\theta} + C\left(\theta,\dot{\theta}\right) + G(\theta) = \tau \tag{24}$$

Where, θ is the joint variable vector and τ is the vector of generalized forces. M $(\theta,\dot{\theta})$ is the inertia matrix, $C\left(\theta,\dot{\theta}\right)$ are Coriolis/centripetal forces, and G (θ) is the gravity vector the PD control with online gravity compensation [10]

$$\tau = K_p e - K_D\dot{\theta} + G(\theta) \tag{25}$$

Where e is $\theta_d - \theta$

To prove the globally asymptotically stable, the Quadratic Lyapunov function with positive derivative was used

$$V\left(\dot{\theta},e\right) = \frac{1}{2}\dot{\theta}^T M(\theta)\dot{\theta} + \frac{1}{2}e^T K_p e > 0\ \forall\ \dot{\theta}, e \neq 0 \tag{26}$$

Differentiating the equation with respect to time, and the $M(\theta)\ddot{\theta}$ can be solved by the equation

$$\dot{V} = \dot{\theta}^T M(\theta)\ddot{\theta} + \frac{1}{2}\dot{\theta}^T\dot{M}(\theta)\dot{\theta} - \dot{\theta}^T K_p e \tag{27}$$

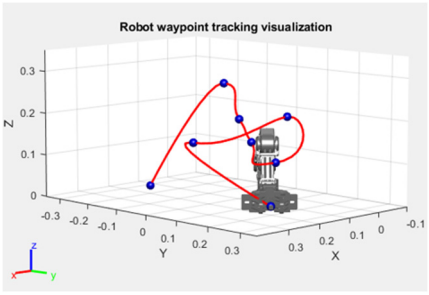
$$\dot{V} = \frac{1}{2}\dot{\theta}^T(\dot{M}(\theta) - 2C\left(\theta,\dot{\theta}\right))\dot{\theta} + \dot{\theta}^T(\tau - G(\theta) - K_p e) \tag{28}$$

The skew symmetry matrix $\dot{M}(\theta) - 2C\left(\theta,\dot{\theta}\right)$ didn't have any meanings, therefore the controller can be following

$$\dot{V} = \dot{\theta}^T\left(\tau - G(\theta) - K_p e\right) = -\dot{\theta}^T K_D\dot{\theta} \tag{29}$$

3. Results

MATLAB with Robotics System Toolbox, Simulink, and Simscape Multibody was used. Inserted the manipulator model and uploaded the algorithm file to Matlab. Then the waypoints were designed by a set of points and they created the smooth line based on these points. The robot arm can perform planned trajectory with inverse or forward kinematics. Figures 8,9 below showed the simple and complex trajectory path that manipulator performed on the simulation.



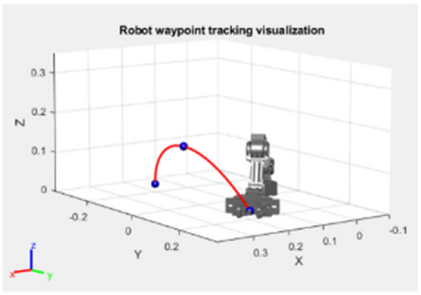


Figure 8. Simple trajectory **Figure 9.** Complex trajectory

The figures blew, simulating the robot arm picking up a ball and moving the ball into another place. To check the accuracy of the controller, the desired and actual pitch angle graphs in end effector x, y, and z axles were drawn. The result showed that with the PD controller, the error was small.

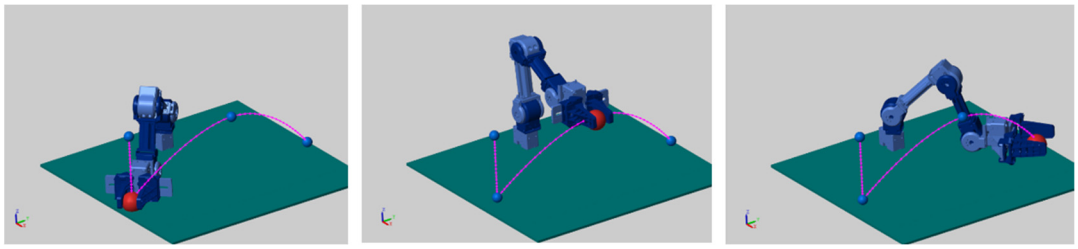


Figure 10. Robot arm to catch the ball

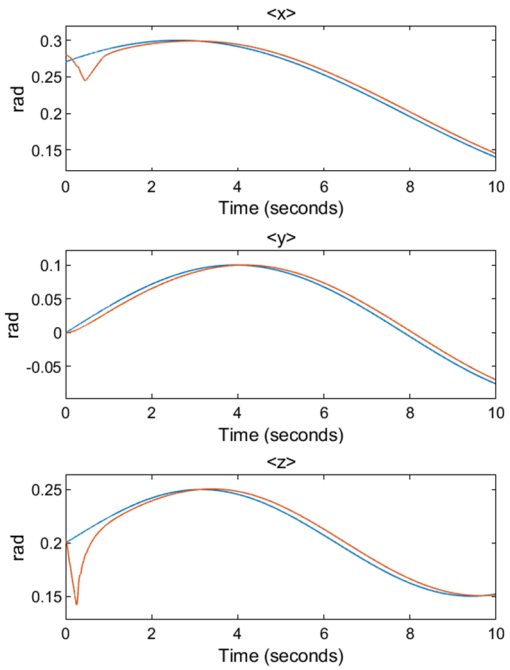


Figure 11. The design and actual angle of x,y,z axles

4. Conclusion

In this paper, the robot model was the Turtlebot with OpenManipulator-X and it was simulated in the hospital environment. There were two parts to the robot: the moving part and the grasping part. For the moving part, under the Gmapping SLAM algorithm, the robot controlled by human can walk around the environment to collect the environment information. Depending on the collected

information, robot can navigate to a target place through the path planning algorithm A* and DWA. For the grasping part, the forward and inverse kinematic of the manipulator was analyzed, through the designed trajectory with the PID control, the robot arm can pick up some simple stuff. Although there were some limitations. For example, the accuracy of the collected map information by Gmapping, the wrong path planning at the corner with a slow response rate, and the trajectory planning were designed in advance. This problem can be mitigated by using accurate hardware and optimizing algorithms. In the future, reinforcement learning like q learning can be added to the path planning to improve its reaction speed, and the Genetic algorithm can be applied to reduce the time of setting the parameters.

References

[1] Peiffer-Smadja N.:Challenges and issues about organizing a hospital to respond to the COVID-19 outbreak: experience from a French reference central, Clinical Microbiology and Infection. vol. 26, 2020, pp. 669-672.

[2] WHO.: Health and Care Worker Deaths during COVID-19, 2022. [Online].

[3] Chekubasheva V., Kravchuk O., Hlukhova H. and Glukhov O.: Creating of a remote-presence robot based on the development board Texas Instruments to monitor the status of infected patients, Biosensors and Bioelectronics. vol. 11, 2022, pp. 1002.

[4] Zhou L., Zhu C. and Su, X.: SLAM algorithm and Navigation for Indoor Mobile Robot Based on ROS, 2022 IEEE 2nd International Conference on Software Engineering and Artificial Intelligence (SEAI). 2022, pp. 230-236.

[5] Tianyu, L., Ruixin Y., Guangrui W. and Lei S.: Local Path Planning Algorithm for Blind-guiding Robot Based on Improved DWA Algorithm, 2019 Chinese Control and Decision Conference (CCDC), 2019, pp. 6169-6173.

[6] Fox D., Burgard W. and Thrun S.: The dynamic window approach to collision avoidance, IEEE Robotics & Automation Magazine. vol. 4, 1997, pp. 23-33.

[7] ROBOTISe-Manual.: Open MANIPULATOR-X, 2022. [Online].

[8] Mohd Zaman M. H., Ibrahim M. F.: Dimensional Optimization of 4-DOF Robot Manipulator Using Artificial Bee Colony Algorithm, 2021 International Conference on Electrical Engineering and Informatics (ICEEI). 2021, pp. 1-4.

[9] Ting HM. Zaman M., Ibrahim M. and Moubark A.: Kinematic Analysis for Trajectory Planning of Open-Source 4-DoF Robot Arm, International Journal of Advanced Computer Science and Applications. vol. 12, , 2021, pp. 769-777.

[10] Sciavicco L. and Siciliano B.: Modelling and Control of Robot Manipulators Modelling and Control of Robot Manipulators, Industrial Robot: An International Journal. vol. 25, 1998, pp. 73-73.