Hybrid Navigation of an Autonomous Mobile Robot to Depress an Elevator Button

Submitted: 4th July 2021; accepted: 29th September 2022

Pan-Long Wu, Zhe-Ming Zhang, Chuin Jiat Liew, Jin-Siang Shaw

DOI: 10.14313/JAMRIS/4-2022/30

Abstract:
The development of an autonomous mobile robot (AMR) with an eye-in-hand robot arm atop for depressing elevator button is proposed. The AMR can construct maps and perform localization using the ORB-SLAM algorithm (the Oriented FAST [Features from Accelerated Segment Test] and Rotated BRIEF [Binary Robust Independent Elementary Features] feature detector-Simultaneous Localization and Mapping). It is also capable of real-time obstacle avoidance using information from 2D-LiDAR sensors. The AMR, robot manipulator, cameras, and sensors are all integrated under a robot operating system (ROS). In experimental investigation to dispatch the AMR to depress an elevator button, AMR navigation initiating from the laboratory is divided into three parts. First, the AMR initiated navigation using ORB-SLAM for most of the journey to a waypoint nearby the elevator. The resulting mean absolute error (MAE) is 8.5 cm on the x-axis, 10.8 cm on the y-axis, 9.2-degree rotation angle about the z-axis, and the linear displacement from the reference point is 15.1 cm. Next, the ORB-SLAM is replaced by an odometry-based 2D-SLAM method for further navigating the AMR from waypoint to a point facing the elevator between 1.5 to 3 meter distance, where the ORB-SLAM is ineffective due to sparse feature points for localization and where the elevator can be clearly detected by an eye-in-hand machine vision onboard the AMR. Finally, the machine vision identifies the position in space of the elevator and again the odometry-based 2D-SLAM method is employed for navigating the AMR to the front of the elevator between 0.3 to 0.5 meter distance. Only at this stage can the small elevator button be detected and reached by the robot arm on the AMR. An average 60% successful rate of button depressing by the AMR starting at the laboratory is obtained in the experiments. Improvements for successful elevator button depressing rate are also pointed out.

Keywords: *ROS, AMR, ORB-SLAM, Robot Manipulator, Machine Vision.*

1. Introduction

With the advent of smart manufacturing in a wide range of industries, unmanned factories and automation have become the future trend. Therefore, automated machines such as AMRs and robot manipulators have been utilized to perform multiple tasks within the production line, such as the transportation of cargo and highly repeated workloads to replace labor resources, and even reduce cost. Commonly used AMRs can be categorized into two different types by guiding methods, namely, rail-guided and trackless automated guided. A rail-guided mobile platform utilizes special tracks that are tiled to the floor, which generate electromagnetic fields to guide movement. A trackless automated guided mobile platform is normally based on laser range-finder and camera as the sensor's data input to create a surrounding map and to determine a possible route within that map. The designated AMR in this report utilizes a trackless automated guided mobile platform to address the limitations associated with track leading, but has additional freedom of movement to perform any possible route.

To navigate an AMR, a map is required to appropriately define its localization. However, there is considerable causality between map construction and localization. For instance, Smith et al. [1] proposed that the presentation and calculation of uncertain spatial data requires an unbiased map, but such a map requires accurate location estimation to build. Durrant-Whyte and Bailey [2,3] proposed a simultaneous localization and mapping algorithm (SLAM), which has become a core technology in the field of mobile robots. SLAM is a solution for mobile robots to facilitate motion in an unknown environment. The SLAM method has been considered in two dimensions in which scanning data is acquired using a 2D laser range-finder, and in three dimensions wherein point cloud information is acquired via 3D laser range-finder or cameras. For 2D-SLAM, commonly used mapping algorithms include GMapping SLAM [4], Hector SLAM [5], and Cartographer SLAM [6]. The 3D SLAM method is also popular and is utilized in MonoSLAM [7], parallel tracking and mapping (PTAM) [8], and ORB-SLAM [9]. Each mentioned SLAM algorithm has its own specific properties. Whenever an AMR is successfully located in a prepared map, it should be navigated and guided along an assigned route when a valid destination is set as a goal on the map. Hart et al. [10] proposed a heuristic search algorithm, Bostel and Saigar [11] proposed the A* algorithm, Koeing and Likhachev [12] proposed the D*lite algorithm, and Fox et al. [13] proposed a dynamic window approach (DWA) to achieve these objectives.

In previously published works, it has been shown that a robot manipulator integrated with AMR has improved utility and working efficiency in the production line. For instance, Kousi et al. [14] simulated SLAM and navigation to a mobile dual-arm robot under ROS framework for a safe and collision-free path during the execution of the different assembly task.

Apart from the AMR and the robot manipulator, machine vision is also one of the key technologies used for automation, which can replace the vision of workers in highly repeated detection tasks by exploiting different image processing techniques. Machine vision is reliable because the methods and algorithms that are used yield consistent and accurate results. Therefore, a robot manipulator integrated with machine vision is a widely implemented technology. For instance, Sangeetha et al. [15] developed a low-cost eye-in-hand system that included a robot arm and a set of stereo cameras. The object position was detected by the stereo camera based on image processing. The inverse kinematics of the robot manipulator were solved to perform a pick-and-place task. Similar eye--in-hand robotic manipulator architecture has been proposed by Shaw and Chi [16] for image-based visual servoing (IBVS) for fetching moving objects in a production line. Hosseininia et al. [17] also employed machine vision to guide a robot arm for porcelain edge polishing. Moreover, a mobile platform can also be integrated with machine vision. Laganowska [18] investigated the detection of road lines as aids in navigation control.

As previously indicated, this investigation will be based on a trackless automated guided vehicle as the mobile platform to obviate the need for presetups such as specific wiring configurations and limitations associated with a particular movement. In our case, both the laser range finder and camera are primarily used to obtain environmental information, wherein the laser scanning data is accurate and the camera can provide 3D spatial information. The experimental field in this study is the corridor in the building basement with approximate size 45m × 20m. The goal is to navigate the AMR from the laboratory to an elevator and depress the button (as if it is going to take the elevator). However for a corridor with this spacious size, the 2D mapping methods such as GMapping, Hector, and Cartographer SLAM, might cause several types of errors in the selected regions. In this work, the ORB-SLAM algorithm, a 3D SLAM method based on an RGB-D camera that acquires feature points from each frame to facilitate positioning is utilized. Nevertheless, calibration of the camera should be performed as a precaution since the cameras might exhibit radial and tangential distortion in the image. Therefore, calibration can modify the camera parameters such as the focal length, center of the image, and the distortion coefficients [19]. Given that the experimental field is vast and the displacement error associated with SLAM might lead to navigation failure, a machine vision system is hence built to lead the AMR to a precise position at the final stage for depressing the elevator button. In this respect, Zhou and Liu [20] installed a camera on a mobile robot to scan a 2D barcode and successfully localized the position based on the acquired image. The vision-based movement control allowed the mobile robot to precisely move to an assigned position. Schueftan et al. [21] used the KUKA robot arm mobile platform to perform autonomous navigation after the creation of a map using LiDAR. Several Vicom cameras were installed around the field to observe and determine the moving deviation, and the positioning accuracy was hence secured.

2. System Design

2.1 AMR Construction

The AMR presented in this study was designed and built using 30 cm × 30 cm aluminum frames to reduce cost and improve strength. The final size of the constructed AMR was 60 cm × 60 cm × 90 cm (in length × width × height). A laptop computer was used on the AMR to operate the ROS system. The AMR was designed based on a differential drive control, in which two sets of 12V DC motors were utilized and controlled using pulse width modulation (PWM) signals. Encoders attached to the DC motors were used for motor PID speed control and for measuring the movement traveled by the AMR as well. Several Arduino microcontrollers were connected to the computer via USB ports and used primarily for motor control and for I/O signals. A SICK LiDAR with better accuracy and longer range was attached to the front end of the AMR for scanning the environment and RBPF (Rao-Blackwellized particle filter) in GMapping SLAM [4] was applied for constructing a 2D map for navigation. Two sets of Hokuyo LiDAR were also installed at the front left and rear right of the cart to facilitate 360° object detection. They were connected to the computer via USB ports to receive laser data for obstacle avoidance during navigation. In order to compensate the positioning errors with SLAM, particularly in the orientation angle, an inertial measurement unit (IMU) sensor was also employed. Kinect v2, an RGB-D camera, was used to obtain 3D point cloud information [22]. A Niryo robot served as the 6-degrees of freedom robot manipulator and was mounted on the top layer of the AMR. It was connected to the computer via Ethernet and programmed based on Python code. An IDS-XS industrial camera was mounted at the end effector of a Niryo robot arm, forming an eye-in-hand system to acquire an RGB image. Image processing was performed using OpenCV libraries for detecting the elevator. An NVIDIA Jetson TX2 module was also connected to the laptop computer via Ethernet to share the calculation burden during image processing. Figure 1 shows the constructed AMR and the associated main components used in the study.

2.2 ORB-SLAM Method

ORB-SLAM is a 3D SLAM method, which is classified as an indirect and sparse mapping procedure based on oriented FAST and rotated BRIEF algorithms [9]. This approach was developed based on PTAM architecture [8] by applying the ideas of place recognition,

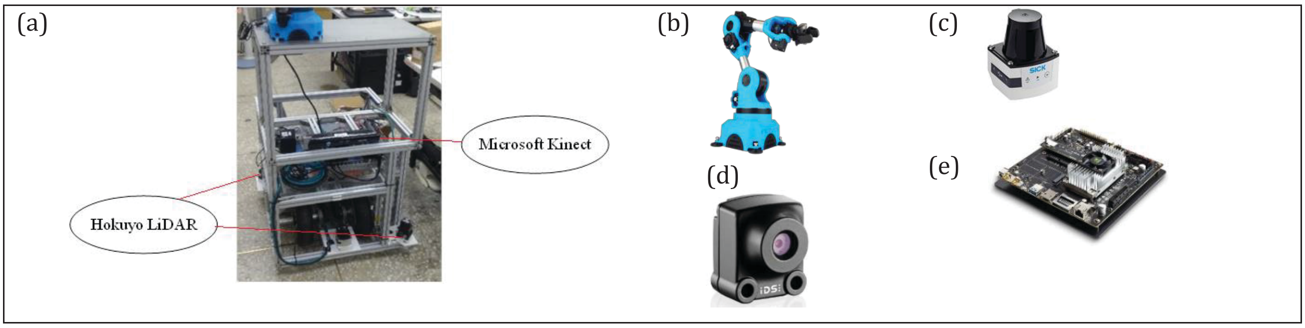


Fig. 1. Equipment used in the study: (a) AMR; (b) Niryo robot arm; (c) SICK LiDAR; (d) IDS-XS camera; (e) NVIDIA TX2

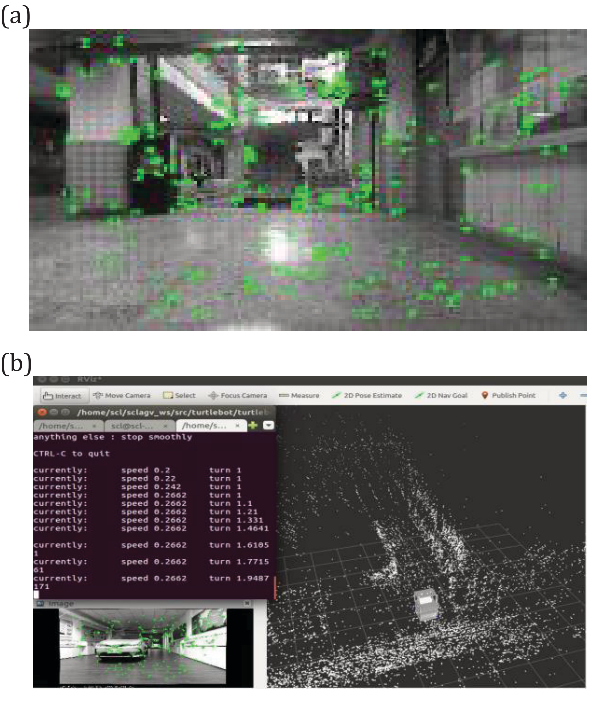
scale-aware loop closing, and covisibility graph, and redesigned into a new methodology. The main body includes map initialization and closed-loop detection functions, optimization of key frame selection, and map construction methods, which results in excellent processing speed, tracking effect, and map accuracy. The features selected by the algorithm are mainly based on the FAST feature extraction method, but with rotation invariance. It is then converted into a binary form using the BRIEF algorithm, which is very efficient for both building and matching. ORB-SLAM was originally developed for monocular cameras but later expanded for application to stereo and RGB-D cameras. In this study, the Kinect V2 was utilized as an RGB-D camera to generate color images and provide depth information to identify valid point clouds for feature matching. Figure 2 shows the image processing result obtained for the ORB-SLAM algorithm and the subsequent constructed mapping and localization.

2.3 Mapping and Navigation Methods

In this investigation, both ORB-SLAM and GMapping SLAM are used because of their different advantages in constructing maps. ORB-SLAM involves the acquisition of featured key points based on each frame image to generate a 3D map, which is mainly used for localizing the AMR position, and is more appropriate in such a long corridor environment. However, a 2D map is still needed. The GMapping algorithm generates a map from the SICK LiDAR scanning data, which is imported as a static map for the path planner to produce a navigating route, and for real-time obstacle avoidance as well.

GMapping was developed based on the RBPF to obtain the correct map. RBPF uses the known initial pose of the mobile platform x_0, and the map environment data collected by the sensor m_0 to represent the map environment information using the Markov process as $z_{1:t} = z_1, z_2, ..., z_t$, and the mobile platform motion information as $u_{1:t} = u_1, u_2, ..., u_t$. The motion trajectory is estimated as $x_{1:t} = x_1, x_2, ..., x_t$, and the posterior probability as p. According to Bayes' theorem, the posterior probability of SLAM at time t can be expressed as a recursive function of the posterior probability, environmental state, and motion state at time t-1, as shown in (1) [23]:

$$\mathrm{p}(x_{1:t},m|z_{1:t},u_{1:t-1})=\mathrm{p}(x_{1:t}|z_{1:t},u_{1:t-1})\mathrm{p}(m|x_{1:t},z_{1:t}) \quad (1)$$



```
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2       turn 1
currently:      speed 0.22      turn 1
currently:      speed 0.242     turn 1
currently:      speed 0.2662    turn 1
currently:      speed 0.2662    turn 1.1
currently:      speed 0.2662    turn 1.21
currently:      speed 0.2662    turn 1.331
currently:      speed 0.2662    turn 1.4641

currently:      speed 0.2662    turn 1.6105
1
currently:      speed 0.2662    turn 1.7715
61
currently:      speed 0.2662    turn 1.9487
171
```

Fig. 2. ORB-SLAM algorithm: (a) result of image processing; (b) constructed mapping and localization

where $x_{1:t}$ is the mobile platform state, $z_{1:t}$ is the observed state, and $u_{1:t}$ is the mobile platform input. Therefore, from (1), the mobile platform pose x can be obtained using the observed state z and the input signal u. Subsequently, the map m can be approximated based on the mobile platform position x and the observed state z. The map is then imported as static map information for the navigation stacks to deliver a global path. The global path planner in the navigation stacks is mainly based on a static map to plan an effective path according to a set target point. Commonly used algorithms such as the A* [11] can search for the shortest path. The A* algorithm uses an evaluation function, as shown in (2):

$$F(n)=G(n)+H(n) \quad (2)$$

where $F(n)$ is the evaluation score for reaching the endpoint, $G(n)$ represents the actual distance from the starting point to the current node, and $H(n)$ is the estimated distance from the current node to the endpoint. The search direction of all nodes points to the target point, which can remove redundant and unnecessary paths. Therefore, it takes less time and the path is more accurate.

The route generated based on the static map can only be applied in the same environment. However, there could be obstacles that exist after the map is established, and thus are not included in the consideration of global path planning. The dynamic window approach (DWA) [13] is hence proposed in local path planning in the obstacle layer for real-time dynamic obstacle avoidance. Finally, the inflation map layer is constructed to prevent the mobile platform from being too close to obstacles.

2.4 Image Processing Method

Using an eye-in-hand architecture, an IDS industrial camera is installed at the end effector of a Niryo robot arm, and machine vision is utilized for leading both the mobile platform and the robot manipulator close to the elevator and elevator button, respectively. First, a 7 × 5 image calibration chessboard is used for camera calibration. Next, an image of the elevator door is acquired and an image processing series is carried out to the image, including color space conversion from RGB to HSV, filtering using median filter and morphology, thresholding, image AND operation, contour detection, area computation, and 3D reconstruction [24]. The purpose of the image processing is to identify the position in space of the elevator and its button so that the mobile platform and the robot manipulator can be guided forward in order to depress the button.

2.5 System Architecture

This study is based on the ROS framework to integrate both hardware and firmware of the mobile robot. Figure 3 shows the system architecture with a laptop computer running the core system. A GPU module NVIDIA-TX2 is used to receive the image from the IDS camera and to perform image processing for detecting the elevator. In addition, an Arduino Mega2560 microcomputer is employed to control the speeds of the two driving motors of the mobile platform based on the encoder signals using the PID controller, upon receiving the velocity commands from the main computer. In navigation the main computer drives the mobile platform based on signals from the Kinect v2 for 3D localization for most of the navigation journey and later from motor encoders and IMU for 2D localization when approaching the elevator, and from the Hokuyo LiDAR sensors for dynamic obstacle avoidance. When the mobile platform arrives to a point facing the elevator between 0.3 to 0.5 meter distance where elevator button is reachable by the robot arm, the last stage SLAM navigation terminates and the IDS camera will guide the robot arm trying to depress the button.

3. Experimental Results

3.1 AMR SLAM Navigation to Elevator

The goal in this study is to navigate the AMR from the laboratory in the building basement to an elevator and to depress the button. The basement corridor that the AMR can navigate is about 45m × 20m. First, a 2D map depicting the floor layout is generated by using GMapping SLAM with the SICK LiDAR sensor, as shown in Figure 4. In this constructed map, point ‘S’ in front of the laboratory is the starting position for AMR navigation. West elevator (26.3 m apart) facing point ‘A’ or east elevator (40 m apart) facing point ‘B’ is the elevator to which the AMR can be heading for. The 2D map along with the well-known and commonly-used 2D localization method AMCL (Adaptive Monte Carlo Localization) [25] was tested for navigation, and the AMR failed to reach either elevator mainly due to the long corridor problem where at some locations no distinguishable features were available on both sides of AMR. Consequently, we resort to the 3D localization method of ORB-SLAM. However, due to the limitation on Kinec V2 such as detection range and field of view,

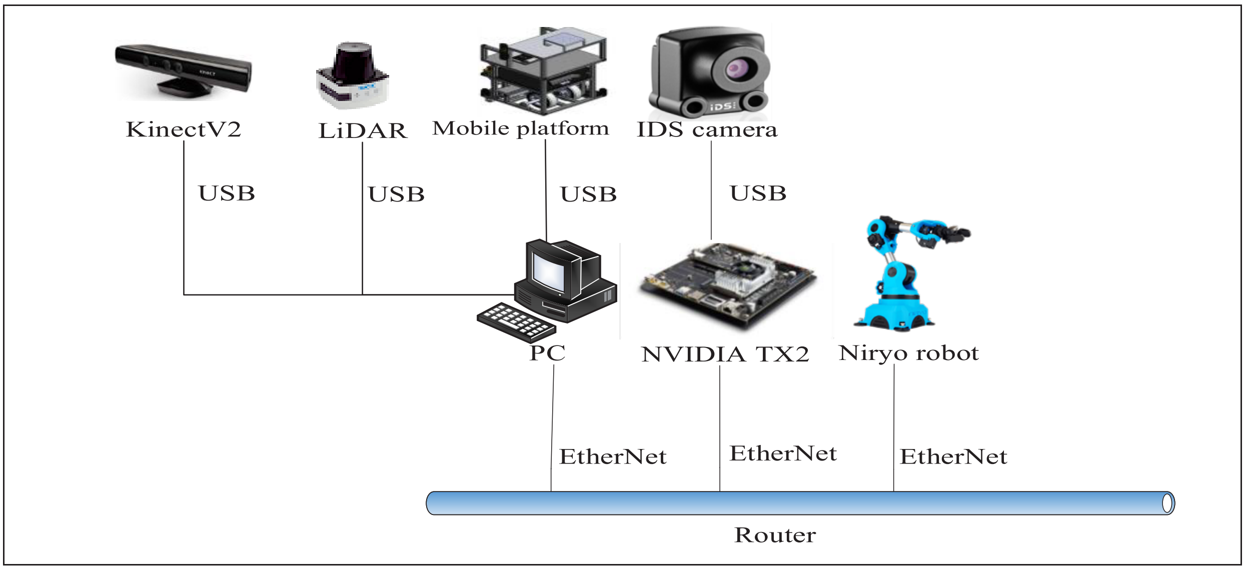


Fig. 3. The system architecture

AMR cannot be guided too close to the elevator where the ORB-SLAM is ineffective due to sparse feature points for localization. Therefore, point 'a' and point 'b' are the two waypoints chosen for AMR to navigate to the west and east elevator, respectively. At such waypoints 'a' and 'b', ORB localization method terminates. For AMR navigation and obstacle avoidance from point 'S' to waypoint 'a' (or 'b'), move_base package in ROS navigation stack is employed. Figure 5 shows the navigation architecture, where ORB localization is adopted and A* and DWA algorithms are used for global and local path planner, respectively. The A* algorithm plans the route from the current location to reach the goal, whereas the DWA algorithm is capable of updating a real-time route to bypass obstacles. Note that the 2D map in Figure 4 is used here for both path planners, data from Hokuyo LiDAR sensor is used in DWA algorithm for dynamic obstacle avoidance, and 3D point clouds from Kinect v2 are employed for ORB localization.

Navigation experiments have been carried out from point 'S' to waypoints 'a' and 'b', each path for 30 runs. The resulting positioning errors in x-y plane for this AMR navigation to waypoints 'a' and 'b' are shown in Figure 6. The mean absolute error (MAE) is 8.5 cm on the x-axis, 10.8 cm on the y-axis, 9.2-degree rotation angle about the z-axis, and the linear displacement from waypoints 'a' and 'b' is 15.1 cm. It is observed that, due to longer distance and non-uniform lighting conditions including area open to the sky, navigation to point 'b' has larger positioning error than point 'a'.

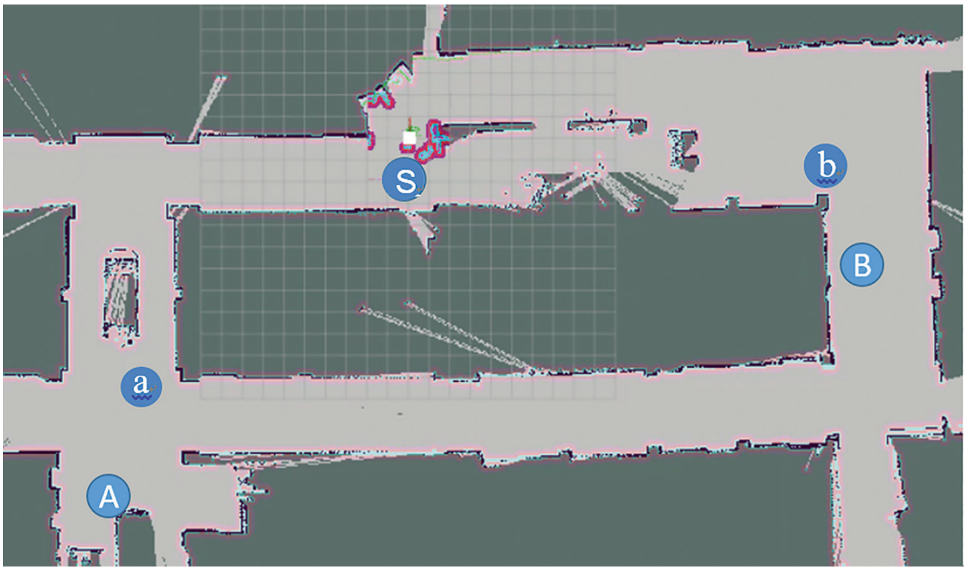


Fig. 4. The constructed 2D map of experiment field

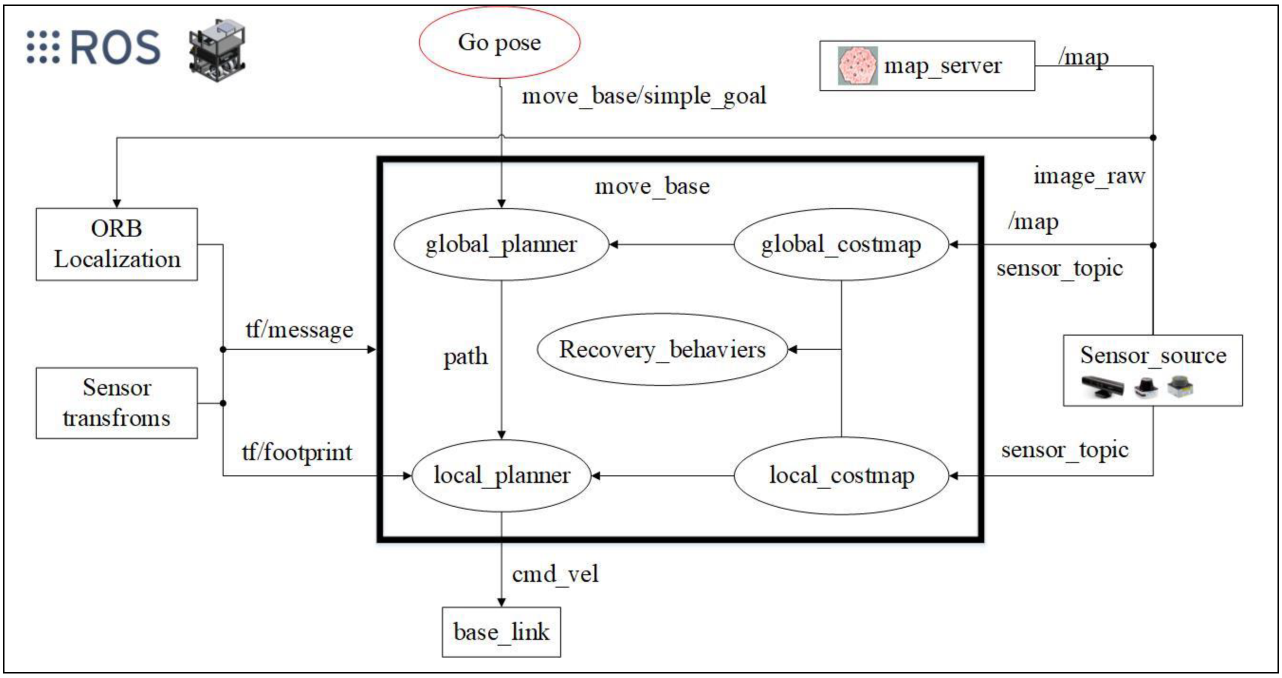


Fig. 5. The AMR navigation architecture

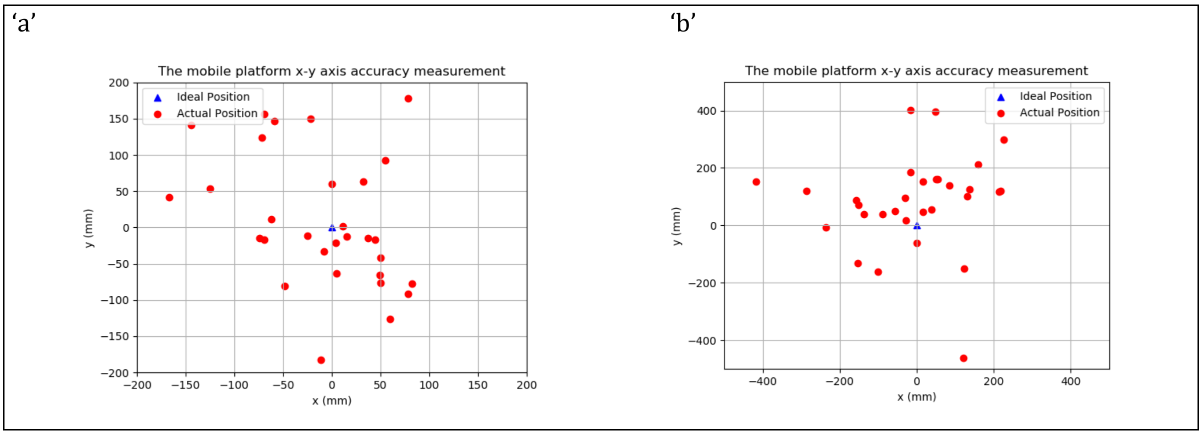


Fig. 6. Positioning errors at waypoint 'a' and 'b' of AMR navigation

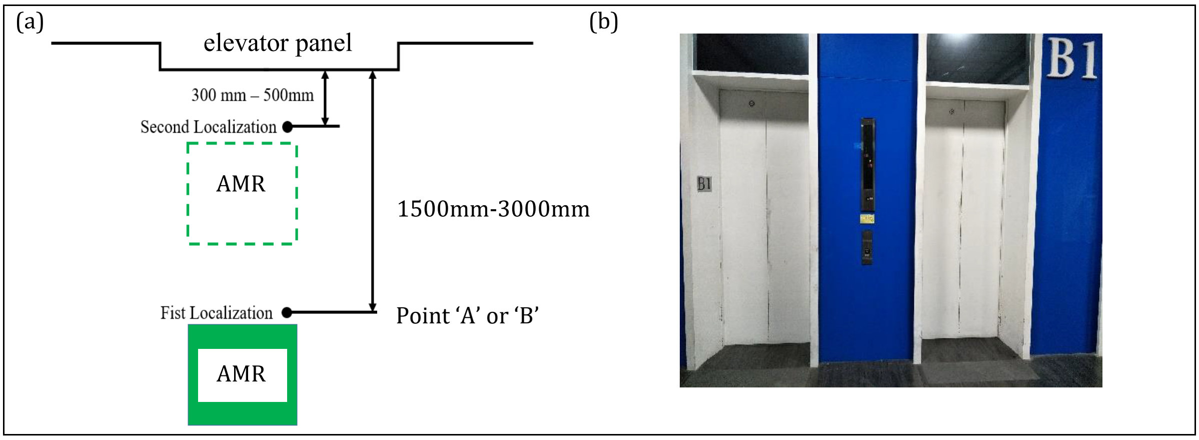


Fig. 7. AMR in front of elevator: (a) AMR location illustration; (b) elevator seen at point 'A' or 'B'

For the next continuous journey from waypoint 'a' to point 'A' (and from waypoint 'b' to point 'B'), the 3D ORB localization in Figure 5 is automatically replaced by an odometry-based 2D localization method using signals from the two motor encoders and an IMU. Points 'A' and 'B' are selected facing the two elevators between 1.5 to 3 meter distance, at which the AMR can see and detect the elevator without difficulty, as shown in Figure 7. Experimental results of navigation from waypoint 'a' to point 'A' (and 'b' to 'B') show MAE of 5.25-degree rotation angle about the z-axis and the linear displacement 50.0 cm from the target point 'A' or 'B'. Hence it is easy to see this 2D SLAM has a bigger positioning error of 50.0 cm than the first 3D ORB-SLAM navigation of 15.1 cm. This is mainly due to accumulation errors from the waypoints and from integrating motor speed to get displacement information as well during this trip. Nonetheless, for all 60 runs from the start point 'S' to points 'A' and 'B' with such positioning errors, the AMR indeed is able to detect the elevator without difficulty so that the third AMR journey to get closer to the elevator for depressing the button can be made possible, as will be seen in the next section.

3.2 Machine Vision Guided SLAM Navigation

For the third (also the last) stage of the AMR journey, AMR has to determine its own navigation target, unlike the first two journeys at which the target points are preset on the 2D map by operators. As AMR arrives at the first location (namely point 'A' or 'B') in Figure 7(a), the IDS camera will be used to detect and determine the true elevator position relative to the AMR. When it is done, a target point (as shown, the second location in Figure 7(a) between 0.3 to 0.5 meter distance from the elevator) can be issued to the AMR for the last navigation. It is noted that at the second location the robot arm atop the AMR with R500 mm working space can reach and depress the elevator button. Consequently, two steps are involved in this stage: IDS camera detects the elevator and the AMR moves to the second location. The success rate for each step will be summarized in Table 1.

As shown in Figure 7(b), the elevator up button (Δ) is not detected in first priority because it is too small and thus unrecognizable in the acquired image when the AMR is at the first location. There is, however, a yellow sticker below the up button, which can be detected as a valid target. A series of image processing algorithms are applied to the acquired image with yellow sticker inside, including RGB to HSV model transformation (as an example see Figure 8(a)), yellow color thresholding (Figure 8(b)), noise removal by median filter, and by morphology transformation such as closing (Figure 8(c)), and finally contour finding and the Perspective-n-Point (PnP) transform [26]

Tab. 1. The success rate for each task

From position	Yellow sticker detection	Moving to second location	Button detection	Button depressed	Combined success rate
'A'	100 % (30/30)	90.0 % (27/30)	59.3 % (16/27)	87.5 % (14/16)	46.7 % (14/30)
'B'	100 % (30/30)	86.7 % (26/30)	80.8 % (21/26)	85.7 % (18/21)	60.0 % (18/30)

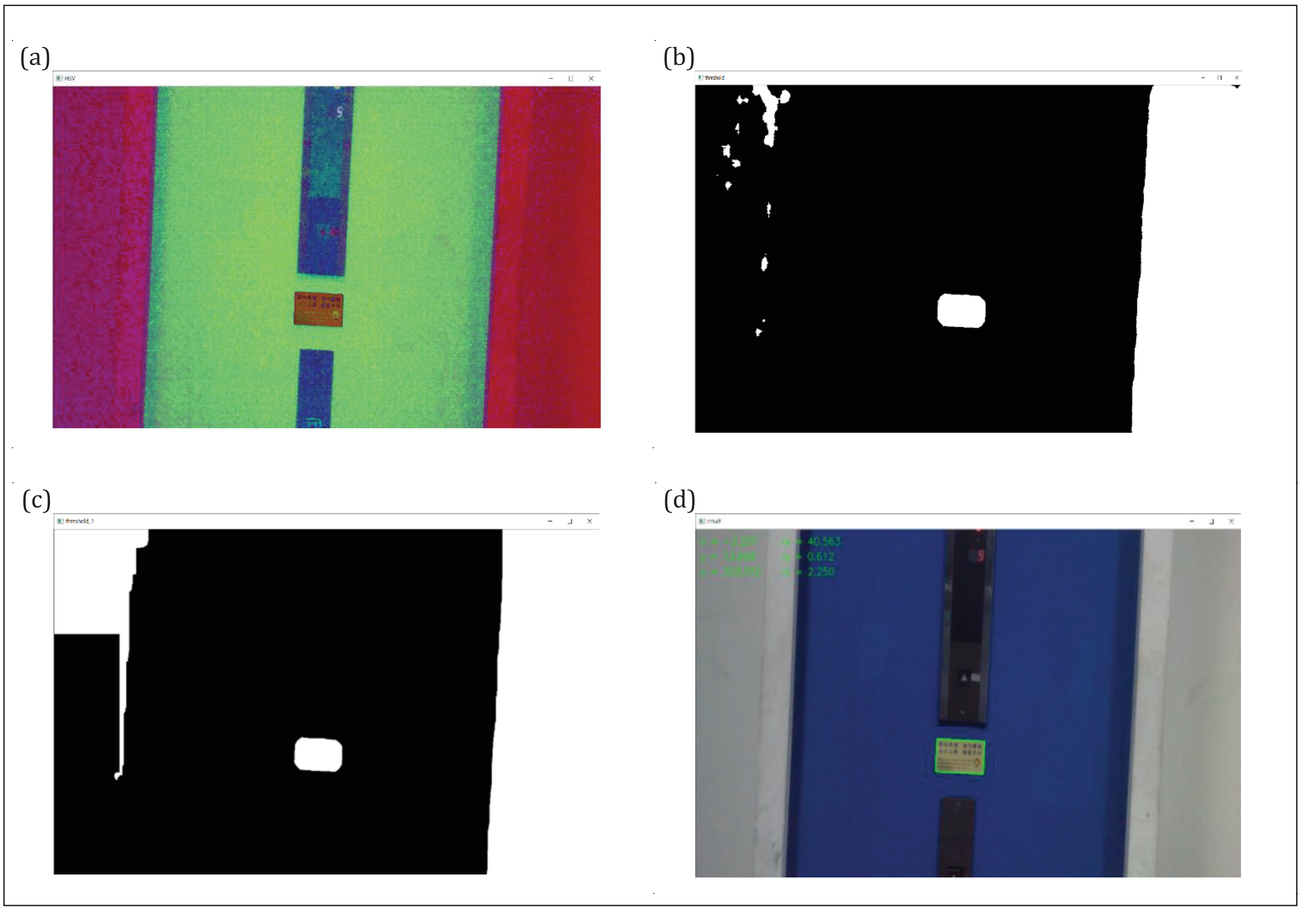


Fig. 8. Image processing to the yellow sticker detection: (a) HSV image; (b) color thresholding; (c) noise filtering; (d) position and rotation of the yellow sticker identified

(Figure 8(d)). The space coordinate of the yellow sticker (and hence the elevator) including position (x, y, z) and rotation (rx, ry, rz) relative to the camera frame is determined, as clearly seen in Figure 8(d). For all 30 experiment runs from point 'S' to point 'A' and another 30 runs to point 'B', the IDS camera succeeds in detecting the yellow sticker location every time at these locations, with approximate 30 mm displacement error in distance.

Once the elevator position and rotation relative to the camera frame is determined, i.e., $T_{vision}^{sticker}$, the target point for AMR to navigate to can be obtained from

$$T_{AMR}^{sticker} = T_{AMR}^{0} T_{0}^{6} T_{6}^{vision} T_{vision}^{sticker} \tag{3}$$

where T_{AMR}^{0}, T_{0}^{6}, T_{6}^{vision} represents respectively the homogeneous transformation matrix from robot arm base frame to AMR frame, from robot manipulator flange frame to robot arm base frame, and from camera frame to robot manipulator flange frame. Then the AMR at the first location can be navigated again toward the elevator to the target point (namely, the second location shown in Figure 7(a)) using the odometry based 2D SLAM technique. Experimental results show navigation success rates of 90.0% (27/30) and 86.7% (26/30) from point 'A' and point 'B' respectively to the target position. That is, three navigations from point 'A' and four journeys from point 'B' fail to reach the second location (in fact the AMR collides with the elevator). The failure is due to both the navigation error and the limited working space of the robot arm (R500 mm), which might overlap with the inflation map layer with radius 400 mm.

3.3 Elevator Button Detection and Depressing

As the AMR has successfully moved forward to the second location with 0.3 to 0.5 meter distance from the elevator, the IDS camera may see the button clearly (for example Figure 9(a)) and hence can guide the robot arm to depress it. Similar to the previous image

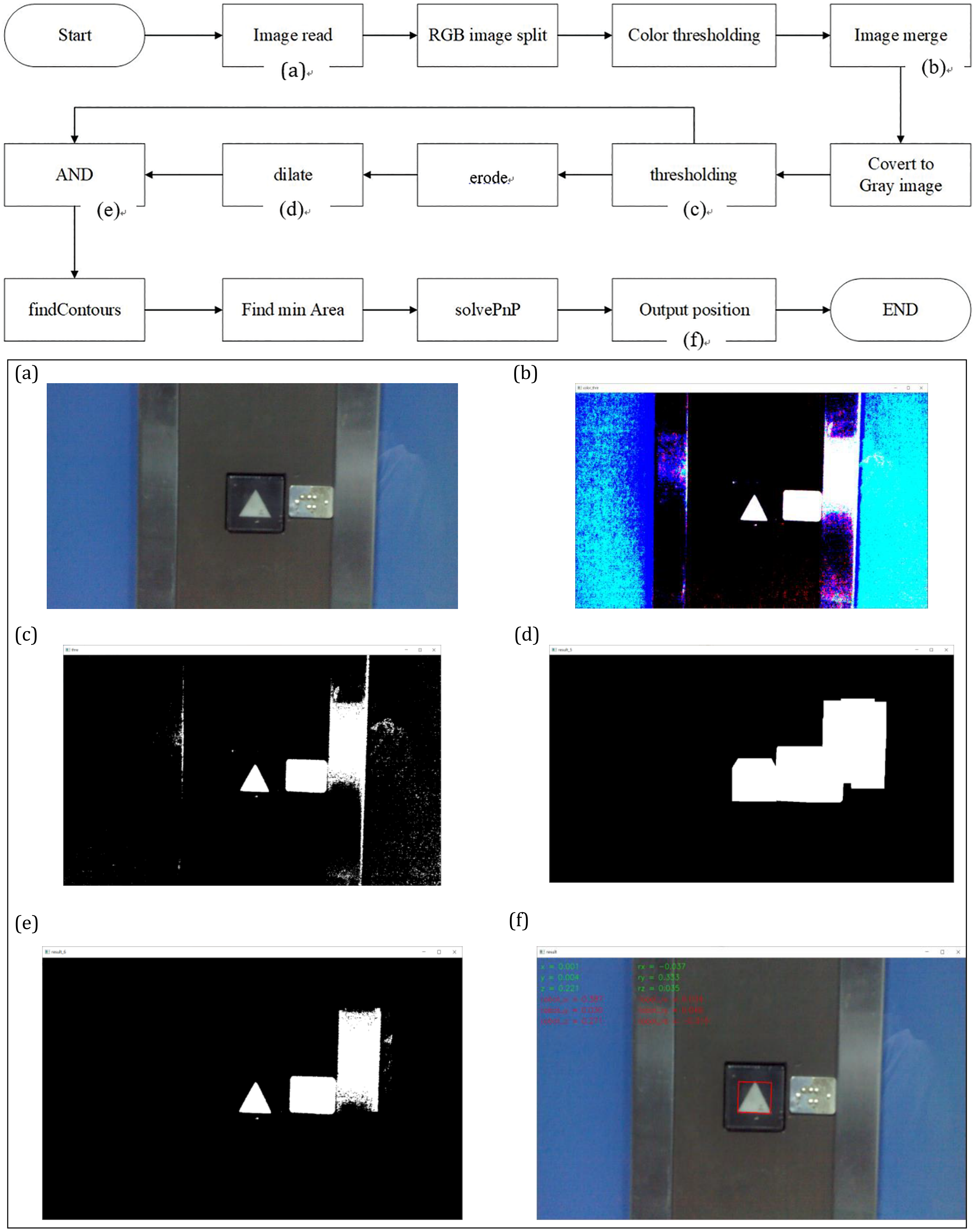


Fig. 9. Image processing to the button detection: (top) flow chart; (bottom) sample results

processing for detecting yellow sticker, Figure 9 depicts a series of image operations applied to the source image of elevator button and shows some resulting samples along the process. It is found that as long as the button is inside the captured image, its space coordinates can be correctly determined, as shown in Figure 9(f). However, because AMR is driven by the differential wheels system, it is difficult to correct displacement error in y-axis, especially during such a short travel distance from point 'A' or point "B' to the second location (about 2 m from point 'A' and 3 m from point 'B'). Therefore, 59.3% (16/27) and 80.8 (21/26) success rates are recorded with the elevator button lying within the camera FOV for path 'A' and path 'B' respectively when AMR arrives at the second location. It is readily seen that with shorter travel distance for path 'A', it is not easy to have a large shift in y-axis displacement while requiring AMR to face

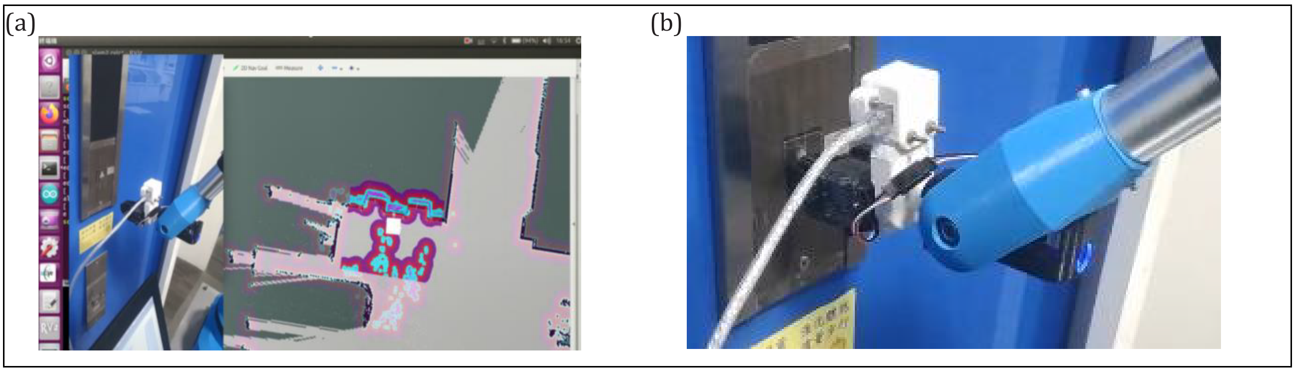


Fig. 10. The elevator button depressing: (a) AMR navigation to the second location and button detection; (b) depressing button successfully

directly the elevator in the end. More often the up buttons are located outside of the camera FOV for path 'A' than path 'B'. In Figure 4, it is seen that an automobile is parked at lower left of point 'A' while point 'B' has nothing around it. This explains why point 'A' is set with a shorter distance to the elevator, for safety concerns.

Finally, the robot arm is ready to depress the button, whose coordinates with respect to the robot base frame can be calculated as

$$T_0^{button} = T_0^6 T_6^{vision} T_{vision}^{button} \tag{4}$$

The reader is referred to Figure 9(f) for detected button position (robot_*x*, robot_*y*, robot_*z*) and rotation (robot_*rx*, robot_*ry*, robot_*rz*) with respect to robot base frame, as an example. The robot gripper is then controlled to depress the target button, as shown in Figure 10. In this final task of button depressing by robot arm, success rate of 87.5% (14/16) and 85.7% (18/21) are achieved for path 'A' and 'B' respectively. It is found the rare depressing failures (five out of 37) are all caused by the limited working space of the robot arm (R500 mm) that could not reach the button due to the AMR positioning error.

As a summary for the performance of the developed mobile AMR to depress the button at west elevator (26.3 m away) or east elevator (40 m away), Table 1 lists the success rate for each task and the overall combined success rate from beginning to end. For path A to west elevator, 14 of 30 navigations beginning at laboratory can finish the button depressing resulting in 46.3% success rate, while 18 of 30 navigations can finish the job with 60% success rate for path B to east elevator. Note that 60 trials for AMR to arrive at point 'A' and 'B' to detect the elevator position were all successful. However, for the ensuing tasks, mistakes happened, such as AMR bumping into the elevator, the small up button was not within the camera FOV, and robot arm was not long enough to touch the button, etc. To improve the success rate and enhance the system's robustness, several possible approaches are proposed: positioning accuracy from the first location to the second location could be increased by multiple SLAM navigations instead of just one journey; the differential drive system of AMR could be replaced by omni-directional drive, such as using Mecanum wheels for easy maneuvering in short travel distance while requiring AMR orientation in the end; scan the up button horizontally by robot arm when it is out of camera FOV; and lastly, use of longer robot arm for larger working space to reach the button, which would also be helpful in avoiding AMR bumping into the elevator.

4. Conclusion

A hardware system for a mobile AMR was developed, including an aluminum mobile platform, a robot manipulator with an eye-in-hand IDS industrial camera, an NVIDIA-TX2 GPU module, some embedded Arduino microcontroller units, and several sensors such as Kinect v2 RGB-D camera, LiDARs, encoders and an IMU. The software framework was based on the ROS architecture installed on a laptop running Ubuntu 16.1.04. The required functions were implemented using both C++ and Python programming. The aim of the study was to navigate the AMR towards an elevator and to summon one by depressing the button. AMR moving to front of the elevator for button depressing was made possible by three consecutive SLAM navigations. A 3D map of the experiment field for localization purposes by Kinec V2 and a 2D map for both localization and path planning by SICK LiDAR were constructed and corresponding 3D ORB localization and 2D odometry-based localization methods were employed in the navigation stacks. In addition, real time and dynamic obstacle avoidance via DWA was also implemented using two Hokuyo LiDARs.

With the developed hardware and software systems integrated in an AMR, an average 60% successful rate of button depressing by the AMR starting at the laboratory was obtained in the experiments. Improvements of successful elevator button depressing rate are also pointed out for future work.

Funding

This study was supported by the National Taipei University of Technology - Nanjing University of Science and Technology Joint Research Program

(NTUT-NUST-108-01) and by the Ministry of Science and Technology, Taiwan (MOST 111-2218-E-027-001).

AUTHORS

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

Zhe-Ming Zhang – National Taipei University of Technology, Taiwan, e-mail: qaz9517532846@gmail.com .

Chuin Jiat Liew – National Taipei University of Technology, Taiwan, e-mail: jianjiat@gmail.com

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

*Corresponding author

References

[1] R. Smith, M. Self and P. Cheeseman, "Estimating uncertain spatial relationships in robotics". In: IEEE International Conference on Robot and Automation, Raleigh, NC, USA, 31 March-3 April 1987, DOI: 10.1109/ROBOT.1987.1087846.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I", IEEE Robot & Automation Magazine, vol. 13, no. 2, 2006, 99-110, DOI: 10.1109/MRA.2006.1638022.

[3] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II", IEEE Robot & Automation Magazine, vol. 13, no. 3, 2006, 108-117, DOI: 10.1109/MRA.2006.1678144

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

[5] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf and O. von Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots", Robot Soccer World Cup, 2013, 624-631, DOI: 10.1007/978-3-662-44468-9_58.

[6] W. Hess, D. Kohler, H. Rapp and D. Andor, "Real-time loop closure in 2D LIDAR SLAM". In: IEEE International Conference on Robot and Automation. Stockholm, Sweden, 16-20 May 2016, 1271-1278. DOI: 10.1109/ICRA.2016.7487258.

[7] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: real-time single camera SLAM", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, 2007, 1052-1067, DOI: 10.1109/TPAMI.2007.1049.

[8] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces", In: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13-16 November 2007, 225-234, DOI: 10.1109/ISMAR.2007.4538852.

[9] R. Mur-Artal, J. M. M. Montiel, J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system", IEEE Transactions on Robot, vol. 31, no. 5, 2015, 1147-1163, DOI: 10.1109/TRO.2015.2463671.

[10] P. E. Hart, N. J. Nilsson and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths", IEEE transactions on Systems Science and Cybernetics, vol. 4, no. 2, 1968, 100-107, DOI: 10.1109/TSSC.1968.300136.

[11] A. J. Bostel and V. K. Saigar, "Dynamic control systems for AGVs", IEEE Trans. Computing & Control Engineering, vol. 7, no. 4, 1996, 169-176, DOI: 10.1049/cce:19960403.

[12] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain", IEEE Transactions on Robot, vol. 21, no. 3, 2005, 354-363, DOI: 10.1109/TRO.2004.838026.

[13] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance", IEEE Robot & Automation Magazine, vol. 4, no. 1, 1997, 23-33, DOI: 10.1109/100.580977.

[14] Kousi N, Gkournelos C, Aivaliotis S, et al. Digital twin for adaptation of robots' behavior in flexible robotic assembly lines, Procedia Manufacturing, 2019, 28: 121-126.

[15] G. R. Sangeetha, N. Kumar, P. R. Hari and S. Sasikumar, "Implementation of a stereo vision based system for visual feedback control of robotic arm for space manipulations", Procedia Computer Science, vol. 133, 2018, 1066-1073, DOI: 10.1016/j.procs.2018.07.031.

[16] J. Shaw and W. L. Chi, "Automatic classification of moving objects on an unknown speed production line with an eye-in-hand robot manipulator. Journal of Marine Science and Technology, vol. 26, no. 3, 2018, 387-396, DOI: 10.6119/JMST.2018.06_(3).0010.

[17] S. J. Hosseininia, K. Khalili and S. M. Emam, "Flexible automation in porcelain edge polishing using machine vision", Procedia Technology,

vol. 22, 2016, 562-569, DOI: 10.1016/j.protcy.2016.01.117.

[18] M. Laganowska, "Application of vision systems to the navigation of the mobile robots using makers", Transportation Research Procedia, vol. 40, 2019, 1449-1452, DOI: 10.1016/j.trpro.2019.07.200.

[19] Y. M. Wang Y. Li and J. B. Zheng, "A camera calibration technique based on OpenCV". In: The 3rd International Conference on Information Sciences and Interaction Sciences, Chengdu, China, 23-25 June 2010, DOI: 10.1109/ICICIS.2010.5534797.

[20] C. Zhou and X. Liu, "The study of applying the AGV navigation system based on two dimensional bar code". In: International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), Wuhan, China, 3-4 Dec. 2016, 206-209, DOI: 10.1109/ICIICII.2016.0057.

[21] D. S. Schueftan, M. J. Colorado and I. F. M. Bernal, "Indoor mapping using SLAM for applications in flexible manufacturing systems". In: IEEE 2nd Colombian Conference on Automatic Control (CCAC), Manizales, Colombia, 14-16 Oct. 2015, DOI: 10.1109/CCAC.2015.7345226.

[22] A. S. Sabale, "Accuracy measurement of depth using Kinect sensor". In: Conference on Advances in Signal Processing (CASP), Pune, India, 9-11 June 2016, DOI: 10.1109/CASP.2016.7746156.

[23] J. P. M. dos Santos, SmokeNav - simultaneous localization and mapping in reduced visibility scenarios. MSc thesis, Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal, September 2013. http://hdl.handle.net/10316/26963

[24] M. Rahchamani, "Developing and evaluating a low-cost tracking method based on a single camera and a large marker". In: 25th National and 3rd International Iranian Conference on Biomedical Enginnering (ICBME). Tehran, Iran, 29-30 November 2018, DOI: 10.1109/ICBME.2018.8703592.

[25] B. P. Gerkey. AMCL package, http://wiki.ros.org/amcl.

[26] Perspective-n-Point (PnP) transform, https://docs.opencv.org/master/dc/d2c/tutorial_real_time_pose.html.