*Key words:*
*mobile robot, robotics, SLAM,*
*localization, mapping, ROS*

Łukasz CHOJNACKI*, Damian BIERNAT*, Kamil KIEŁBASA*

# SIMULTANEOUS LOCALIZATION AND MAPPING OF MOBILE ROBOT USING ROBOT OPERATING SYSTEM

Robot is a device that should autonomously and intelligently move in an environment with static obstacle and people without any collision. In that case, a robot should create a map of an environment, localize itself and move to a given point. The presented problem is called Simultaneous Localization and Mapping (SLAM). Robot Operating System is a robotic framework, which has been already implemented algorithms solving most common robotic problems. The paper presents practical aspects of building differential drive mobile robot solving SLAM problem like how to connect single board computer (high-level tasks e.g. localization, mapping, motion planning) with microcontroller (low-level tasks e.g. controlling motors and reading impulses from encoders), and how to use Navigation Stack package from ROS framework.

## 1. INTRODUCTION

Nowadays mobile robots can perform many different tasks such as line following, maze solving, cleaning a floor and moving to dangerous places. In order to solve these tasks a robot should know its surroundings, where actually it is located and how to move to the next position. Various methods and algorithm solving simultaneous localization and mapping problem have been presented by researchers [1-3].

Robot Operating System [4] is an open-source robotic framework that allows building complex robotic systems connecting nodes by its inputs and outputs. The connection between nodes is called topic and an information send through the topic is called mes-

* Student Interest Group ‘’KoNaR”, Chair of Cybernetics and Robotics, Faculty of Electronics, Wroclaw University and Technology, Z. Janiszewskiego11/17 50-372 Wroclaw

sage. There are implemented a lot of ROS packages (set of nodes) that solve many robotics problems, e.g. image-processing, image-recognition, localization, mapping, trajectory planning, inverse kinematics. This paper presents practical approach of building mobile robot solving SLAM problem using ROS framework. For this reason, a robot was constructed.

Turtlebot (Fig. 1)is a differential drive platform with Raspberry Pi 3 as a main computer and STM Nucleo-L476RG board as a low-level controller. The robot uses several sensors as RPLidar A2 and Pololu's magnetic encoders. X-NUCLEO-IHM12A1 expansion board with STSPIN240 dual brush DC motor driver is used to control two Pololu's micro motors. Figure 2 presents connection of hardware components which were connected by Łukasz Chojnacki.

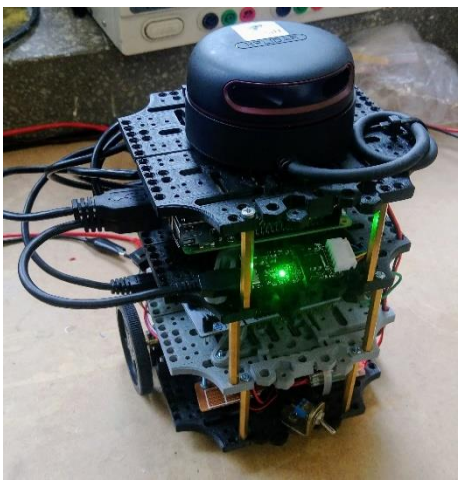

Fig. 1. Turtlebot

## 2. PRACTICAL ASPECTS

### 2.1. COMMUNICATION BETWEEN BOARDS

UART interface was used to exchange data between Raspberry Pi and Nucleo board. Data can be send and received at the same time in a form of a byte stream. In this case, suitable mechanism for serializing structured data was used to. This mechanism is implemented in Protocol Buffer framework developed by Google. Advantages of serializing data is size reduction and easy separation of sending information. Encoder values and speed of motors data are sending through this communication channel.

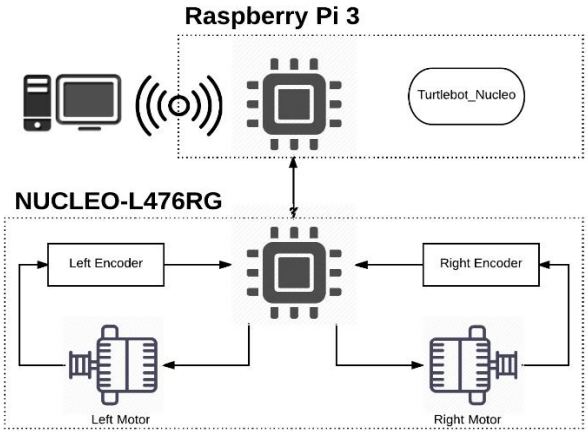


Fig. 2. Connection of hardware components

### 2.2. MOTORS SPEED

Generate correctly speed is depending on two parts:

1. The task of Turtlebot_nucleo is used in order to convert for angular velocity and next for the voltage value.

2. The task of Nucleo is to set PWM value correctly.

To control speed Pulse-Width Modulation technique is used. PWM is a quadrature signal with variable fill pulse. In this case, the maximal fill is equal 6V. Turtlebot_nucleo guarantee to send value correctly to Nucleo board. PWM signal is generated by using Timer. The setting of Timer allows getting resolution of speed at 1% point.

### 2.3. ROBOT POSITION - ENCODERS

Magnetic encoders have been used to define displacement of the robot. Encoders send quadrature signal, in microprocessor, it is understanding by one of Timer setup. Encoder allows to track the number of revolutions each wheel has made. For one wheel rotation occurs 360 pulses, it gives displacement with 1º of wheel circuit. Timer increment or decrement encoder information, it depends on moving forward or back.

The total distance can be computed using the following equations [6]:

$$x' = x + D_C \times \cos(\Theta) \tag{1}$$

$$y' = y + D_C \times \sin(\Theta) \tag{2}$$

$$\Theta' = \Theta + \frac{D_L - D_R}{L} \tag{3}$$

$x$ - temporary displacement in x direction,
$x'$ - main value of robot displacement in x direction,
y - temporary displacement in y direction,
y' - main value of robot displacement in y direction,
$\Theta$ - temporary displacement in Θ direction,
$\Theta'$ - main value of robot displacement in Θ direction
$D_C$ - average value of both encoders.
Distance made by each wheel is calculated as follows

$$D_{L,R} = 2 \times \Pi \times R \times \frac{\Delta tick}{N} \tag{4}$$

and number of pulses made within each time interval in milliseconds:

$$\Delta tick = tick' - tick \tag{5}$$

*tick'* - new value from encoders,
*tick* - saved last encoder value,
*N* - tick per revolution,
*R* - radius of wheel.

## 2.4. TURTLEBOT NUCLEO NODE

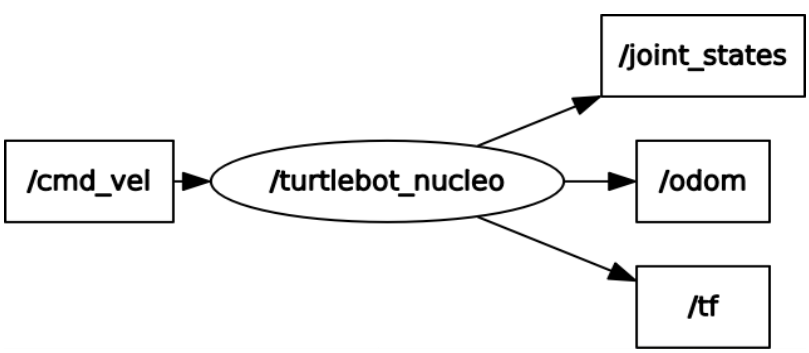


Fig.3. Turtlebot_nucleo node

Programming Structure in ROS depends on subscribing or publishing information

by node to Topic. In this case, the node is publishing to three topics using information from encoders:

- joint_States – robot distance in radians,
- odom – information about robot position,
- tf – information about robot position,

moreover, subscribing from one topic:

- cmd_vel – node get the value of robot speed.

Robots speed is calculated for each wheel, and it is converted to voltage values.

## 3. NAVIGATION STACK

The main aim of the ROS navigation package is to move a robot from the start position to the goal position, without making any collision with an environment. The ROS Navigation package comes with an implementation of several navigation related algorithms which can easily help implement autonomous navigation in the mobile robots. These algorithm are presented in [5] and grouped into the following packages:

- gmapping : The gmapping package is an implementation of an algorithm called Fast SLAM which takes the laser scan data and odometry to build a 2D occupancy grid map.
- amcl : amcl is a method to localize the robot in map. This approach uses particle filter to track the pose of the robot with respect to the map, with the help of probability theory. In the ROS system, amcl can only work with maps which were built using laser scans.
- map server : Map server package allows us to save and load the map generated by the costmap-2D package.
- move_base : The main function of move_base is to move a robot from its current position to a goal position with the help of other Navigation nodes.

## 4. SIMULATION

Robot simulation is a very important step to test a robotic system and its behavior. A well-thought-out simulator makes it easy to test algorithms and robot design in realistic scenarios. Gazebo is an accurate and efficient robotic simulator with robust physics engine, high-quality graphics, and convenient graphical interface. Moreover, Gazebo is open source and is used by professional researchers. This simulator is well integrated with ROS. The next advantage is that the simulated robot software can be easy ported to the real robot.

The simulation consists of a robot model and an environment. An apartment with many rooms has been chosen as an environment. Kinematics and dynamics of a robot

is described in URDF (Unified Robot Description Format) file. In the figure 2. is presented the robot performing go-to-goal task. The green arrow represents a goal position, and a blue line is a path that robot should follow.

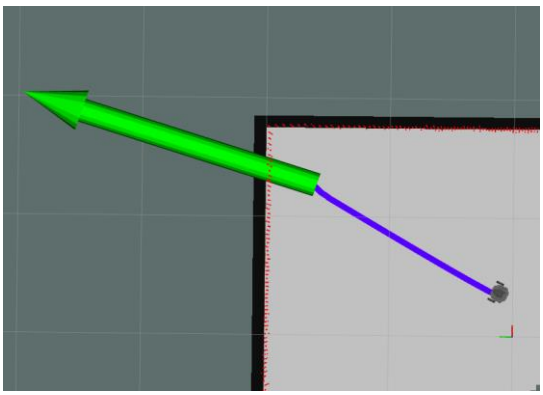

Fig. 4. View of simulation

## 5. CONCLUSIONS

A robot that can move in an environment without collision should perform several closely-related tasks parallelly. These tasks are the following: create a map of an environment, localize itself and plan a free collision movement to the goal position. All these tasks can be considered as a Simultaneous Localization and Mapping (SLAM) problem. Implementation of algorithms that can solve this problem can be time-consuming and tedious. Fortunately, in ROS framework have been implemented state of the art algorithms that are open-source and free.

## REFERENCES

[1] SICILIANO B., OUSSAMA K., ,,Springer handbook of robotics'', Springer, 2016.
[2] NOURBAKHSH, ILLAH R., and Roland SIEGWART. "Introduction to autonomous mobile robots." The MIT Press, Cambridge, Massachusetts, England, ISBN 0 262.19502 (2004): 142-150.
[3] THRUN Sebastian, Wolfram BURGARD, and Dieter FOX. "Probabilistic Robotics MIT Press 2005." ISBN: 0-262-20162-3.
[4] QUIGLEY, Morgan, et al. "ROS: an open-source Robot Operating System." ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.
[5] LENTIN Joseph, ,,Mastering ROS for Robotics Programming'', 2015
[6] http://www.robotnav.com/position-estimation/ "Position Estimation" ,September 2013