A Development of Mobile Robot Based on ROS2 for Navigation Application

Phuwanat Phueakthong
School of Mechatronics Engineering
Suranaree University of Technology
Nakhon Ratchasima 30000, Thailand
Email: phuwanat.aerod@gmail.com

Jittima Varagul
School of Manufacturing Engineering
Suranaree University of Technology
Nakhon Ratchasima 30000, Thailand
Email: jittima@sut.ac.th

2021 International Electronics Symposium (IES) | 978-1-6654-4346-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IES53407.2021.9593984

***Abstract*—This paper proposes an automatic navigation mobile robot using Robot Operating System2 (ROS2) with low-cost embedded hardware. Utilizing Data Distribution Service (DDS) in ROS2 makes the ROS2 more safe and reliable than ROS1. Cartographer and Navigation2 projects in ROS2 are used for Simultaneous Localization and Mapping (SLAM) with 2D LIDAR and navigation, respectively. Micro-ROS which utilizes DDS for eXtremely Resource-Constrained Environments micro-XRCE-DDS is used for communication between main embedded computer and microcontroller replaces ROS serial communication which is less reliable. The experiments prove that the robot can perform mapping and navigation tasks. A robot can generate a global trajectory in a static map to the goal point, can re-plan the local path in the local map area to avoid coming dynamic obstacles during the mission and navigate itself to reach the goal.**

Keywords*—*ROS2; navigation; autonomous mobile robot; micro-ROS; SLAM

I. Introduction

Nowadays, mobile robot with autonomous navigation system is used in various works, such as delivery robot in medical work and in industrial factory. Robot Operating System (ROS) is a popular framework that is used in these clever robots. ROS provides useful packages for autonomous robot, such as mapping, localization and navigation package, also provides useful tools for robot development. Using ROS simply the autonomous robot invention. Recently, ROS has been improved to the new version which is called ROS2 and the original ROS version is called ROS1. ROS2 is designed to support real-time robotic systems and, increase reliability and enhance safe data communication [1]. These results come from utilizing Data Distribution Service (DDS) which is an industrial-standard communication middleware [2], in ROS2.

For automate driving, ROS2 provides tools for developed and evaluate performances of the autonomous mobile robot, for example, remote control, remote monitoring tools and simulations. The important part of autonomous navigation robot is Simultaneous Localization and Mapping (SLAM) which obtains data from sensors, such as LIDAR laser scan data, sonar distance and depth cameras data to build a map and localize the robot in the map. In this research, Cartographer is used to generating a 2D grid map for robot navigation.

In this essay, we have invented a differential drive mobile robot with cheap hardware. The robot has Raspberry PI4 RAM 4 GB as the main processor, YD Lidar X4 Laser scanner as a data collector, Motor with encoder and use Raspberry Pi Pico RP2040 microcontroller as motor controller unit.

This article is structured as follows. Section II explains methodology which is the details of design, system organization and software components. Experiments, results and analysis are in section III followed by the final conclusion in section IV.

II. Methodology

A. Mechanical Design and hardware architecture

The robot is divided into three layers to contains all components. The top plate supports LIDAR. The mid-plate and bottom plate support Raspberry Pi4 computer and Pi Pico with motor driver board, respectively. The design from Fusion360 software is in Figure 1.

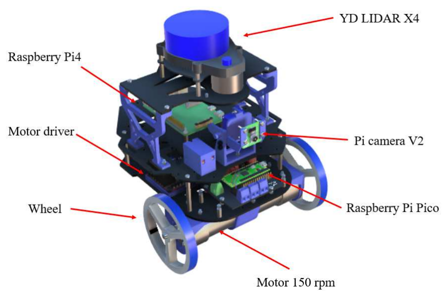


Figure 1. A design of robot.

The selected motor is 150 rpm without load, torque 1.8 kgf-cm and there is quadratic encoder built-in with it. It is used with a wheel which has diameter of 85.1 millimeters. Both wheels are attached at the front of the robot and there is a caster ball locate at the backward of a robot. The Raspberry Pi camera v2.0 is installed in the front for surveying.

B. Elecronic devices integration

Electronic devices consist of the computer, sensors, microcontroller and motors of robot. Details of these devices are in the following. Figure 2 illustrated the hardware architecture of the robot.

The computer used is Raspberry Pi4 RAM4GB with 64GB micro SD-card. It has 40 GPIO pinouts. It is the main processor for robot. LIDAR, Pi camera V2.0 and Raspberry Pi Pico microcontroller connect with it.

The microcontroller used is Raspberry Pi Pico with RP2040 chip. It has dual cores ARM Cortex-M0+ processor with 264KB on-chip SRAM, 2MB onboard QSPI flash and 16 PWN channels. Pico waits for command from Raspberry Pi to control speed of motors and wait for pulse signal from wheel encoders then publish it to Raspberry Pi.

The LIDAR used is YD Lidar X4 which has minimum and maximum ranges are 0.12 to 10 meters. The scan frequency is 6-12 Hz. and 360 degrees scan angle. YD Lidar X4 provides a separate data transfer and power supply USB port which allows users can power the LIDAR with an external power source separately from Raspberry Pi. YD LIDAR X4 requires 12800 bps communication baud rates to work properly.

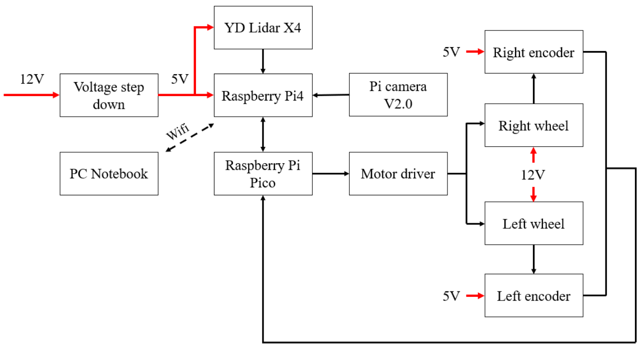


Figure 2. Hardware architecture.

The motor requires 12V power and a built-in PIC16F quadrature encoder which requires 5V power supply. Motor's rated load torque and rated load speed are 1.8 kgf-cm and 112 ± 12 rpm. Gear ratio is 30:1. Motor's power supply is from a 3S LiPo battery.

C. *Robot Operating System2 (ROS2)*

Robot Operating System 2 (ROS) is the second generation Of ROS. ROS2 builds upon Data Distribution Service (DDS) [3]. ROS2 is cleaner than the older version [4]. DDS provides distributed discovery feature (not centralized in ROS1) that allows each ROS2 node can discover each other without ROS master and DDS make ROS2 support real-time operation. [3] propose comparative data about capabilities and performance of ROS2 over ROS1, such as data transmission, supported platforms, Quality of Service (QoS), number of threads and real-time characteristics. ROS2 provides navigation2 and cartographer package which is relevant packages for mobile robot navigation.

ROS navigation2 can be applied in the autonomous navigation mobile robot to find safe way from A to point B. It consists of package for localization, path planning, dynamic obstacle avoidance and etc. Require input for navigation2 are robot's TF (transform) which explains the relative of the reference frame of robot, robot odometry data, sensor data source and map. Then, it will send out the motor's velocity command to control robot [5].

ROS2 used is ROS Foxy Fitzroy which is compatible with Ubuntu 20.04. ROS2 will be installed on both PC (Ubuntu 20.04) and Raspberry Pi4 (Ubuntu mate 20.04).

D. *Micro-ROS (ROS for micro controllers)*

Micro-ROS is ROS for microcontroller. Micro-ROS allows microcontroller can use all major core concepts of ROS with C or C++ programming. Client API of micro-ROS in MCU based on ROS2 client library [6].

Even DDS implementation is lightweight, but the memory footprint is still large too much to bear in embedded systems [9]. This problem leads micro-ROS to use micro-XRCE-DDS (DDS for eXtremely Resource-Constrained Environments) middleware by eProsima which is a middleware for embedded systems. It supports WiFi, 6LoWPAN, Bluetooth, serial transport and UDP communications. Micro-ROS is supported by RTOSes, FreeRTOS, Zephyr and NuttX. In this paper, micro-ROS will be added to Raspberry Pi Pico to create micro-ROS node in Pico which waits for the command from Raspberry Pi to control speed of motor and obtain feedback from encoders.

E. *Mapping with Cartographer*

Cartographer is an open-source package for real-time simultaneous localization and mapping (SLAM) from Google for various sensor configuration and platforms in 2D and 3D [7]. Cartographer uses graph optimization algorithm which use lower computing resource than particle filter method [8].

F. *AMCL and Navigation*

ROS2 navigation stack provides AMCL (Adaptive Monte Carlo Localization) package for localization which use map data, sensors data and odometry data from robot with particle filters, Monte Carlo Localization (MCL) and Kullback-Leibler Distance (KLD) sampling method to estimate pose of robot in an environment [10]. The number of samples will be chosen by KLD using consideration of uncertainty [11]. The number of samples influences the efficiency of particle filters. AMCL will scatter particles randomly in map. When the robot moved, the particles will move together. The particle which matched the sensor data will obtain the higher weight than others. The process occurs repeatedly and the particles will gradually converge to the actual robot position over time [12].

In this paper, grid-based map is used to represent or model the environment of robot. Map data is a static map. It will be taken by path planner to generate trajectory for robot. ROS2 divides path planner into global path planner and local path planner. The global planner will generate path for known obstacles over a map. Nav2Fn is a plugin for global planner in ROS 2 which use A* or Dijkstra's algorithm. The selected algorithm is A* algorithm which is more efficient algorithm for finding the shortest path than Dijkstra's algorithm [13]. In local path planning, local planner is used to avoiding dynamic obstacles which there is no on map. The selected local planner is DWB local planner which is the upgraded version of DWA local planner in ROS1.

III. Experiment and Results

A completed robot for testing in shown in Figure3. Laboratory area is testing environment. Testing are map generation test and navigation test.

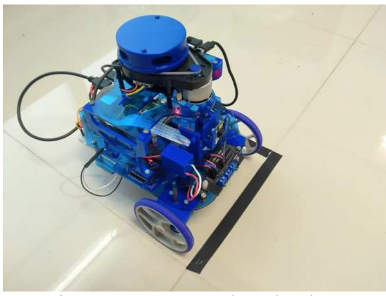


Figure 3. A completed robot.

A. Map generation

A robot was controlled by teleoperation from PC to explore the area. Cartographer used odometry data, and laser scan data as input for map generation. After process, result map is shown in Figure 4. The black area is walls and white area is space. Map size is 448×343 pixels. It consist of, .pgm and .yaml file which are the map picture and map description file, respectively.

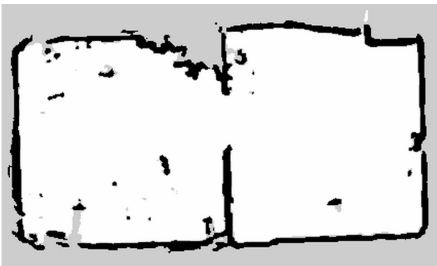


Figure 4. Map from exploration.

B. Navigation test

We divided navigation test into two parts, navigation in only static map test and navigation in static map with unknown or dynamic obstacle. During navigation, Rviz2, a software for 3D data visualization in ROS2, was used to visualize the data and send navigation goal to a robot. When navigation2 is activated, a map which obtained in prior was called and used to create global cost map and local cost map as shown in Figure 5. Planner in ROS2 uses global cost map to plan trajectory of a robot until the end of operation and uses local cost map to plan trajectory of a robot to avoid coming obstacle. Local cost map is only a small area around a robot. We used only 3×3 meters local cost map. Selected position tolerance of navigation goal in Navigation2 controller parameters of x and y coordinates is 25 centimeters and a robot footprint radius is 15 centimeters

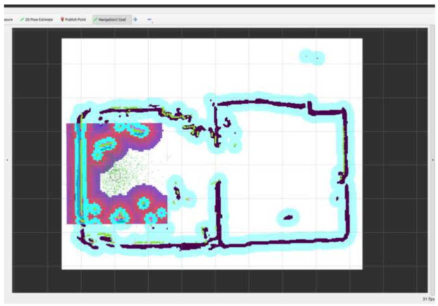
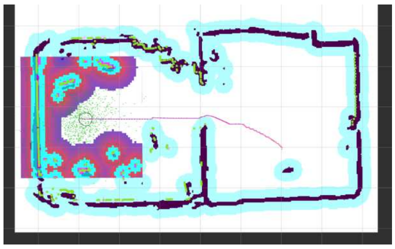
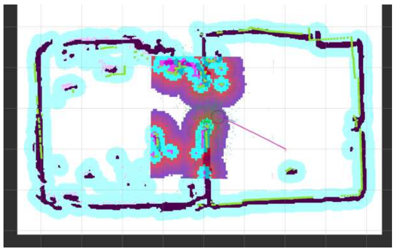


Figure 5. global and local cost map.

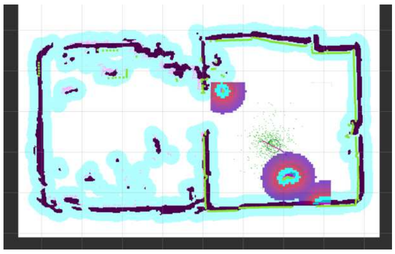
We set start point of the robot in boundary as in Figure 3 and sent navigation goal to a robot. This point is the same for all test cases. After it obtained a goal point, it generated a global path as in Figure 6 and started moving.



(a)



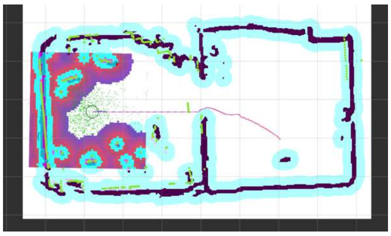
(b)



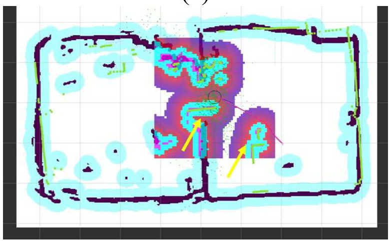
(c)

Figure 6. (a) A robot start moving. (b) A robot followed path. (c) A robot approached to destination.

Above figures illustrate a moving of robot follow a path from global planner without unknown obstacle. When we added new unknown obstacles to the map, a local planner plays an important role. When a robot detected unknown obstacles. Local planner tried to generate new trajectory in local area to avoid obstacles as in Figure 7. Compared to Figure 6(b) the yellow arrow in Figure 7(b) points to new added dynamic obstacles.



(a)



(b)

Figure 7. (a) A robot started moving to goal. (b) A robot tried to avoid unknown obstacles.

From 5 times testing in both navigation case, the results of position and travel time of a robot in each navigation case are shown in Table I and Table II.

TABLE I. POSITION ERROR OF GOAL AND MOVING TIMES OF ROBOT IN STATIC MAP WITHOUT UNKNOWN OBSTACLES.

Experiment	**Position error**		**Times [s]**
	x [cm]	***y [cm]***	
1	8.30	-2.54	14.63
2	0.72	-3.39	14.42
3	11.88	-14.01	14.60
4	5.42	-7.01	15.03
5	8.45	-12.17	14.93
average	6.96	-7.82	14.72

TABLE II. POSITION ERROR OF GOAL AND MOVING TIMES OF ROBOT IN STATIC MAP WITH UNKNOWN OBSTACLES.

Experiment	**Position error**		**Times [s]**
	x [cm]	***y [cm]***	
1	1.39	4.57	15.62
2	6.79	6.91	15.77
3	5.97	14.99	15.99
4	2.60	9.85	15.59
5	5.21	11.80	16.09
average	4.392	9.62	15.81

A robot can automatically navigate itself to destination without collision. A robot can detect unknown obstacles and re-plan the trajectory in local cost map to avoid all obstacles. With the dynamic obstacles, a robot took a longer time to reach the destination. A maximum of goal position tolerances is 14.99 centimeters in Table II which is in range of defined x and y goal tolerances.

IV. CONCLUSSION

In this article, a design, software system of an autonomous navigation robot based on ROS2 with low-cost embedded hardware is presented. ROS2 and micro-ROS is implemented in embedded computer and microcontroller of robot. A vehicle is capable of Teleoperation with command from PC, Mapping with Cartographer, Localizing with AMCL and Navigation with Navigation2 project. Selected algorithm for global path planning is A* algorithm and selected planner for local path planner is DWB planner. A robot can navigate to the goal, avoid the dynamic obstacle in a map and reached the goal point successfully with acceptable x and y position tolerances. The maximum tolerance of goal position is 14.99 centimeters.

REFERENCES

[1] Y. Tang et al., "Response Time Analysis and Priority Assignment of Processing Chains on ROS2 Executors," IEEE Real-Time Systems Symposium (RTSS), pp. 231-243, 2020.

[2] H. Cui, J. Zhang and W. R. Norris, "An Enhanced Safe and Reliable Autonomous Driving Platform using ROS2," IEEE International Conference on Mechatronics and Automation (ICMA), pp. 290-295, 2020.

[3] Y. Maruyama, S. Kato and T. Azumi, "Exploring the performance of ROS2," International Conference on Embedded Software (EMSOFT), pp. 1-10, 2016.

[4] C. Heggem, N. M. Wahl and L. Tingelstad, "Configuration and Control of KMR iiwa Mobile Robots using ROS2," 3rd International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS), pp. 1-6, 2020.

[5] S. Macenski, F. Martín, R. White and J. G. Clavero, "The Marathon 2: A Navigation System," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2718-2725, 2020.

[6] Features and Architecture of Micro-ROS, "micro.ros.org" [Online]. Available: https://micro.ros.org//docs/overview/features/

[7] Cartographer with ROS Integration, "https://google-cartographer-ros.readthedocs.io/en/latest/."

[8] X. Zhang, J. Lai, D. Xu, H. Li, and M. Fu, "2D Lidar-Based SLAM and Path Planning for Indoor Rescue Using Mobile Robots," Journal of Advanced Transportation, 2020.

[9] S. Dehnavi, D. Goswami, M. Koedam, A. Nelson and K. Goossens, "Modeling, implementation, and analysis of XRCE-DDS applications in distributed multi-processor real-time embedded systems," 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1148-1151, 2021.

[10] I. Wasisto, N. Istiqomah, I. K. N. Trisnawan and A. N. Jati, "Implementation of Mobile Sensor Navigation System Based on Adaptive Monte Carlo Localization," 2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA), 2019, pp. 187-192

[11] R. Mishra and A. Javed, "ROS based service robot platform," 4th International Conference on Control, Automation and Robotics (ICCAR), pp. 55-59, 2018.

[12] L. Zhi and M. Xuesong, "Navigation and Control System of Mobile Robot Based on ROS," IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 368-372, 2018.

[13] Z. Li, Y. Xiong and L. Zhou, "ROS-Based Indoor Autonomous Exploration and Navigation Wheelchair," 10th International Symposium on Computational Intelligence and Design (ISCID), 2017, pp. 132-135, 2017.