

Autonomous Indoor Delivery Robot using Inertial Measurement Unit and 2D LIDAR

Duy Bach Nguyen, Khang Di Nguyen, Nguyen Hoang Khoi Tran

Department of Control Engineering and Automation

Ho Chi Minh City University of Technology (HCMUT), VNU-HCM

Ho Chi Minh City, Vietnam

bach.nguyenndbach@hcmut.edu.vn, khang.nguyen19820@hcmut.edu.vn, thknguyen@hcmut.edu.vn

Abstract— Autonomous delivery system is the new heated trend in many developed countries due to high demand in urban environments such as universities, dorms and residential area. In this paper, we present an indoor autonomous delivery robot using 2D LIDAR and inertial measurement unit (IMU). To achieve good fusion, we have to calibrate the IMU, set up the hardware configuration for the robot and set up ROS parameters to synchronize the nodes together. The calibration process was carefully recorded, measuring in order to give correct data for further fusion in the robot system. The robot physical model will be described carefully to map the frames together. After preparing all the key criteria, we will perform localization process by calling ROS localization package with parameters modifying and tuning to enhance system performance. This ROS algorithm generates decisions based on global and local modules that create optimal path to the destination point with object avoidance requirement. This work has been validated by a test inside our university area, where our robot has performed indoor navigation from the starting point to the destination without any collision.

Index Terms— autonomous mobile robot, navigation system, IMU, LIDAR, ROS

1 INTRODUCTION

The visual navigation [1] is a very heated topic in recent years. We can divide visual devices into 3 hardware types, namely monocular, binocular and depth cameras. This technology which uses cameras to generate maps for data fusion purpose is called Simultaneous Localization and Mapping (SLAM) technology. The stereo camera which used in a lot of localization process (e.g., A Robust Stereo Camera Localization Method with Prior LiDAR Map Constrains paper [2] or Stereo Camera Localization in 3D LiDAR Maps[3] have mentioned the use of stereo system in Localization system) can achieve a very good result in planning process after calculating a lot of feature parameters in the environmental image obtained by the camera however the enormous computational cost is not

necessarily needed for the building of an indoor mobile robot so we will use a RPLIDAR 2D instead. The LiDAR we use in this project can do both self-localization and path planning. However, in the localization process, there are noises when doing rotation and moving forward. Hence, we need to fusing data from other sensors [4] namely roll, pitch, yaw angles Inertial Measurement Unit (IMU) and linear and angular velocity provided by two encoders. The LiDAR sensor creates laser scan frame, the IMU provides data for tf frame and the signal from encoders will be handled by 32-bit microcontroller integrated circuit STM32F411 to output angular and linear velocity to Jetson Nano B1, which will be used to calculate the left and right wheel velocity for the transmission and receive between cmd_vel node in ROS and 2 wheels.

Our autonomous delivery system based its decisions on global and local calculations. These calculations will help our robot find the most optimize path within the minimum computational time and discover static and moving object in indoor environments. The robot uses the STM32F411 microcontroller to control the motor, read data from the IMU sensor and communicate with the Jetson Nano embedded computer using the UART communication. The Jetson Nano embedded computer is used to read data from the RPLiDAR A1 sensor, and this will be the robot's brain, solving the problems outlined above.

In this paper, we focus mainly on the use of autonomous delivery system on indoor environment so we only need the help mainly on LiDAR in both global and local path finding algorithms. The global calculation required a stored map, which requires the Simultaneous Localization and Mapping (SLAM) method since we can't use global positioning system in indoor environments. Sensor fusion process will be occurred at different nodes in ROS, depends on the function of that node in the general operation. Each indoor localization system has different ways to combine the sensors

together to generate different data for processing purpose (e.g., the used of Extended Kalman Filter to fuse difference sources of odometry sources into a single source [5] or IMU and fisheye camera data fusing [6]).

The remainder of this paper is organized as follow. Section 2 introduce the hardware system and algorithms used to calculate the kinematic of the robot. Section 3 provide a brief overview of the IMU calibration process with the include of calibrated parameters needed. Section 4 give us a deep knowledge about what are inside the ROS path planning algorithms [7] including the global and local planner. Section 5 demonstrates the experimental results of our indoor localization system. Finally, the work conclusion will be held in section 6.

2 METHODOLOGY

2.1 Vehicle kinematic

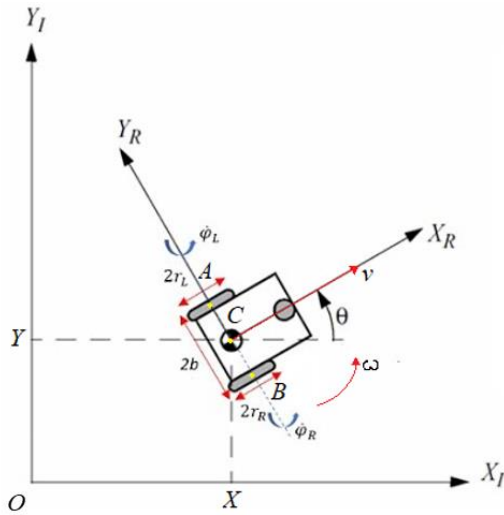


Fig. 1. The robot is placed in an Inertial frame (I). The frame attached to the robot is Robot frame (R).

Pose of the robot in inertial frame in Fig. 1 is determined by 3 parameters: $[X \ Y \ \theta]^T$.

- X, Y : coordinates of the robot relative to the inertial frame.
- θ : angle between X_R axis and X_I axis.
- r_R, r_L : radius of wheel.
- b : distance from robot center to wheel.
- $\dot{\phi}_R, \dot{\phi}_L$: angular velocity of wheel.

- v : linear velocity of the robot.

- ω : angular velocity of the robot.

Linear velocity and angular velocity are calculated by:

$$v = \frac{v_L + v_R}{2} = \frac{r_L \times \dot{\phi}_L + r_R \times \dot{\phi}_R}{2} \quad (1)$$

$$\omega = \frac{v_R - v_L}{2b} = \frac{-r_L \times \dot{\phi}_L + r_R \times \dot{\phi}_R}{2b} \quad (2)$$

From (1) and (2) we have:

$$v_R = v + \omega b \quad (3)$$

$$v_L = v - \omega b \quad (4)$$

DWA algorithm will calculate the linear velocity and angular velocity, from the kinematic model, we will calculate the speed set on 2 wheels.

2.2 IMU calibration

Calibrated by software Magneto1.2, using the method “Least squares ellipsoid specific fitting” researched by Dr. Quing De Li and published in 2004. This method is used to calibrate both accelerometer and magnetometer. [9]

The calibrated data is calculated according to the formula:

$$\begin{pmatrix} ax \\ ay \\ az \end{pmatrix}_{calib} = \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ s_{21} & s_{22} & s_{23} \\ s_{31} & s_{32} & s_{33} \end{pmatrix} \times \begin{pmatrix} ax_{raw} - b_x \\ ay_{raw} - b_y \\ az_{raw} - b_z \end{pmatrix} \quad (5)$$

$$Roll = \arctan\left(\frac{ay}{az}\right) \quad (6)$$

$$Pitch = \arctan\left(\frac{-ax}{\sqrt{ay^2 + az^2}}\right) \quad (7)$$

$$Mx = mx \times \cos(pitch) + mz \times \sin(pitch) \quad (8)$$

$$My = mx \times \sin(roll) \times \sin(pitch) + my \times \cos(roll) - mz \times \sin(roll) \times \cos(pitch) \quad (9)$$

$$yaw = \arctan\left(\frac{My}{Mx}\right) \quad (10)$$

With $ax, ay, az, gx, gy, gz, mx, my, mz$ are the axes, respectively x, y, z of accelerometer, gyroscope and magnetometer.

3 ROS PATH PLANNING PROCESS

3.1 GLOBAL PATH PLANNER

Global planning process requires the entire map of the surrounding environment we are trying to put the robot in. To analyze and generate a map, Hector Slam method is being used. In this paper, we will use the navigation package provided by ROS as a core structure for the project. There are many global calculation methods [11] but in this global planner package, we use the A* method which we will discuss further below.

Let divide the surface we are trying to do the global planning calculation into several small grid. We have to find the most optimize path, hence the least movement cost path from our robot grid to the destination grid. The A* algorithm divide our math problem into several smaller local tasks. Each task contains a sum of the movement cost to move from the initial point to a random node generated by ROS algorithm and the movement cost to move from that node to the final destination. After examined all the potential nodes, the chain of nodes with minimum **D** will be picked to create the most optimize path to our goal. Let the former parameter be **h** and the latter parameter be **g** and the final cost of the total movement be **D**, we have:

$$D = h + g \quad (11)$$

We can calculate the **h** parameter but the calculation of a chosen grid to the final destination is very time consuming. Therefore, in this paper, an approximation method is used to calculate **g**. There are 3 approximation heuristics to choose from, namely Manhattan Distance, Diagonal Distance and Euclidean Distance approximation will be described in Fig. 2. We personally choose Euclidean Distance (Fig. 3) method since it matches with the system we design in term of accuracy and rapidity.

Let the given grid's x and y coordinates be x_1 and y_1 . This will also be our initial position for the first loop. The final destination grid's x and y coordinates be x_d and y_d , the **g** approximation by Euclidean Distance algorithm can be described as:

$$g = \sqrt{(x_1 - x_d)^2 + (y_1 - y_d)^2} \quad (12)$$

After map scanning process as will be described further in the Result section, the dots which represents the object in the map that was detected by laser scan from LiDAR, will be circled around to create a new spherical object in 3D dimension. The

radius of the circle will be set manually to 20cm, since our LiDAR can't detect the edge of an object which is lower than the LiDAR position and locate quite close to our robot.

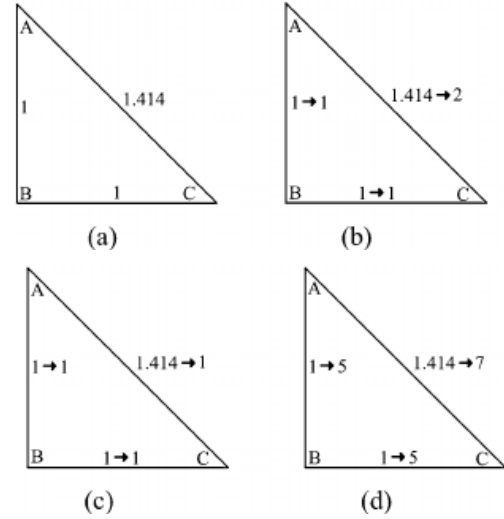


Fig. 2: Approximation of Euclidean distance. (a) Euclidean distance. (b) Approximation by Manhattan distance. (c) Approximation by Chessboard distance. (d) Approximation by Chamfer 5 / 7 distance [10]

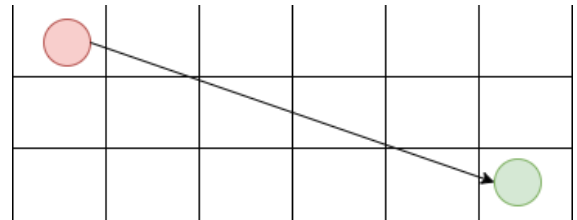


Fig. 3. The path figure by Euclidean Distance Approximation

After figure out the chain of circle nodes that create the great path to our destination, we have to draw the path again. Since our **g** calculation is only based on Euclidean Distance Approximation, which is a zero-order derivative function, it can't be apply to such long distance since the error in calculation will vary unpredictably. To tighten the path, we add the quadratic approximation algorithm to the process. This algorithm will add first-order and second-order approximation to our initial zero-order function, which will reduce the error in calculation by eliminate the nodes created by noise along the path way and sharpen the path to our goal as seen in Fig. 4 and Fig. 5 [12] [13].

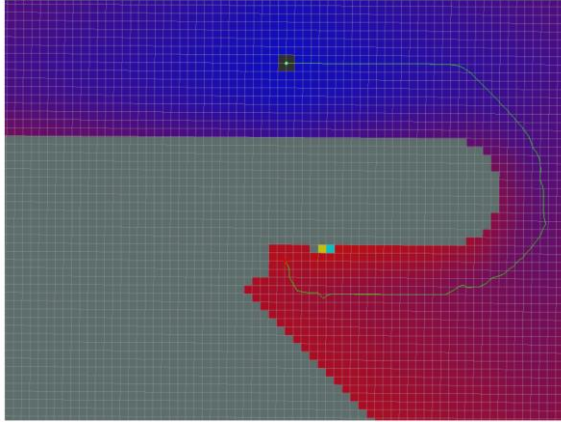


Fig. 4. Path finding without quadratic approximation

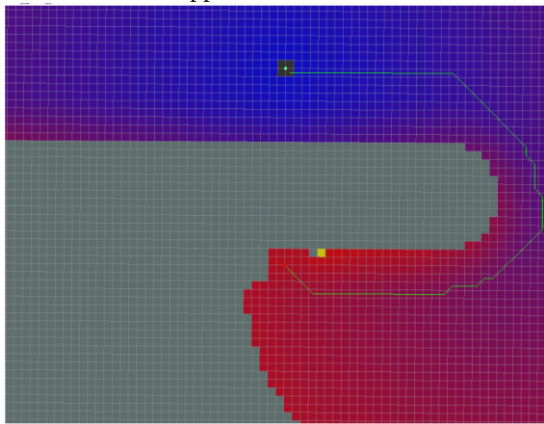


Fig. 5. Path finding with quadratic approximation

3.2 LOCAL PATH PLANNER

For local planning, we use base_local_planner ROS package. It is based on trajectory rollout and dynamic window approach which is a common approach for local control. The planner worked well with static objects so it fitted in easily with the indoor environment we examined. The used of DWA algorithm [12] to create local planning movements will be explained in Fig. 6.

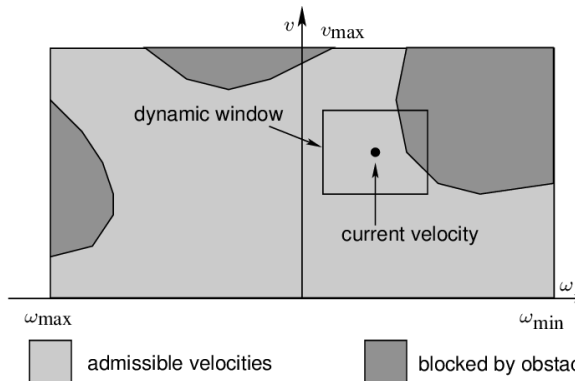


Fig. 6. Dynamic window approach for object avoidance

As seen in the Fig. 6, The DWA algorithm will generate a dynamic rectangular to cover the moving robot. The final path to achieve to the short-term goal will be calculated by generating pairs of solutions contain 2 parameters, the new position of our robot compares to our current position and the pose of itself when it gets to the new position. After multiple case generated, the algorithm will choose the best local new positions for the robot to dodge objects and at the same time after finish avoiding new obstacles, the robot will return to the path that generated by global calculation process.

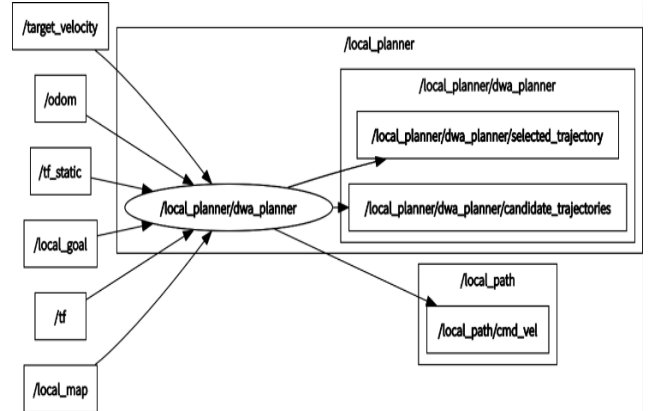


Fig. 7. Connection between local_planner package to nodes in ROS

Fig. 7 describes the construction of dwa local planner algorithm. The dwa_planner node get required velocity data from /target_velocity node, linear velocity, angular velocity and x,y position from odom node, transformation between different axis systems in /tf and /tf_static, the final required position from /local_goal and map data base from /local_map.

4 EXPERIMENTAL RESULT

The Kalman filter algorithm consists of two main processes: the prediction process and the adjustment process as seen in Fig. 8.

We will test this Kalman on many different data sets to see how effective it is. One of the results is described in Fig. 10 and Fig. 11 below. Both of the graphs show the left and right motor speed comparing to the set point velocity when we put a load of 18kg into the robot system.

Since both motors have large weight inertia, the increase in load mass causes more stabilize in velocity. In the graph when the speed change suddenly.

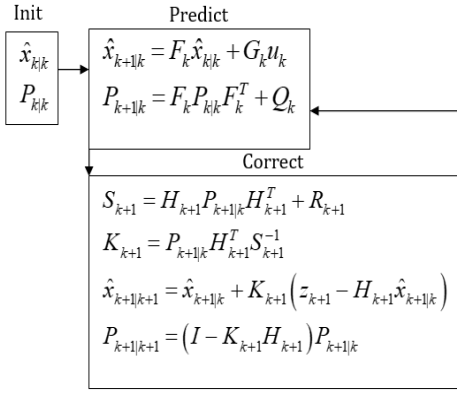


Fig. 8. Summary of Kalman algorithm

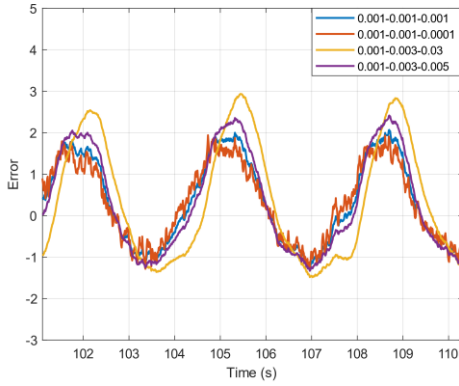


Fig. 9. Error (degree) between estimated angle – encoder when changing the set of Kalman numbers

It can be seen in Fig. 9 that the smaller r is, the smaller the error, but it fluctuates a lot, because the yaw angle of the measurement fluctuates a lot. The set of numbers $Q_\theta = 0.001$, $Q_{\dot{\theta}} = 0.003$, $R = 0.005$ has a satisfactory error and is not too volatile, so we will choose this as the set of numbers for the Kalman filter.

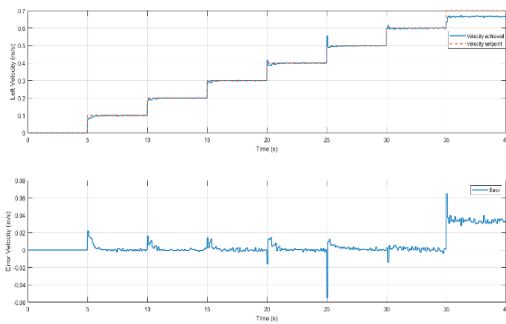


Fig. 10. PID response of left motor when 18kg of load is carried

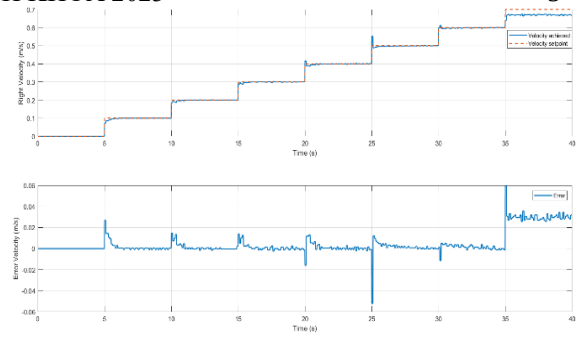


Fig. 11. PID response of right motor when 18kg of load is carried

In this paper, we use Hector SLAM method to create map for global planner purpose. This method only requires data from laser scan's source. Hector SLAM system subscribes to the sensor_msgs/LaserScan message and publishes nav_msgs/OccupancyGrid, tf transformation and the pose with covariance message geometry_msgs/PoseWithCovarianceStamped. The small dots appear in the map presents objects that have been set up for testing purpose.

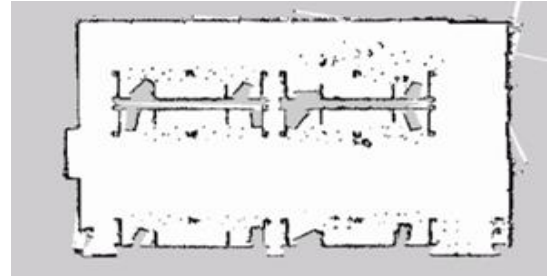


Fig. 12. Map stored after using SLAM from LiDAR

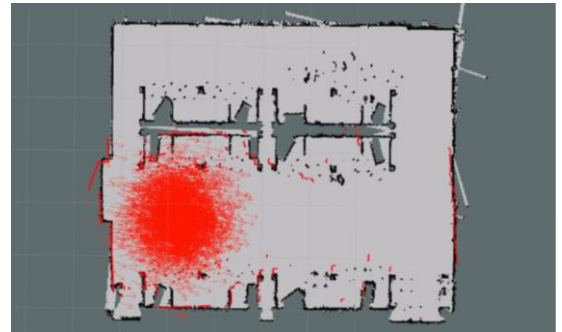


Fig. 13. Initializing AMCL tuning process

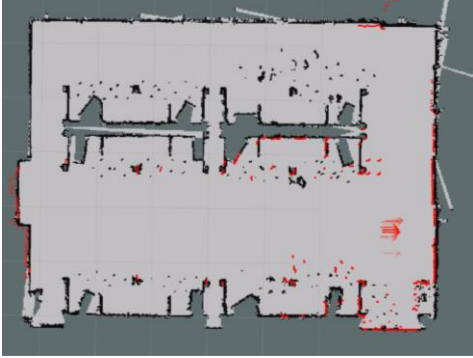


Fig. 14. AMCL particles merging process completed

Before doing navigation works, we have to tune the AMCL parameters. The tuning process happens by changing the rotation and forward error parameters when the vehicle is moving forward or rotating. In Fig. 13. describes the beginning state of the vehicle when AMCL particles are not converged. After tuning process, in a short period of time, as seen in Fig. 14., the AMCL particles are emerged and the vehicle can localize its current position in the map more precisely. The distance for this work to be done after tuning is between 3 meters to 5 meters and the time estimation is 6 seconds to 8 seconds.

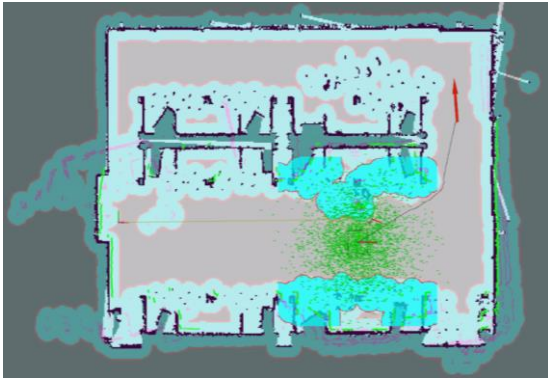


Fig. 15. Initializing navigation process

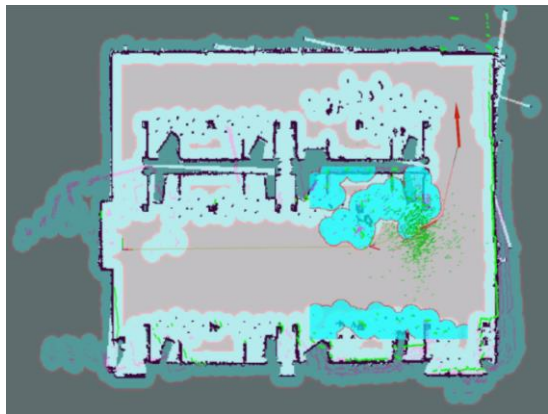


Fig. 16. New object appears in the path

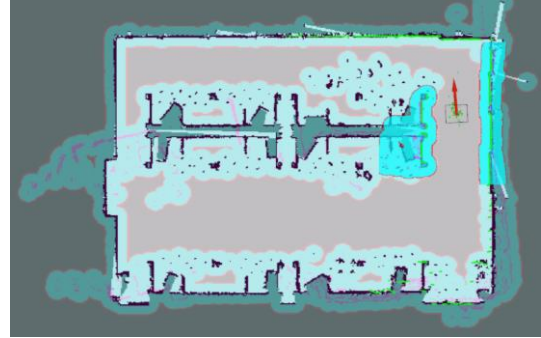


Fig. 17. The robot has reached its required destination

After all the setup has completed, the navigation process will be started. As seen in Fig. 15., we only need to choose the direction and position of the destination by dragging the red pointer in the map, the model will return the most optimize path by using global calculation. When moving forward, the pale cyan part represents objects after map scanning process while the brighter cyan part represents the object detection by our LiDAR sensor, which will become fusion for local calculation. The path calculation and destination reaching process happen quite smoothly under good condition, however, in such situation when object appears quite sudden and the distance between object to our indoor mobile system is too close, the local planning algorithm from ROS is slow to react. When new object appears that make the previous planning calculation inaccurate, the recalculation process appear as seen in Fig. 16. to avoid collision. As seen in Fig. 17., with no new object appears in the map, in only a short period of time, the robot has come to the goal with the same pose as instructed.

5 CONCLUSION

In this work, we presented an indoor mobile localization robot for delivery small packages from one place to another inside campus area. The IMU calibration algorithm happens smoothly to give accurate yaw parameters for fusing purpose in ROS packages while the vehicle dynamics have been defined precisely in order to input correct positions for different frames using in ROS. Real-world experiments showed that the LiDAR has successfully finish its works required by the system and the robot has generated a very accurate path to the goal in a short of time. After fail to follow global plan suggested by the system, the robot immediately create new path to avoid objects due to the furious update of local costmap which prove the great sensor fusion and ROS nodes linkage. Further research will combine GPS and stereo system in

order to track lane in outdoor environments and detect pedestrians to avoid any collision from occurring.

ACKNOWLEDGMENT

“This research is funded by Ho Chi Minh City University of Technology – VNU-HCM under grant number SVCQ-2022-ĐĐT -69”. We acknowledge the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for this study.

REFERENCES

- [1] Jonáš Kulháněk, Erik Derner, Robert Babuška. “Visual Navigation in Real-World Indoor Environments Using End-to-End Deep Reinforcement Learning.” *arXiv preprint arXiv:2210.10903*
- [2] Dong Han, Zuhao Zou, Lujia Wang, Cheng-Zhong Xu. “A Robust Stereo Camera Localization Method with Prior LiDAR Map Constrains” *arXiv preprint arXiv:1912.05023*
- [3] Y. Kim, J. Jeong and A. Kim, "Stereo Camera Localization in 3D LiDAR Maps," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018, pp. 1-9, doi: 10.1109/IROS.2018.8594362.
- [4] N. Cong Khoa and P. Xuan Minh, "An Adaptive Filter for IMU/Encoder Data Fusion for Acceleration Estimation in Robot Arms," *2018 International Conference on System Science and Engineering (ICSSE)*, New Taipei, Taiwan, 2018, pp. 1-6, doi: 10.1109/ICSSE.2018.8520128.
- [5] Yan, Y.; Zhang, B.; Zhou, J.; Zhang, Y.; Liu, X. Real-Time Localization and Mapping Utilizing Multi-Sensor Fusion and Visual-IMU-Wheel Odometry for Agricultural Robots in Unstructured, Dynamic and GPS-Denied Greenhouse Environments. *Agronomy*, pp. 7-8, 2022, 12, 1740
- [6] Uppunda, Praveen L., Vinayaka Raju, and C. Gururaj. "Autonomous Vehicle Navigation with LIDAR using Path Planning." *arXiv preprint arXiv:2212.07155*, pp. 3-4, 2022
- [7] http://wiki.ros.org/robot_localization
- [8] Lai, X., Wu, D., Wu, D., Li, J.H. 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>
- [9] Fakoorian, Seyed & Palieri, Matteo & Santamaria, Angel & Guaragnella, Cataldo & Simon, Dan & Agha-Mohammadi, Ali-Akbar, “Maximum Correntropy Kalman Filter for Orientation Estimation With Application to LiDAR Inertial Odometry.”, 2020, 10.1115/DSCC2020-3256.
- [10] A Linear Time Algorithm of Computing Hausdorff Distance for Content-based Image Analysis - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Approximation-of-Euclidean-distance-a-Euclidean-distance-b-Approximation-by_fig1_251301980 [accessed 30 Jan, 2023]
- [11] http://wiki.ros.org/global_planner
- [12] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.
- [13] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance", *IEEE Robotics Automation Magazine*, 1997