VOLUME 16, N° 1 2022
Journal of Automation, Mobile Robotics and Intelligent Systems

Development of an Omnidirectional AGV by Applying ORB-SLAM for Navigation Under ROS Framework

Submitted: 10th October 2021; accepted: 8th February 2022

Pan-Long Wu, Jyun-Jhen Li, Jin-Siang Shaw

DOI: 10.14313/JAMRIS/1-2022/2

Abstract:
This paper presents the development of an automated guided vehicle with omni-wheels for autonomous navigation under a robot operating system framework. Specifically, a laser rangefinder-constructed two-dimensional environment map is integrated with a three-dimensional point cloud map to achieve real-time robot positioning, using the oriented features from accelerated segment testing and a rotated binary robust independent elementary feature detector-simultaneous localization and mapping algorithm. In the path planning for autonomous navigation of the omnidirectional mobile robot, we applied the A global path search algorithm, which uses a heuristic function to estimate the robot position difference and searches for the best direction. Moreover, we employed the time-elastic-band method for local path planning, which merges the time interval of two locations to realize time optimization for dynamic obstacle avoidance. The experimental results verified the effectiveness of the applied algorithms for the omni-wheeled mobile robot. Furthermore, the results showed a superior performance over the adaptive Monte Carlo localization for robot localization and dynamic window approach for local path planning.*

Keywords: *ROS, ORB-SLAM, Omnidirectional AGV, Autonomous Navigation*

1. Introduction

With the advancement of technology, the production process has gradually developed automation, which has improved the production efficiency and quality of products. Consequently, robots have become an indispensable part of industrial automation. Robot manipulators and mobile platforms are key elements in robot automation. To enable a mobile robot to navigate autonomously, it is first necessary for the robot to construct an environment map and locate itself in the map. For this, the simultaneous localization and mapping (SLAM) technique is applied, which can be divided into three-dimensional (3D) visual SLAM (VSLAM) and two-dimensional (2D) laser SLAM, based on the sensor types. Specifically, 3D SLAM techniques, such as oriented features from accelerated segment test and rotated binary robust independent elementary feature detector-SLAM (ORB-SLAM), use visual scanning to match 3D point clouds [1, 2]. Gmapping, which was proposed by Grisetti et al. [3], is commonly adopted in 2D SLAM techniques. Path planning in robot navigation also needs to specify a goal on the map such that the robot will move forward along the specified path and simultaneously avoid obstacles. Currently, the most widely used algorithm for global path planning is the A* algorithm, which was proposed by Bostel and Sagar [4]. The dynamic window approach (DWA) or timed-elastic-band (TEB) method is employed as a local path planning algorithm for dynamic obstacle avoidance [5, 6]. Autonomous navigation for a mobile robot can be made possible only after the aforementioned processes have been completed.

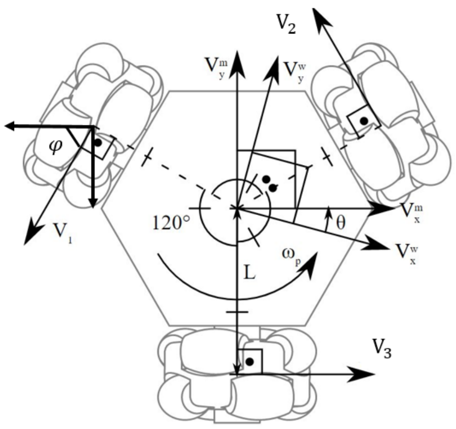
Mobile robots may have different types of traction. The most common types are differential traction, tricycles, and omnidirectional. An omnidirectional mobile robot can rotate 360° and is capable of X-axis transverse motion, thereby offering a higher degree of freedom than a differential wheel robot [7, 8]. Although there is abundant research on the autonomous navigation of mobile robots with differential wheels using SLAM techniques under a robot operating system (ROS) framework [9, 10], studies on autonomous navigation for omnidirectional mobile robots are relatively rare [11]. In this study, we investigated autonomous SLAM navigation for an omni-wheeled mobile robot under an ROS framework. In particular, 3D ORB-SLAM for robot precision positioning, along with A* and TEB algorithms for path planning, was emphasised for better performance. To the best of the authors' knowledge, this is the first application of this kind for an omnidirectional mobile robot in the literature. The experimental results were evaluated with those obtained by the well-known 2D localization methods of adaptive Monte Carlo localization (AMCL) [12] and encoder odometers [13].

2. Experimental Methods

To navigate an automated guided vehicle (AGV) smoothly to a desired location, an ROS navigation stack was employed. We set a target point within the map first. Then, a costmap was generated through a static map with inflation layer constructed around the boundary and obstacles. By applying a localization algorithm to locate the robot in the map, the goal could be reached with global path planning, while avoiding obstacles through local path planning.

2.1 Kinematics Analysis of the Omnidirectional Robot

The AGV used in this research is a three-wheeled omnidirectional robot, and the kinematic model [14, 15] and constructed robot are shown in Figure 1.



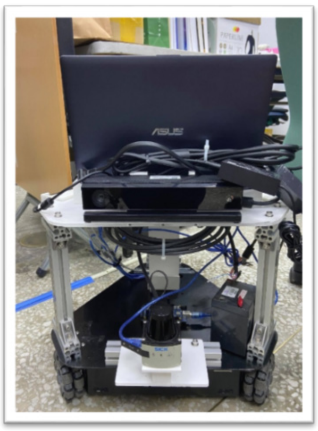


Fig. 1. Omnidirectional robot

The angle between the three wheels of the AGV is 120°, forming an equilateral triangle; thus, the distances from the center of the robot to the three wheels are equal. V_1, V_2, and V_3 are the linear speeds of the left, right, and rear wheels, respectively; L is the distance from the center of the vehicle to the center of the wheel; and φ (= 60°) is the angle between wheel 1 (wheel 2) and the X-axis of the robot coordinate system. Assume that the linear speed of the AGV at any moment is V_x^m in the X-direction and V_y^m in the Y-direction (the heading direction), and ω_p is the angular speed of the AGV center point relative to the robot frame. The kinematics equations for each wheel can be derived as follows:

$$V_1 = -V_x^m \cos\varphi - V_y^m \sin\varphi + L\omega_p \tag{1}$$

$$V_2 = -V_x^m \cos\varphi + V_y^m \sin\varphi + L\omega_p \tag{2}$$

$$V_3 = V_x^m + L\omega_p \tag{3}$$

Subsequently, they can be written in a vector-matrix form as follows:

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} -\cos\varphi & -\sin\varphi & L \\ -\cos\varphi & \sin\varphi & L \\ 1 & 0 & L \end{bmatrix} \begin{bmatrix} V_x^m \\ V_y^m \\ \omega_p \end{bmatrix} \tag{4}$$

Equation 4 represents the kinematics of the AGV in its coordinate system. The AGV center point velocity $\mathrm{AGV}[V_x^m, V_y^m, \omega_p]$ is the "cmd-vel" output from the move_base package, a path-planning package in ROS framework [16]. However, the actual speeds of the three omni-directional wheels detected by the attached encoders may deviate from the target speeds V_1, V_2, and V_3 in Equation 4 during the navigation task; therefore, a proportional-integral-derivative (PID) controller [17] is applied to each wheel's motor to compensate for the error, thereby ensuring a precision AGV motion control. A corresponding AGV speed control node graph in ROS structure is configured to implement such a closed-loop motor control.

2.2. 2D SLAM

Several open-source SLAM algorithms exist in the ROS framework. Various 2D SLAM methods were compared, and the Gmapping algorithm was proven the best in constructing a corridor environment [18]. Moreover, the localization algorithm often used the AMCL and encoder odometer.

The center position and yaw angle of the AGV can be expressed as $q_w = (x, y, \theta)$ in world coordinates. In encoder odometer localization, the omni-wheel speeds V_1, V_2, and V_3 are detected first by

$$V_{1,2,3} = \frac{100}{PPR} \times \frac{1}{t} \times 2\pi R \tag{5}$$

where *PPR* (= 500 in this study) is the number of pulses per revolution of the attached encoder, *t* is the time passed for every 100 pulses, and *R* is the radius of the wheel. After the actual wheel speeds are obtained from Equation 4, the X-axis speed, Y-axis speed, and angular speed of the vehicle relative to the robot frame can be computed as follows:

$$\begin{bmatrix} V_x^m \\ V_y^m \\ \omega_p \end{bmatrix} = \begin{bmatrix} -\cos\varphi & -\sin\varphi & L \\ -\cos\varphi & \sin\varphi & L \\ 1 & 0 & L \end{bmatrix}^{-1} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} \tag{6}$$

Then by integrating Equation 6 for a unit time, we obtain

$$\begin{aligned} \Delta d_x &= \int_{t-1}^{t} V_x^m \cdot dt \\ \Delta\theta &= \int_{t-1}^{t} \omega_p \cdot dt \end{aligned} \tag{7}$$

where Δd_x is the displacement in the X-axis, Δd_y is the displacement in the Y-axis, and $\Delta\theta_x$ is the angular displacement per unit time of the vehicle in robot frame. Subsequently, the X-axis and Y-axis displacements of the AGV per unit time in world coordinate can be calculated as follows:

$$\begin{aligned} \Delta x &= \Delta d_x \cos(\theta) - \Delta d_y \sin(\theta) \\ \Delta y &= \Delta d_x \sin(\theta) + \Delta d_y \cos(\theta) \end{aligned} \tag{8}$$

Finally, the robot center position and yaw angle are constantly updated as follows:

$$x = x + \Delta x \quad y = y + \Delta y \quad \theta = \theta + \Delta\theta \tag{9}$$

The AMCL localization algorithm distributes particles uniformly on the map and simulates the movement of the robot through the particles. Assuming that the robot moves forward a unit distance, the particles also move the same way. The position of the particles in the environment was used to simulate the sensor information and compare it with the sensor information from the one placed on the AGV. Each particle is thus given a probability based on the comparison, and the algorithm redistributes the particles according to this probability. A particle with a higher probability will have more particles around it. In this iterative process, all the particles tend to converge to a location, and the position of the robot is identified accordingly.

2.3. 3D SLAM

In this experiment, visual SLAM used the ORB-SLAM algorithm, where a red-green-blue-depth camera was utilized to obtain abundant environmental information. The localization of ORB-SLAM is to use visual odometry to track unmapped areas and to match their feature points. The system sees robust to severe motion clutter, allows wide baseline loop closing and relocalization, and is capable of automatic map initialization. There are three main aspects to ORB-SLAM that were executed simultaneously: tracking, local mapping, and loop closing. After confirming the loop closing, the system enters the global bundle adjustment (BA) optimization to form a 3+1 parallel threading. In the tracking part, localization of the camera is completed according to the ORB features of the image, and the time needed for inserting a keyframe is determined as well. Local mapping is responsible for processing new keyframes, using local BA to the camera position, and then filtering the inserted keyframes and removing redundant keyframes. Loop closing is performed by searching for loops with every new keyframe to confirm whether or not a closed loop is formed, computing a similarity transformation that provides information about the drift errors accumulated in the loop and merging nearby duplicate points. Finally, the global BA optimizes the camera position and map point simultaneously to achieve the best results.

The ROS node graph of ORB-SLAM is shown in Figure 2. The /kinect2 node is responsible for describing the coordinate relationship between the base link of the AGV and the Kinect-v2 camera. Both the color image information (/image_colour_rect) and image depth information (/image_depth_rect) with quad high definition pixels are used as the two messages into the ORB-SLAM algorithm, from which the point cloud information of the map (/map_points), positioning information of the AGV (/pose), and result of the image processing (/image_topics) are calculated. Finally, the AGV position is inputted into the odometer to calculate the relative displacement between the map and AGV, and the current coordinates of the map and AGV are updated accordingly.

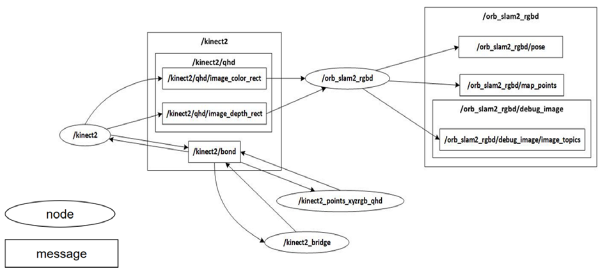
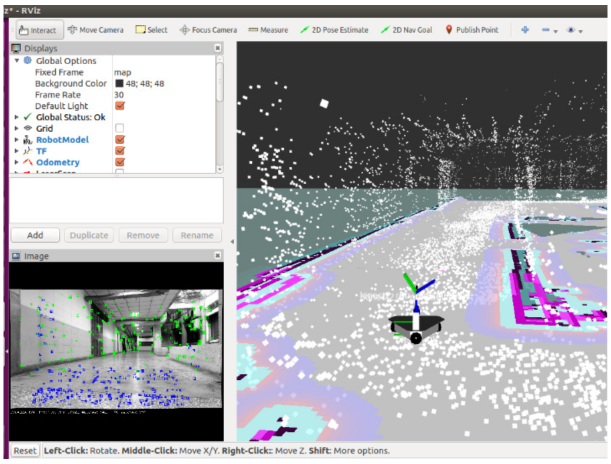

/kinect2
/kinect2/qhd
/kinect2/qhd/image_color_rect
/kinect2/qhd/image_depth_rect
/kinect2
/kinect2/bond
/orb_slam2_rgbd
/orb_slam2_rgbd
/orb_slam2_rgbd/pose
/orb_slam2_rgbd/map_points
/orb_slam2_rgbd/debug_image
/orb_slam2_rgbd/debug_image/image_topics
/kinect2_points_xyzrgb_qhd
/kinect2_bridge
node
message

Fig. 2. Node graph of ORB-SLAM

Under the ROS 3D visualization tool (RVIZ), this experiment used the point cloud map constructed by ORB-SLAM and static map constructed by Gmapping, such that the two maps are overlapped through coordinate transformation, as shown in Figure 3. Specifically, the same origin was set for the two maps and their boundaries were made to overlap, such that the difference between 2D and 3D SLAM could be intuitively compared. The actual scene captured by the Kinect-v2 camera was shown in the lower left corner of Figure 3. The green points were the ORB features after image processing, and the blue points were obtained by visual odometry which were to match the ORB features to locate the AGV within the map in the localization mode.



Global Options
Fixed Frame map
Background Color 48; 48; 48
Frame Rate 30
Default Light ✓
Global Status: Ok
Grid ☐
RobotModel ✓
TF ✓
Odometry ✓

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options.

Fig. 3. Point cloud map overlapped with a static map

2.4. Path Planning

The move_base in ROS is a package for path planning, which includes a global planner, local planner, global costmap, local costmap, and recovery behaviour node. The global costmap establishes a path plan in the entire area and generates an inflation layer on the boundary of the static map. In the local costmap, there is an inflation layer that is generated around obstacles near the AGV, limiting the accessible part of the entire area. When a kidnapped robot problem occurs, it triggers recovery behaviours: rereading the sensor messages to update the map, deleting the invalid obstacle layers, and planning the route again.

The two algorithms most used in the move_base package for global path planning are Dijkstra and A* [19], which are extremely similar. The only difference between the two is that A* attempts to find a better path using a heuristic function, while Dijkstra's explores all possible paths. Therefore, we chose A* for the global path planning for efficiency concern. During navigation, if obstacles that did not exist at the time of mapping appear on the map, they cause a collision problem between the AGV and obstacles, which is solved by the local path planning. The DWA and TEB are the most commonly used methods in such cases. The DWA simulates the trajectory of the AGV, according to the sensors and odometry information, and scores the paths at each speed. The TEB optimizes the execution time of the A* trajectory, minimises the deviations of speed, acceleration, obstacle avoidance, etc., and finally optimizes the best local path. According to Rösmann et al. [20], it is known that the DWA slows down before obstacles to avoid collisions; however, the TEB method, having previously predicted the collisions, chooses an alternative route earlier. Therefore, TEB is the better choice for this study.

3. Result and Discussion

The experiments were conducted in part of the basement corridor of a campus building, which is a semi-open environment. A SICK LiDAR was employed to construct the corresponding 2D map using Gmapping SLAM algorithm, as shown in Figure 4 where the garden area is open to the sky. We designed two routes, and , for the experiment. We assume that the start point of the AGV is point O in front of our laboratory, and a waypoint is point A for avoiding the automatic path planning finding the shortest path through the garden; further, points B and C are the first and second target points, respectively.

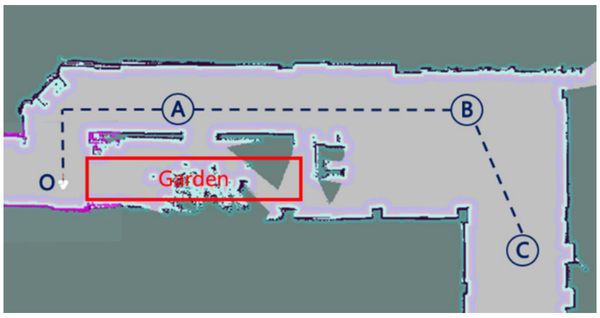


Fig. 4. Experimental field and route map

3.1. Path 1 Experiment

The first experimental route was , which had a length of 25 m. In this route, the encoder odometer, AMCL, and ORB-SLAM were separately used to test the positioning accuracy 15 times. One of the navigation results of the encoder odometer is shown in Figure 5, where the green line was the LiDAR scanning points and the red line was the local costmap in the move_base. The left figure showed clearly that the laser scanning point from the RVIZ did not fit well the actual wall boundaries. Several reasons were accountable for this positioning error: this experiment relied only on the encoder and inertial measurement unit for the sensing information, and the error in computing travel distance (Equation 9) by integrating AGV velocity (Equation 7) became more significant as the travel distance increased. Furthermore, it was susceptible to interference from external factors such as excessive wheel friction due to AGV weight, slipping or motor idling due to the floor material used, and floor flatness. In summary, for this experiment, the mean absolute error (MAE) of positioning error from the goal was 57 cm, and MAE of the angle deviation from the target direction was approximately 5.4°.

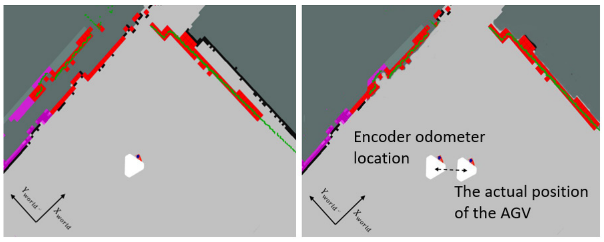


Fig. 5. Navigation results using the encoder odometer

In the AMCL experiment, sometimes the AGV could not obtain the current valid LiDAR sensor information because of the lack of apparent features on both sides of the corridor; therefore, it was possible to lose the location information. To address this, one can place mirrors on both sides of the corridor to increase the features to solve the problem. However, particle divergence did not occur very often in this experiment. For example, Figure 6 illustrated one of the successful experimental results: the AMCL particles converged in front of the AGV, and the positioning information during navigation was relatively accurate and stable. The right figure showed a matched laser scanning line with the wall in the X-axis of the word coordinate. The final MAE of positioning error was 34 cm, 40% less than the encoder odometer method, and the MAE of the angle deviation was 5.8°.

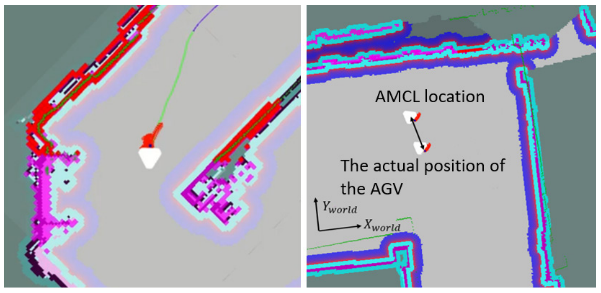


Fig. 6. Navigation results using AMCL localization

In ORB-SLAM localization, several key points were found for successful navigation: stable lighting condition; reduced angular speed about Z-axis, especially during the corner turning midway between Points O and A; and reduced weighting of the X-axis movement (transverse direction) in path planning. These key points ensured accurate feature points extraction and matching without too much frame rate loss due to unnecessary rapid lens rotation. One of the experimental results was depicted in Figure 7, where the green scanning line overlapped with the wall closely. The final MAE of positioning error was 13 cm, 77% and 62% less than the respective encoder odometer method and AMCL, and MAE of the angle deviation was 6°.

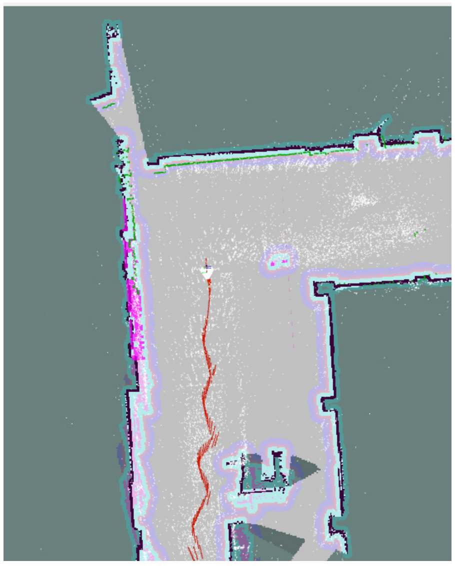


Fig. 7. Navigation results using ORB-SLAM localization

3.2. Path 2 Experiment

The experimental route is , which has a length of 40 m. Note that Point C is located 2 m in front of an elevator, resulting in Path 2 simulating a robot navigation from the laboratory to an elevator. Based on the experiment results in Path 1, we knew that the deviation from the goal using the encoder odometer increased with distance and was the worst compared to the other two localization algorithms. Therefore, we applied AMCL and ORB-SLAM to navigate the robot 30 times for each method on this path. One of the experimental results is shown in Figure 8, where both methods successfully dispatched the robot to an elevator 40 m away and gave closely matched laser scanning lines with the front wall.

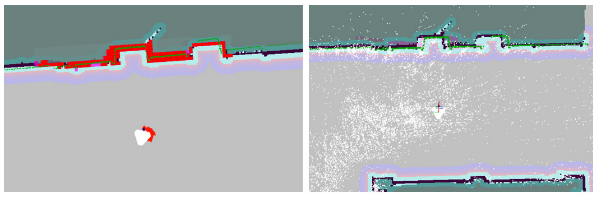


Fig. 8. Navigation results using AMCL (left) and ORB-SLAM (right) to point C

The test route was repeated 30 times for each localization method; the deviation error from the goal for each run in the X-Y plane is depicted in Figure 9, and the MAE of positioning error was summarized in Table 1. The MAE of positioning error for AMCL was 34.8 cm, and MAE of the angle deviation was 8.6° (recall it was 34 cm and 5.8° in Path 1). Comparing results of Path 1 with those of Path 2 showed that the AMCL robot navigating to the elevator still was mainly affected by the long corridor, and the rich number of feature points from point B to point C (15 m long) contributed not much to the MAE. Consequently, the AMCL was verified again to be unsuitable for the long corridor environment. For ORB-SLAM navigation to the elevator, it was relatively more accurate than AMCL, as clearly shown in Figure 9 and Table 1; the MAE of positioning error was 17.7 cm (34.8 cm for AMCL), and MAE of the angle deviation was 7.3° (8.6 for AMCL). Note also that the advantage of ORB-SLAM over AMCL in positioning error in Path 2 (49% less) was worse than that in Path 1 (62% less) due to another corner turning near Point B. Indeed, it was observed that recovery behaviour for robot re-localization occurred multiple times nearby Point B for corner turning for ORB-SLAM, leading to this reduction in the advantage of ORB-SLAM over AMCL.

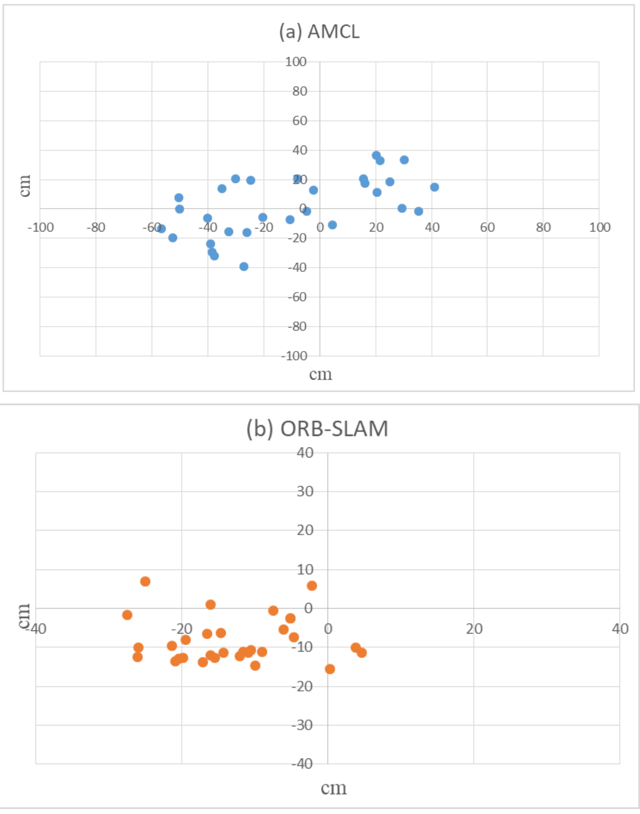


Fig. 9. Positioning errors with (a) AMCL and (b) ORB-SLAM

Table 1: MAE of AMCL and ORB-SLAM

	Δx	Δy	$\Delta\theta$	Δl
AMCL	28.2	16.7	8.6	34.8
ORB-SLAM	13.8	9.3	7.3	17.7

4. Conclusions

In this paper, a static map established with a 2D LiDAR sensor was integrated with a point cloud map constructed with an RGBD camera to increase robot localization accuracy. Moreover, the A* algorithm in global path planning and TEB method in local path planning for dynamic obstacle avoidance were employed in a three-wheeled omnidirectional AGV for autonomous navigation. Experimental results showed that the 3D ORB-SLAM localization outperformed the 2D AMCL for 49% MAE reduction of positioning error in a test route from the laboratory to an elevator 40 m away.

To highlight the effectiveness of the TEB method over DWA in local path planning and maneuvering of the omni-wheel drive over differential-wheel drive, the authors also applied DWA with ORB-SLAM to a differential wheel robot for the same test route ($\overline{OC}$) [21]. Unfortunately, the robot seldom succeeded in reaching the goal (Point C) right in front of the elevator, mainly due to localization divergence, especially near the two corner turnings. In summary, an omnidirectional mobile robot with TEB and ORB-SLAM can navigate autonomously and more precisely to an elevator 40 m distance away than other cases studied in this paper [21].

ACKNOWLEDGEMENTS

This study was supported by the National Taipei University of Technology – Nanjing University of Science and Technology Joint Research Program (NTUT-NUST-108-01)

AUTHORS

Pan-Long Wu – Nanjing University of Science and Technology, China, e-mail: plwu@njust.edu.cn.

Jyun-Jhen Li – National Taipei University of Technology, Taiwan, e-mail: dory110220@gmail.com.

Jinsiang Shaw* – National Taipei University of Technology, Taiwan, e-mail: jshaw@ntut.edu.tw.

*Corresponding author

REFERENCES

[1] R. Mur-Artal, J.M.M. Montiel and J.D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system", *IEEE Trans Robot*, vol. 31, no. 5, 2015, pp. 1147–1163.

[2] R. Mur-Artal and J.D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras", *IEEE Trans Robot*, vol. 33, no. 5, 2017, pp. 1255–1262.

[3] G. Grisetti, C. Stachniss and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters", *IEEE Trans Robot*, vol. 23, no. 1, 2007, pp. 34–46.

[4] A.J. Bostel and V.K. Sagar, "Dynamic control systems for AGVs", *Comput Control Eng J,* vol. 7, no. 4, 1996, pp. 169–176.

[5] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance", *IEEE Robot Autom Mag,* vol. 4, no. 1, 1997, pp. 23–33.

[6] C. Rösmann, W. Feiten, T. Wösch, et al., "Efficient trajectory optimization using a sparse model", *European Conference on Mobile Robots*, Barcelona, Spain, 25–27 September 2013, pp. 138-143.

[7] K.V. Ignatiev, M.M. Kopichev and A.V. Putov, "Autonomous omni-wheeled mobile robots", *2nd International Conference on Industrial Engineering, Applications and Manufacturing*, Chelyabinsk, Russia, 19–20 May 2016, pp. 1-4.

[8] S.A. Magalhães, A.P. Moreira and P. Costa, "Omnidirectional robot modeling and simulation", *IEEE International Conference on Autonomous Robot Systems and Competitions*, Ponta Delgada, Portugal, 15–16 April 2020, pp. 251-256.

[9] J. Xin, X.L. Jiao, Y. Yang, et al., "Visual navigation for mobile robot with Kinect camera in dynamic environment", *35th Chinese Control Conference*, Chengdu, China, 27–29 July 2016, pp. 4757-4764.

[10] Z. Meng, C. Wang, Z. Han, et al., "Research on SLAM navigation of wheeled mobile robot based on ROS", *5th International Conference on Automation, Control and Robotics Engineering*, Dalian, China, 19–20 September 2020, pp. 110-116.

[11] Y. Feng, C. Ding, X. Li, et al., "Integrating Mecanum wheeled omni-directional mobile robots in ROS", *IEEE International Conference on Robotics and Biomimetics*, Qingdao, China, 3–7 December 2016, pp. 643-648.

[12] B.A. Berg, *Markov chain Monte Carlo simulations and their statistical analysis*. Hackensack, New Jersey: World Scientific, 2004.

[13] R. Akkaya and F.A. Kazan, "A new method for angular speed measurement with absolute encoder", *Elektron Elektrotech*, vol. 26, no. 1, 2020, pp. 18–22.

[14] Figure source: Open-Base. https://github.com/GuiRitter/OpenBase

[15] J. Goncalves, J. Lima and P. Costa, "Real time tracking of an omnidirectional robot – An extended Kalman filter approach", *5th International Conference on Informatics in Control, Automation and Robotics*, Funchal, Portugal, 11–15 May 2008, pp. 5–10.

[16] Package source: move_base. http://wiki.ros.org/move_base

[17] J.G. Ziegler and N.B. Nichols, "Optimum settings for automatic controllers", *J Dyn Sys Meas Control,* vol. 115, no. 2B, 1993, pp. 220–222.

[18] E.B. Olson, "Real-time correlative scan matching", *IEEE International Conference on Robotics and Automation*, Kobe, Japan, 12–17 May 2009, pp. 4387-4393.

[19] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles", *IEEE International Conference on Robotics and Automation*, Rome, Italy, 10–14 April 2007, pp. 1986-1991.

[20] C. Rösmann, F. Hoffmann and T. Bertram, "Planning of multiple robot trajectories in distinctive topologies", *European Conference on Mobile Robots*, Lincoln, UK, 2–4 September 2015, pp. 1-6.

[21] P.L. Wu, Z.M. Zhang, C.J. Liew, et al., "Hybrid navigation of an autonomous mobile robot to depress an elevator button", *Journal of Automation, Mobile Robotics and Intelligent Systems*, accepted, 2022.