SSRG International Journal of Computer Science and Engineering
ISSN: 2348 – 8387 /doi:10.14445/23488387/IJCSE-V8I1P107

Volume 8 Issue 1, 39-43, January 2021

Researching And Building Environmental Awareness System for Self-Propelled Three-Wheeled Omni Robot based on Algorithm EKF-SLAM And ROS Operating System

Pham Minh Thai

#University of Economics - Technology for Industries, Viet Nam

Abstract — *Motion trajectory is an important problem in motion control for autonomous robots, in which the environmental perception system plays a core role because it provides information about the operating environment for the robot. The environmental awareness system is responsible for mapping and self-locating the robot in the operating environment (SLAM - Simultaneous Localization and Mapping) and detecting obstacles during the robot's movement. The paper presents the design and construction of an operating environment awareness system for three-wheeled Omni robots based on the EKF-SLAM algorithm and the ROS (Robot Operating System) robot programming operating system. The results obtained show the effectiveness of the cognitive system built.*

Keywords — *Robot operating system, Rviz, Robot Omni, Simultaneous localization and mapping (SLAM), EKF-SLAM*

I. INTRODUCTION

Robots bring many conveniences in modern civilized life, helping to replace simple human labor, improving labor productivity and product quality, especially changing production methods, and quickly speeding up socio-economic development. The intelligence in these autonomous robot systems depends on whether the robot can calculate a suitable trajectory for the environment in which it operates. Therefore, perceptions of the surrounding environment, including fixed information or changes in the environment, are the prerequisites that directly affect the robot's movement. Therefore, a cognitive system that provides robots' environmental information is important in autonomous robot systems [3, 4]. A fully autonomous robot system typically consists of two main parts: the trajectory and the orbital control systems. In particular, the system of setting up the motion trajectory needs information about the operating environment. The design of a system capable of providing environmental information has been focused on in [6-9]. These studies focus on nonholonomic robot systems, wheelless or wheelless robot systems. However, these systems have not been designed for omnidirectional robot systems holonomic form. The robot's location mapping and estimation in its operating environment (SLAM) are the two most important information when the robot is operating in an unknown environment. The SLAM problem arises in navigating mobile robots through unspecified environments without maps [1, 2, 5, 6, 8, 11]. Techniques using robotic probabilities have been studied to suggest SLAM problem-solving techniques [14] and [15]. When the ROS operating system came into existence, the construction of robotic systems using SLAM techniques for mapping and positioning was more and more popular, as shown in [5, 8, 9] and [10]. SLAM problem-solving techniques derive input from the depth sensors, distance sensors, and 3D pixels and then compute values for the robot's mapping and positioning. In addition, the signals from sensors received during travel are also used to detect dynamic obstacles that suddenly appear. This paper presents a perception system for an Omni-directional robot based on SLAM technology and ROS platform [12, 13, 16]. The content of the article is divided into 5 main parts. The kinetic model for the Omni robot is presented in section 2. Next, the paper's main contribution is designing a cognitive system based on the EKF-SLAM technique and ROS platform in section 3. Part 4 is the conclusions. Simulation and experimental results to verify the feasibility and effectiveness of the system. Finally, part 5 is the conclusion and research direction.

II. MODEL OF SELF-PROPELLED ROBOT OMNI THREE-WHEEL

The robot kinetic model will be used to input the EKF-SLAM algorithm to calculate the robot's position displacement at a time.

To control the robot to move flexibly, we need an algorithm controlling the robot's omnidirectional wheel movement system to give each wheel the necessary speed for each specific movement. Figure 1 shows the installation parameters for a three-wheel omnidirectional traveling base system.

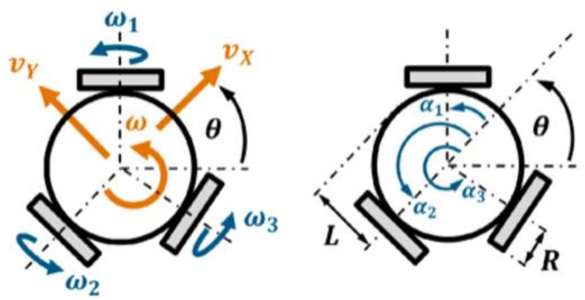


Figure 1: Parameters of the moving base system

Based on these parameters, we can give the kinetic equation of the system as follows:

$$\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} = \frac{R}{3}\begin{bmatrix} \sin\alpha_1 & \sin\alpha_2 & \sin\alpha_3 \\ -\cos\alpha_1 & -\cos\alpha_2 & -\cos\alpha_3 \\ -1/\mathrm{L} & -1/\mathrm{L} & -1/\mathrm{L} \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} = M\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \quad (2)$$

The above kinetic equation with input is the angular velocity of each wheel, from which we can calculate the output as the linear velocity in the 2 directions x, y, and angular velocity ω of the whole robot. From there, depending on the movement of the robot forwards, backward, or forwards, we can reverse the speed of each wheel with the following formula:

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = M^{-1}\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \quad (3)$$

III. AWARENESS SYSTEM FOR SELF-PROPELLED ROBOT

A. Building EKF-SLAM Algorithm for Omni Robot

The Extended Kalman Filter (EKF) is a mathematical tool that can estimate variables of various processes for nonlinear systems. It works by linearizing the nonlinear state dynamics and measurement models. It is widely used in robotics engineering, popular in navigation, navigation, and control applications. This type of filter works very well in practice, and that is why it is often deployed in embedded control systems and because robots need to estimate the process variables accurately. The Extended Kalman filter is a smarter way to integrate measurement data into an estimate by realizing that the measurements are noisy and that they should sometimes be ignored or only slightly affect the estimate. Thai. It smooths out the effect of noise in the estimated state variables by combining more information from more reliable data from unreliable data. The user can tell the extended Kalman filter how much noise in the system, and it calculates a position estimate taking into account the noise.

The robot system for the Kalman algorithm requires IPS, wheel encoder, IMU. I used to embed EKF in the Central Processing Unit on the robot. The input data is a 2D data format that includes: Coordinates from IPS (MarvelMind Beacon Indoor Position System), Velocity from Wheel Encoder, Revs and Angular Velocity as measured with a Gyro sensor Gyroscope, Accelerometer measured by sensor Accelerometer, 2 sensors are integrated into one 6DOF - Sensor is called IMU-MPU6050. The EKF algorithm is divided into 3 parts: Initialization and linearization, Prediction, and Update. Assume that the robot has *x y* coordinates from the IPS, linear velocity v_x v_y from the Wheel Encoder, linear acceleration a_x a_y y_{aw} v_{yaw}, direction, and angular velocity IMU. Our goal is to predict, update, and process 2D coordinates for the robot. So the kinetic and kinetic equations for mobile robots:

$$\hat{x}_i = \begin{bmatrix} v_i \\ v_{x_i} \\ \alpha_{x_i} \end{bmatrix} = \begin{bmatrix} x_{i-1} + \Delta t \cdot v_{x_{i-1}} + \frac{1}{2} \cdot \Delta t^2 \cdot \alpha_{x_{i-1}} + z_{1_{i-1}} \\ v_{x_{i-1}} + \Delta t \cdot \alpha_{x_{i-1}} + z_{2_{i-1}} \\ \alpha_{x_{i-1}} + z_{3_{i-1}} \end{bmatrix} \quad (4)$$

$$\hat{x}'_i = \begin{bmatrix} v'_i \\ v'_{x_i} \\ \alpha'_{x_i} \end{bmatrix} = \begin{bmatrix} x_i + u_{1_i} \\ v_{x_i} + u_{2_i} \\ \alpha_{x_i} + u_{3_i} \end{bmatrix} \quad (5)$$

All equations are done to get x_i updates, y_i computations are similar. The relationship of each state to the previous state is shown as follows:

$$x_i = f\left(x_{i-1}\right) + z_{i-1} \quad (6)$$

Where x_i is the state parameter of the signal at the time *i*, *f* is a nonlinear function representing x_i x_{i-1}.

In addition, the 8 parameters, including the coordinates, linear velocity, linear acceleration, angle of rotation, and angular velocity, are represented by 3 vectors. Because the unit and scale of the sensor data may not be the same as the unit and scale of the Measurement. So the function transforms the state into a measurement system represented:

$$x'_i = h\left(x_i\right) + u_i \quad (7)$$

Where x_i is the state parameter of the signal at the time *i*, *k* is the signal mapping the function to the measuring space containing the signal x_i. z_{i-1} and u_i are the process and observed disturbances. These two types of noise have a multivariate Gaussian form with covariance, *P* and *Q*, respectively. *f* and *k* can be used to calculate and predict the state of a signal. However, if these two functions are used, the robot will not use the covariance to estimate. The Jacobi matrix is used interchangeably with its derivative for the calculation.

Let F and K be the Jacobi matrices of the partial derivatives of f and k, for x_i, F, and K of the form:

$$F = \frac{\partial f}{\partial x} | \hat{x}_{i-1}, z_{i-1} \tag{8}$$

$$K = \frac{\partial k}{\partial x} | \hat{x}_i \tag{9}$$

$$\hat{x}_i = F \cdot \hat{x}_{i-1} + z_{i-1} \tag{10}$$

$$\hat{x}_i' = K \cdot \hat{x}_i + u_i \tag{11}$$

For z_{i-1}=0, u_i=0, the Jacobian matrix F, K is:

$$F = \begin{bmatrix} 1 & \Delta t & \Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

$$K = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

Here's an estimate of the current covariance over time, represented by the equation, which estimates the predicted covariance:

$$Q'_i = F \cdot Q_{i-1} \cdot F^T + P \tag{14}$$

Q_{i-1} is the covariance matrix of the signal at the time *i - 1*, F^T is the displacement matrix of *F*, *P* is the noise representation matrix in the estimation process, set manually. As for the first signal received, since there is no previous signal for calculating the state, the matrix Q_0 will be manually set. This is also the first initial step.

$$Q'_1 = F \cdot Q_0 \cdot F^T + P \tag{15}$$

The matrix Q'_i will be included to update the signal's new state. I use the Joseph-form covariance update equation to promote the filter's stability by ensuring that Q_i remains half positive. Determined. The specific steps are as follows:

First, let's calculate Kalman Gain:

$$K_G = Q'_i \cdot K^T \left(K \cdot Q'_i \cdot K^T + M \right)^{-1} \tag{16}$$

Second, update the new state estimate:

$$x_i = \hat{x}_i + K_G \left(x'_i - K \cdot \hat{x}_i \right) \tag{17}$$

Finally, update the new covariance estimate:

$$Q_i = \left(I - K_G \cdot K \right) Q'_i \left(I - K_G \cdot K \right)^T + K_G \cdot M \cdot K_G^T \tag{18}$$

K^T is the displacement matrix of *H*, *M* is the covariance matrix representing the signal noise of the data from the sensors. I also write a program that calculates *M* with the equation:

$$M = \begin{bmatrix} \sum x & 0 & 0 \\ 0 & \sum v_x & 0 \\ 0 & 0 & \sum a_x \end{bmatrix} \tag{19}$$

$$\sum x = \frac{1}{i-1} \sum_{n=1}^{i} \left(x_n - \bar{x} \right)^2 \tag{20}$$

Q_n^-: a priori estimation error covariance; Q_n: posterior estimation error covariance; *B*: the partial derivative Jacobian matrix of f with *x*; *Z*: partial derivative Jacobian matrix of f with *z*; *D*: entrance noise covariance matrix

Estimation correction

$$K_n = Q_n^- H_n^T \left(H_n Q_n^- H_n^T + V_n R_n V_n^T \right)^{-1} \tag{21}$$

$$\hat{x}_n = \hat{x}_n^- + K_n \left(s_n - k\left(\hat{x}_n^-, 0 \right) \right) \tag{22}$$

$$Q_n = \left(I - K_n H_n \right) Q_n^- \tag{23}$$

where $\hat{x}_n$: estimates the posterior state at step n with the measurement s_n; K_n: Kalman coefficient; *R*: measurement noise covariance matrix; *H*: the partial derivative Jacobian matrix of *k* with *x*; *V*: the partial derivative Jacobian matrix of *k* with *v*

Finally, get and use the *x* and *y* coordinates after the update instead of the raw data from the IPS.

B. Building Awareness System for Omni Robot based on ROS Platform

The ROS operating system is a flexible platform for programming software for robot systems, as shown in Figure 2. It includes tools and libraries that simplify complex robot systems by combining Match the robot platforms. More than that, ROS is built to facilitate the convenient development and combination of the robot software.

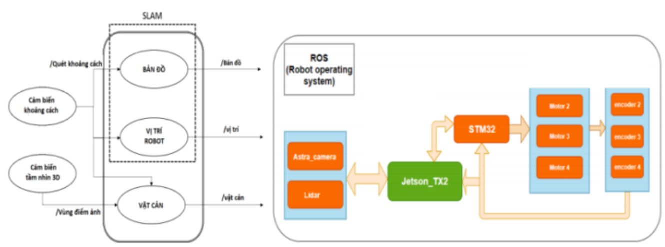


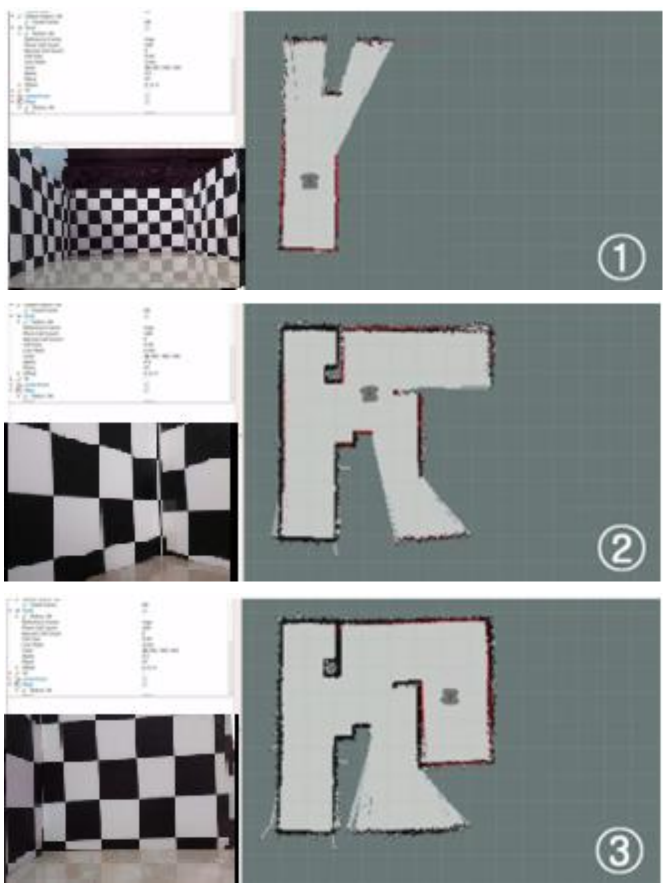
Figure 2: ROS-based perception system and hardware configuration of the robot system

The EKF-SLAM algorithm will be implemented by 2 nodes, namely map and robot position, outputting data about the map and robot location.

The cognitive system also has the function of detecting obstacles based on the signal of the distance sensor and 3D pixels (3D point cloud). An area that surrounds the robot is called the local coast map region. The local coast map area selected is a square area with the robot in the center. The square's size is chosen to ensure that the robot can detect obstructions from a safe distance. Besides, the value of 3D pixels is also limited to the impact area inside the cost map.

IV. SIMULATION RESULTS

The simulation was performed on the simulation software Gazebo by ROS. The white areas are the perceived environmental zones. The black lines are the obstacles that the robot identifies and mapped using the EKF-SLAM technique. The EKF-SLAM continuously calculates and maps the environment based on these data and locates the robot on the map. Once moved to locations sufficient to collect data for the entire environment mapping as shown in Figure 3, this map data can be saved for reuse in succession. Conducting environmental re-mapping.



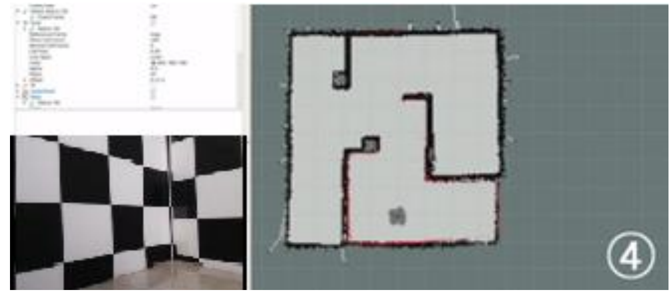


Figure 3: Robot SLAM process and environment mapping result

V. CONCLUSIONS

The paper has designed and built the environmental awareness system for Omni three-wheeled self-propelled robot based on the EKF-SLAM algorithm and ROS operating system. The system is tested on both a simulated and a real robot model. The receiving system The robot's formula is built based on Astra Camera and Rplidar A2 data during the robot's motion. After that, the environment will be mapped, and the robot can simultaneously locate in that environment. The perception system also provides information about obstacles that appear while the robot moves. The results obtained show the practical effectiveness of the built-in cognitive system.

ACKNOWLEDGMENT

This study was supported by the University of Economics - Technology for Industries, Viet Nam; http://www.uneti.edu.vn/.

REFERENCES

[1] Roan Van Hoa, L. K. Lai, Le Thi Hoan, “Mobile Robot Navigation Using Deep Reinforcement Learning in Unknown Environments”, SSRG International Journal of Electrical and Electronics Engineering (SSRG-IJEEE), pp. 15-20, Volume 7 Issue 8, Aug 2020.

[2] Pham Ngoc Sam, Tran Duc Chuyen, “Research and Designing a Positioning System, Timeline Chemical Mapping for Multi-Direction Mobile Robot,” SSRG International Journal of Electronics and Communication Engineering, Volume 7 Issue 11, 7-12, November 2020.

[3] Hashemi, Ehsan, Maani Ghaffari Jadidi, and Navid Ghaffari Jadidi, "Model-based PI– fuzzy control of four-wheeled Omni-directional mobile robots," Robotics and Autonomous Systems, (2011), pp. 930-942.

[4] L. a. S. H. Lin, "Modeling and Adaptive Control of an Omni-Mecanum-Wheeled Robot," Intelligent Control and Automation, vol. 4 (2013), pp. 166-179.

[5] R. L. e. a. Guimarães, "ROS navigation: Concepts and tutorial," Springer, Cham, (2016), pp. 121-160.

[6] A. a. P. A. Pajaziti, "SLAM–map building and navigation via ROS," International Journal of Intelligent Systems and Applications in Engineering, vol. 2(4) (2014), pp. 71-75.

[7] A. I. A. a. E. M. Buyval, "Comparative analysis of ROS-based monocular SLAM methods for indoor navigation," Ninth International Conference on Machine Vision, vol. 10341 (2016).

[8] Z. e. a. An, "Development of Mobile Robot SLAM Based on ROS," International Journal of Mechanical Engineering and Robotics Research, (2016), pp. 47-51.

[9] R. e. a. Giubilato, "An evaluation of ROS-compatible stereo visual

SLAM methods on an nVidia Jetson TX2," Measurement, (2019), pp. 161-170.
[10] R. N. a. M. K. B. Darmanin, "Autonomous Exploration and Mapping using a Mobile Robot Running ROS," ICINCO, 2016.
[11] R. K. e. a. Megalingam, "ROS based autonomous indoor navigation simulation using SLAM algorithm," Int. J. Pure Appl. Math, (2018), pp. 199-205.
[12] P. e. a. Marin-Plaza, "Global and local path planning study in a ROS-based research platform for autonomous vehicles," Journal of Advanced Transportation, (2018).
[13] C. F. H. a. T. B. Rösmann, "Online trajectory planning in ROS under kinodynamic constraints with timed-elastic-bands," Robot Operating System (ROS). Springer, Cham, (2017), pp. 231-261.
[14] F. e. a. Albers, "Online Trajectory Optimization and Navigation in Dynamic Environments in ROS," Robot Operating System (ROS). Springer, Cham, (2019), pp. 241-274.
[15] S. Thrun, "Probabilistic Robotics," Communications of the ACM, (2002), pp. 52-57.
[16] S. W. B. a. D. F. Thrun, Probalistic robotics, (2006).