Navigation for Two-Wheeled Differential Mobile Robot in the Special Environment

Tran Thuan Hoang[1,2], Nguyen Ngo Anh Quan[1], Vo Chi Thanh[1], and Tran Le Thang Dong[1,2(✉)]

[1] Center of Electrical Engineering, Duy Tan University, Danang 550000, Vietnam
tranthangdong@duytan.edu.vn
[2] Faculty of Electrical-Electronic Engineering, Duy Tan University, Danang 550000, Vietnam

Abstract. In this paper, a new navigation method for the two-wheeled differential mobile robot operating in a specific environment is proposed. The orientation angle and position of the mobile robot are estimated by the data collected from the inertial sensors. The encoder sensor at the motor axis is calculated by the Kalman filter algorithm. Further, angular and positional errors are corrected as the robot passes through magnetic reference points installed under the floor on virtual paths. Next, the control function Lyapunov and the program to avoid obstacles by the VFH+ method in local space were also developed to help the robot reach the destination safely. Simulation results and some analysis clarify the correctness of our proposed algorithm. All of the results show promise and applicable.

Keywords: Mobile robot · Kalman filter · Robot tracking control · Lyapunov function · VFH+

1 Introduction

Nowadays, the world is starting to enter the Industrial Revolution 4.0, mobile robots are increasingly widely used in life. With the function of automatic transportation, flexibility, and ability to replace people to perform jobs more efficiently and effectively, it is widely applied in the manufacturing and warehousing industry. In particular, mobile robots are very important in hazardous environments or isolated areas [1–3].

Ignoring the operation of the parts attached to the robot base, the navigation problem for the safe movement of the base from a starting point to a destination, or called "*navigation for mobile robots*", is considered as major in current robotics research. Navigation is not so different from human behavior. To deal with this problem, the robot needs positioning, mapping if necessary, path planning, and exporting the path control as well as avoiding obstacles on the track.

With today's technology, many navigation solutions have been researched and developed, which can divide the path system for robots into two types: *fixed path* and *free path.*

N.-S. Vo et al. (Eds.): INISCOM 2022, LNICST 444, pp. 167–183, 2022.
https://doi.org/10.1007/978-3-031-08878-0_12

The fixed paths are defined as physical paths including magnetic tapes, wires, paint lines, etc. They are all sticked on the ground where the mobile robot stops by or goes through. This system has the following benefits: low cost, ease of deployment, and high reliability. However, this technology has the disadvantage of being less flexible and adaptable to the environment, particularly in outdoor locations or where there is a lot of dirt, which raises the path's maintenance costs.

The free path is considered as a virtual path that is generated and decided using data from current navigation sensing devices on a computer including ultrasonic sensor, 3D camera, laser sensors., LiDAR sensor, inertial sensor, etc. The advantages of this approach include a flexible and highly efficient operating system. These technologies, on the other hand, have the disadvantage of being expensive and difficult to increase accuracy [4–8].

Due to their great accuracy and versatility, solutions based on laser sensors, LiDAR sensors, or cameras have recently become popular. However, in some settings, such as those that are outside or dusty, the system's accuracy will quickly deteriorate. At current time, the inertial sensor is a viable option because it is adaptable to a wide range of settings. However, because of the nature of the inertial sensor, errors accumulate during operation, lowering the system's accuracy over time [9–12].

In oder to overcome the problems, in this paper, we propose a navigation solution for a two-wheeled differential mobile robot. In the method, the inertial sensor and encoder data are used to estimate the robot's direction angle and location, which is then calculated using the Kalman filter algorithm. Magnetic sensors are added on top of this to remedy the accumulated inaccuracies. When the robot travels over magnetic reference points installed under the floor on virtual routes (known as *trajectories of the robot*), its angular and positional faults are corrected. The content is presented in Sect. 2 of the paper. Based on the established global trajectory with path-continuous control using the Lyapunov function [13–16]. Some theories related to these tests will be presented in Sect. 3. For the task of avoiding close obstacles in the local area, we applied the VFH+ method [26, 27] with a system of ultrasonic sensors, developed a control program that allows avoiding obstacles in the distance area from 0.3 m to 4 m. Obstacle avoidance results during navigation are also presented in Sect. 3 of the paper. Section 4 presents experimental measurement results showing the effectiveness of data aggregation from sensors for autonomous robot navigation, allowing to open up application possibilities with intelligent autonomous vehicles in real life.

2 Robot Positioning Using Sensors and Virtual Paths

2.1 Kinematic Model of Mobile Robot

In this work, the mobile robot is a two-wheeled moving vehicle with differential drive. Each wheel is connected to an independently controlled electric motor, and the speed of these two motors is used to control the vehicle's movement and orientation.

The dynamic analysis will be influenced by a number of parameters, including vehicle load, control circuitry, engine, and body construction, among others. A somewhat simple robot motion model is described in this paper, omitting certain relatively minor aspects. The following Fig. 1 depicts the robot's structure.

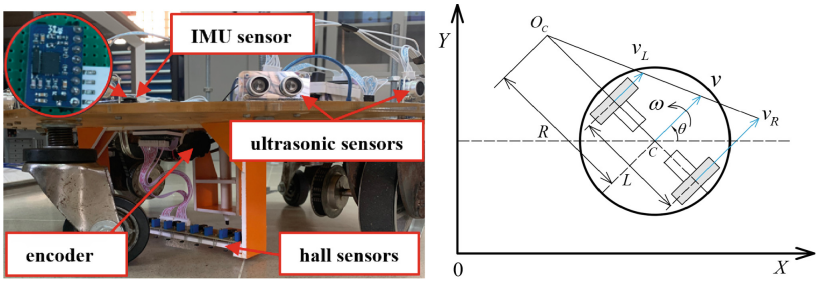


Fig. 1. (a) Mobile robot structure model; (b) Kinetic model of mobile robot

The OXY coordinate system associated with the active area plane of the mobile robot is chosen where the origin O is the initiating point of the mobile robot. The gravity center of the robot C is the point among the two-wheeled differential drive. The robot AGV will travel through an arc of the center O_c and the radius R. The rotation angle of the center OC will coincide with the center C of the robot when $v_r = v_l$.

Defining v, ω are the translational speed and angular velocity of the robot. Based on the kinetic model of the robot in Fig. 1, we achieve

$$v = \frac{v_r + v_l}{2}. \tag{1}$$

Defining x, y are the coordinates of the robot that create the angle between the vertical axis of the robot and the axis OX.

The robot's state is illustrated by the vector $X = \begin{bmatrix} x\ y\ \theta \end{bmatrix}^T$. At the different time of k and time $k-1$, we got the robot's states which are presented as $X_k = \begin{bmatrix} x_k\ y_k\ \theta_k \end{bmatrix}^T$ and $X_{k-1} = \begin{bmatrix} x_{k-1}\ y_{k-1}\ \theta_{k-1} \end{bmatrix}^T$. Where, X_k is determined in terms of X_{k-1} by the following calculation:

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} cos\theta_{k-1} & 0 \\ sin\theta_{k-1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{k-1}T \\ \omega_{k-1}T \end{bmatrix}, \tag{2}$$

where, T is the sampling time from the measuring signal.

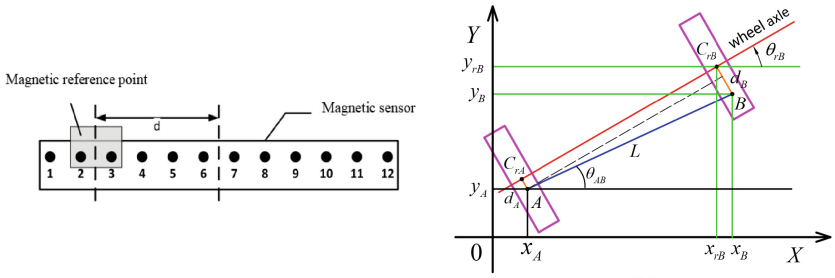
2.2 Position and Orientation of Mobile Robot Based on Reference Points

Tracking the trajectory of the robot is needed to be able to control it. In our paper, the model uses an inertial sensor (IMU) adn encoders to define either the direction angle or the coordinates of the robot. As a result, the trajectory is determined using the initial input values paired with sensor feedback values, and the cumulative inaccuracy will steadily increase as the robot moves.

When a result, a magnetic sensor is added to detect the waypoints with known coordinates so that the trajectory can be recalibrated as the Robot passes through them. The magnetic sensor is a bar made up of 16 hall sensors arranged in a row and separated

by a distance of $l = 10$ mm. As a result, the magnetic sensor's width is 160 mm. At the midway of the sensor bar, it will be perpendicular to the body's longitudinal axis.

The hall sensors traveling above the reference point will be engaged when the robot passes through the magnetic reference points on the floor. The deviation of the longitudinal axis of the body can be determined from the reference point using the position of the activated hall sensors, as shown in Fig. 2.



(a) (b)

Fig. 2. (a) The magnetic reference point and the magnetic sensor model; (b) The Position and the orientation of Robot at the reference point calculation.

We assume both reference points, $A(x_A, y_A)$ and $B(x_B, y_B)$ having eithe the distance L or known coordinates. The angle of segment AB considering axis OX is θ_{AB}. The robot will travel through A to B. At the moment, the robot travels through point A, the deviation d_A is detected by the magnetic sensor. When the robot moves to point B, the sensor can determine the deviation d_B. Based on Fig. 2(b), it is obvious to be able to calculate both the actual position and the angle of the robot at the point B.

Define δ as the angle of deviation of the line segment AB and the longitudinal axis of the body, we achieve:

$$\delta = \arcsin[(d_A - d_B)/L] \tag{3}$$

Hence, the actual direction angle of the robot AGV at the point B can be calculated as

$$\theta_{rB} = \theta_{rB} + \delta. \tag{4}$$

Based on that, the coordinates of the actual position of the AGV at point B is defined as

$$\begin{cases} x_{rB} = x_B + d_B \sin(\theta_{rB}) \\ y_{rB} = y_B + d_B \sin(\theta_{rB}) \end{cases} \tag{5}$$

Hence, the the robot AGV's state at point B can be adjusted based on the actual state as follows

$$X_{rB} = \begin{bmatrix} x_{rB} \\ y_{rB} \\ \theta_{rB} \end{bmatrix} = \begin{bmatrix} x_{rB} = x_B + d_B \sin(\theta_{rB}) \\ y_{rB} = y_B + d_B \sin(\theta_{rB}) \\ \theta_{AB} + \arcsin[(d_A - d_B)/L] \end{bmatrix} \tag{6}$$

2.3 Application Filter for Mobile Robot

Measured disturbances and system noise frequently alter the robot's signal during operation. The Kalman filter, which is widely used in the field of signal processing and control, is utilized in the article to identify the exact value of the robot's state [4–8].

Based on Eq. (2), the state model system is built and measured following

$$\begin{aligned} X_k &= AX_{k-1} + Bu_{k-1} + w_{k-1} \\ Z_{k-1} &= HX_{k-1} + v_{k-1} \end{aligned}, \tag{7}$$

where, X_k is defined as the state of the robot at the time k, and Z_{k-1} is the measuring state at the time $k-1$. The system noise and measurement noise are defined as w and v. The state transition matrix and the input control matrix are defined as A and B, respectively. H denotes the observation transfer matrix. The matrices are calculated in details as

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \cos\theta_{k-1} & 0 \\ \sin\theta_{k-1} & 0 \\ 0 & 1 \end{bmatrix}, u_{k-1} = \begin{bmatrix} v_{k-1}T \\ \omega_{k-1}T \end{bmatrix}, H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

The Kalman filter includes two stages, prediction and correction, that are provided as follows.

Stage 1: Based on the present state and the value of the incoming control signal, predicts the next state:

$$\begin{aligned} \hat{X}_k^- &= A\hat{X}_{k-1}^- + Bu_{k-1} \\ P_k^- &= AP_{k-1}A^T + Q \end{aligned} \tag{9}$$

Stage 2: Recalibrate the estimated value based on the projected value and the signal value measured:

$$\begin{aligned} K_k &= P_k^- H^T \left(HP_k^- H^T + R\right)^1 \\ \hat{X}_k &= \hat{X}_k^- + K_k\left(Z_k - H\hat{X}_k^-\right), \\ P_k &= (I_n - K_k H)P_k^- \end{aligned} \tag{10}$$

where Q *and* R denote the system noise correlation matrices. The measurement noise is defined by using the noise variance.

In our work with the robot control system, the measurement values as Z_k are the sum of collected values from the Encoder, IMU, and the time reference while the robot is operating.

3 Tracking the Trajectory and Obstacle Avoidance

3.1 Trajectory Tracking

The main goal is to control the mobile robot to track a certain trajectory. A different trajectory with a path with time constraints added to it, which makes the control target not only minimize the distance between the robot and the path, but also to ensure the

travel time. We define the actual robot state as: $X = \begin{bmatrix} x\ y\ \theta \end{bmatrix}^T$ and according to the pattern trajectory is: $X_r = \begin{bmatrix} x_r\ y_r\ \theta_r \end{bmatrix}^T$.

When the robot moves, the error will appear (Fig. 3):

$$e = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{11}$$

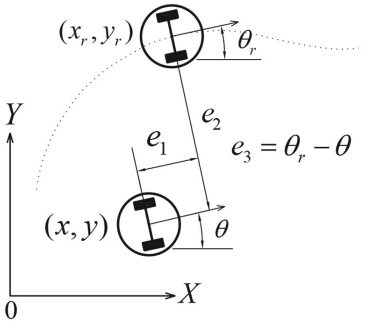


Fig. 3. The positional error between the robot's actual coordinates and the reference coordinates in the trajectory.

From the kinetic and derivative model (11), we get the error model as follows:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} \cos e_3 & 0 \\ \sin e_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{12}$$

where v_r, ω_r define the linear and angular velocities of the robot according to the trajectory.

The robot controller is built following

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_r \cos e_3 \\ \omega_r \end{bmatrix} + \begin{bmatrix} v_{fb} \\ \omega_{fb} \end{bmatrix}, \tag{13}$$

where v_{fb}, ω_{fb} is the feedback signal of the controller.

Substituting (13) into the error model (12), becomes:

$$\begin{aligned} \dot{e}_1 &= \omega_r e_2 - v_{fb} + \omega_{fb} e_2 \\ \dot{e}_2 &= -\omega_r e_1 + v_r \sin e_3 - \omega_{fb} e_1 \\ \dot{e}_3 &= -\omega_{fb} \end{aligned} \tag{14}$$

The control target is to bring the error of the error model to zero by choosing signals v_{fb} and ω_{fb} appropriately. Using the Lyapunov stability theorem for stability control

design purposes and if asymptotic stabilization is possible. It means that all trajectories finally will be converged to the reference trajectories. The most likely Lyapunov function candidate is calculated as the sum of three square errors:

$$V(e) = (e_1^2 + e_2^3)k_2/2 + (1/2)e_3^2 \tag{15}$$

It is also considered as the weighted sum of the squared error of distance and squared error of direction. The constant $k_2 > 0$ must be added because the units are different. The time derivative of V can be calculated as

$$\dot{V} = k_2 e_1 \dot{e}_1 + k_2 e_2 \dot{e}_2 + e_3 \dot{e}_3. \tag{16}$$

Substitute (14) into (16) to get:

$$\dot{V} = -k_2 e_1 v_{fb} + k_2 v_r e_3 \sin e_3 - e_3 \omega_{fb} \tag{17}$$

The purpose of the Lyapunov-based control design aims to achieve the derivative of the Lyapunov function negative by picking a appropriate control law. The velocity v_{fb} is linearly chosen to have the first term in Eq. (7) negative by squaring the negative term. The angular velocity is chosen to eliminate the second and third terms as well to square the negative term. The control laws include

$$\begin{aligned} v_{fb} &= k_1 e_1 \\ \omega_{fb} &= k_2 v_r (\sin e_3 / e_3) e_2 + k_3 e_3 \end{aligned} \tag{18}$$

Then $\dot{V}$ becomes:

$$\dot{V} = -k_1 k_2 e_x^2 - k_3 e_3^2 \tag{19}$$

It is clear that A is always negative for $k_1 > 0$, $k_2 > 0$, and $k_3 > 0$, satisfying the Lyapunov stability conditions. Then the control law (3) for trajectory tracking will be rewritten as follows:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_r \cos e_3 + k_1 e_1 \\ \omega_r + k_2 v_r (\sin e_3 / e_3) e_2 + k_3 e_3 \end{bmatrix} \tag{20}$$

3.2 Obstacle Avoidance Using VFH+ with Ultrasonic Sensors

Along the way, the robot must be able to detect and avoid unexpected obstacles. In these cases, a proximity sensor system is used. That is 12 ultrasonic distance sensors installed on the robot as shown in Fig. 4, allowing to detect obstacles in front and two sides of the robot.

The VFH+ method uses a histogram grid to map the environment around the robot. This map is continuously updated with distance to obstacle data obtained from the ultrasonic sensors mounted on the robot as shown in Fig. 4. The method will find the

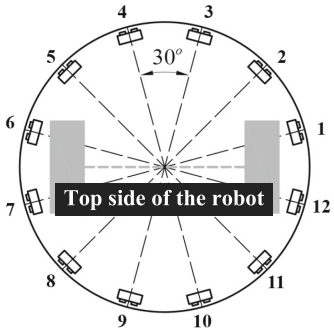


Fig. 4. Layout diagram of ultrasonic sensors.

optimal direction of movement when encountering obstacles, and appropriate velocity control for the robot (linear velocity, angular velocity).

The Histogram Grid

In this step, a two-dimensional Cartesian histogram grid (C) is generated containing the information transmitted from the ultrasonic sensors (the selected C has dimensions of 81 $\times$ 81 and a resolution of 0.1 m/cell). Each grid cell $C[i, j]$ contains a value representing the reliability of the existence of an obstacle at the coordinate position (i, j). The grid is filled using distances measured by ultrasonic sensors, each of which updates the value of only one cell at a time (Fig. 5).

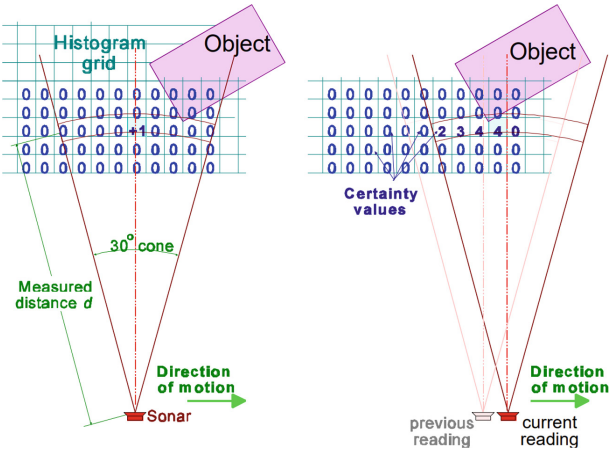


Fig. 5. Histogram grid [25].

The Primary Polar Histogram

In the second step, a window C $*$ has a fixed orientation, whose center is attached to the robot and moves along its motion (the selected C $*$ is 33 $\times$ 33). It is called the active window and it overlays the histogram grid C at the robot's current position. The content

of each active cell in the map grid is considered as an obstacle vector, its direction β being towards the center of the active region (robot center point-RCP)

$$\beta_{i,j} = \tan^{-1}\left(\frac{y_i - y_o}{x_i - x_0}\right) \tag{21}$$

The magnitude vector of the active cell $C[i, j]$ is provided by

$$m_{i,j} = c_{i,j}^2(a - bd_{i,j}^2), \tag{22}$$

where both a and b are positive constants; $c_{i,j}$ denotes the certainty value of an active cell (i, j); $d_{i,j}$ is the distance from each cell to the center of the active area (RCP); the magnitude of the obstacle vector at the cell (i, j) is denoted as $m_{i,j}$; (x_0, y_0) is the coordinate of the robot center; (x_i, y_i) is the coordinate of each cell (i, j).

The Primary Polar Histogram H^p is built based on the obstacle vectors. H^p contains an arbitrary angular resolution, denoted as α. Hence, $n = 360^o/\alpha$ is an integral number. Specifically, in our work, we choose $\alpha = 5^o$ and $n = 72$ angular sectors. A angular sector k corresponds to a discrete angle $k\alpha$.

For each obstacle cell, the enlargement angle $\gamma_{i,j}$ is defined by:

$$\gamma_{i,j} = \arcsin\frac{r}{d_{i,j}}, \tag{23}$$

where r is the radius of the robot's safe zone (Fig. 6).

For a angular sector k, the polar obstacle density is then defined as

$$H_k^p = \sum m_{i,j} h'_{i,j}, \tag{24}$$

with

$$h'_{i,j} = \begin{cases} 1 \text{ if } k\alpha \in [\beta_{i,j} - \gamma_{i,j}, \beta_{i,j} + \gamma_{i,j}] \\ 0 \qquad \text{elsewhere} \end{cases} \tag{25}$$

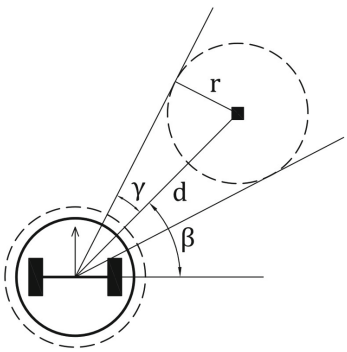


Fig. 6. Enlargement angle from robot to the obstacle.

Binary Polar Histogram

The Binary Polar Histogram H^b shows which directions are free for a robot that can change direction of motion instantly without colliding with obstacles. The following rules are used to update the binary polar histogram.

$$H^b_{k,i} = \begin{cases} 1 & \text{if } H^p_k > \tau_{high} \\ 0 & \text{if } H^p_k < \tau_{low} \\ H^b_{k,i-1} & elsewhere \end{cases} \tag{26}$$

where τ_{high} and τ_{low} are high thresholds and low thresholds are chosen to ensure that obstacles too far away from the robot do not affect the robot's movement, even though the obstacle is in front of the moving direction.

The Masked Polar Histogram

The original VFH technique ignores the robot's dynamics and kinematics. It implicitly presupposes, as shown in Fig. 7(a), that the robot can change its direction of motion at any time. Unless the robot can move in all directions, this assumption is not true for two-wheeled differential robots.

(a) (b)

Fig. 7. Flexible mobility of the robot: a) omnidirectional, b) limited multidirectional movement [26]

The VFH+ [27] approach uses a simple but more accurate approximation of most mobile robots' trajectory. As shown in Fig. 7(b), the robot's route is based on circular arcs (constant curvature curves) and straight lines.

In the example shown in Fig. 8, because of the robot dynamics, obstacle A blocks all directions to its left. Obstacle B, on the other hand, does not obstruct the directions to its right. The original VFH approach considers the directions to the left of obstacle A to be appropriate motion directions. The original VFH algorithm will instruct the robot to the left and collide with the obstacle A.

The positions of the right and left rotation centers are related to the current position of the robot which are determined by

$$\begin{matrix} \Delta x_r = R\sin(\theta - \frac{\pi}{2}) & \Delta x_l = R\cos(\theta + \frac{\pi}{2}) \\ \Delta y_r = R\cos(\theta - \frac{\pi}{2}) & \Delta y_l = R\sin(\theta + \frac{\pi}{2}) \end{matrix}. \tag{27}$$

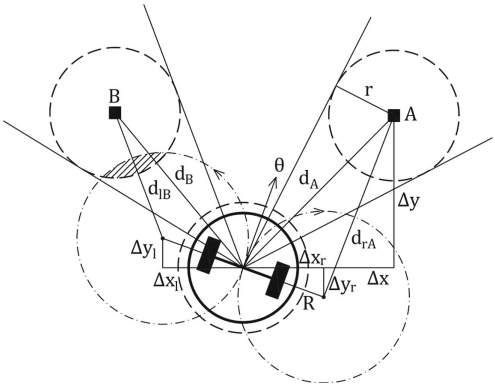


Fig. 8. An illustration of blocked directions

Distance from obstacle to robot's right and left rotation centers:

$$\begin{aligned} d_r &= \sqrt{(\Delta x - \Delta x_r)^2 + (\Delta y - \Delta y_r)^2} \\ d_l &= \sqrt{(\Delta x - \Delta x_l)^2 + (\Delta y - \Delta y_l)^2}. \end{aligned} \tag{28}$$

There is an obstacle blocking the robot's right direction if.

$$d_r < R + r \quad \text{[condition 1]} \tag{29}$$

Then, an obstacle will block the robot's right direction if

$$d_l < R + r \quad \text{[condition 2]} \tag{30}$$

The right limit angle φ_r and the left φ_l are calculated as follows:

- Let: $\varphi_r = \varphi_l = \theta + \pi$. Check the cells in the *active window* with c > threshold:
  - If β to the right of θ and the left of φ_r, check condition 1. If the condition is satisfied, set $\varphi_r = \beta$.
  - If β to the left of θ and the right of φ_l, check condition 2. If the condition is satisfied, set $\varphi_l = \beta$.

Based on φ_r, φ_l and binary polar histogram, a masked polar histogram can be built in Eq. (31) and illustrated in Fig. 9.

$$H_k^m = \begin{cases} 0 & \text{if } H_k^b = 0 \text{ and } k\alpha \in \{[\varphi_r, \theta], [\theta, \varphi_l]\} \\ 1 & otherwise \end{cases} \tag{31}$$

Selection of the Steering Direction

The VFH+ approach first determines a collection of probable candidate directions by finding all vacancies in the masked polar histogram. These candidate directions are then subjected to a cost function that takes into account more than simply the difference between the candidate and goal directions. The candidate motion k_n with the lowest cost is then selected as the new motion $\varphi_n = \alpha\, k_n$ direction.

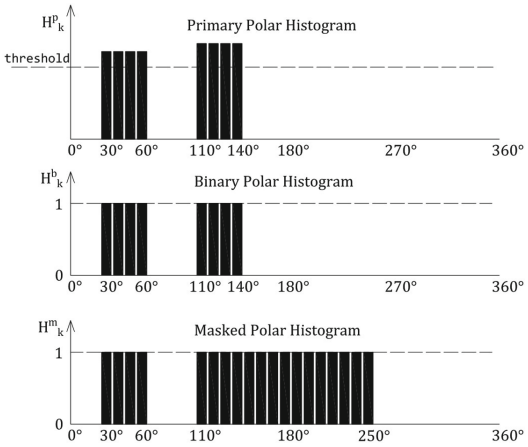


Fig. 9. a) Primary polar histogram, b) Binary polar histogram, c) Masked polar histogram.

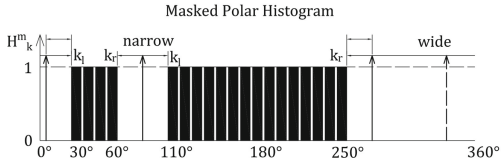


Fig. 10. Selecting the most optimal direction among candidate directions

In the beginning, the right and left borders k_r and k_r of all openings are considered. It is similar to the original VFH method, two kinds of openings are recognized and defined as *wide* and *narrow* ones. If the difference between its two borders is larger than s_{max} time α (in our system $s_{max} = 16$ sectors), then an opening is considered wide. Otherwise, it is considered narrow, as shown in Fig. 10.

The only one candidate direction can be chosen for narrow openings:

$$c_n = (k_r + k_l)/2 \tag{32}$$

For wide openings, 3 candidate directions are selected:

$$\begin{aligned} c_1 &= k_r + s_{\max}/2 \\ c_2 &= k_l - s_{\max}/2 \\ c_3 &= k_{\text{target}} \quad \text{if} \quad k_{\text{target}} \in [c_1, c_2] \end{aligned} \tag{33}$$

With k_{target} is target direction.

We substitute the selected directions into the cost function:

$$g(c) = \mu_1 \Delta(c, k_{\text{target}}) + \mu_2 \Delta(c, k_\theta) + \mu_3 \Delta(c, c^{-1}) \tag{34}$$

With k_θ is the current direction of the robot.

The function $g(c)$ with the lowest result will be the next optimal direction for the robot to move.

The higher the value of μ_1, the more goal-oriented the robot's behavior will be. The higher μ_2, the more the robot strives to follow an efficient course with the fewest possible changes in motion direction. The higher μ_3, the more the robot strives to follow the previously chosen path, and the smoother the trajectory becomes. To prioritize towards the goal, we choose as follows

$$\mu_1 > \mu_2 + \mu_3 \tag{35}$$

4 Simulation and Experimental Results

4.1 Track the Trajectory When There Are no Obstacles

The robot in Fig. 1 is used for the experimentation. With a distance between two wheels $L = 0.47$ m, wheel diameter $d = 0.15$ m, maximum translational speed $v_{max} = 1$ m/s. The robot will be controlled to track the reference trajectory according to the control law based on the Lyapunov function as shown in Fig. 11(b). The robot will start from the origin O (0,0) traveling through the magnet points glued on the floor: A, B, C, D, E, F, G, H, and destination I (4.5, 3). The robot will accelerate at point O with an acceleration of 0.15 m/s^2, when it reaches a certain speed as 0.2 m/s, it can move stably on the trajectory and will begin to decelerate when 0.2 m away from the destination with an acceleration of -0.15 m/s^2. The robot will come to a halt when reaching the destination I.

The robot will update the status values of the parameters (including position parameters from the Encoder and orientation angle from the IMU) and a controller based on these parameters to control the robot to track the reference trajectory as it moves through the correction points from A, B, C, D, E, F, G, H.

Figure 11 shows the results of gathering the robot's trajectories in three scenarios, together with a reference trajectory. The path calculated only by odometry (Encoder), the path estimated using a Kalman filter and an IMU sensor (Encoder + IMU), and the predicted path when incorporating additional Hall sensors (Encoder + IMU + Hall) are the three scenarios.

The deviations in the X and Y directions and the position of the robot when following the reference trajectory are shown in Figs. 12, 13 and 14. When combining a fusion of encoder sensors, IMU, and magnetic hall sensors, the trajectory is visually closer to the true path than when using only odometry.

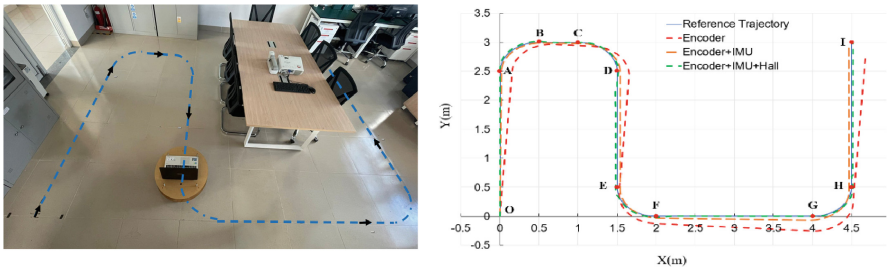
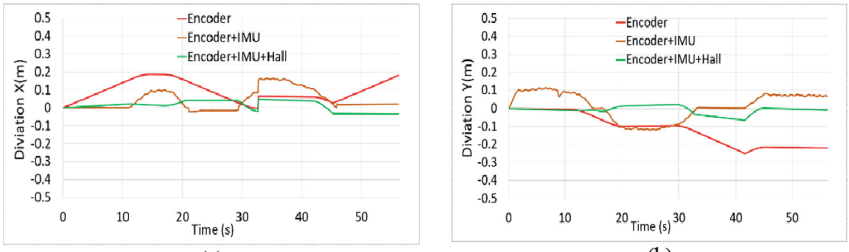
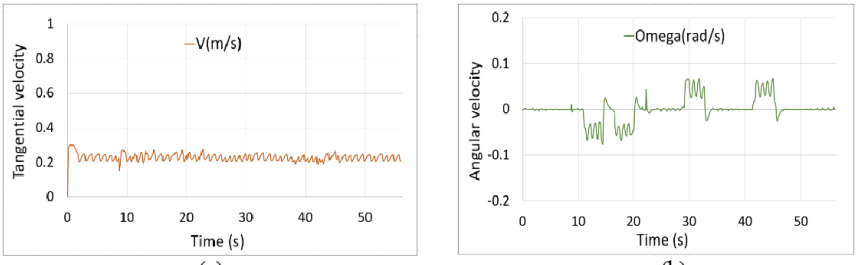


Fig. 11. Experimental images & Tracing reference trajectory



(a) (b)

Fig. 12. When tracking the reference trajectory, the deviation between the robot's positions is calculated: (a) in X(m) direction; (b) in Y(m) direction



(a) (b)

Fig. 13. Linear and angular velocities of the robot in case of using three sensors Encoder + IMU + Hall.

4.2 Track the Trajectory and Avoid Obstacles

The robot will use all three sensors Encoder + IMU + Hall and be controlled to track the trajectory set up in the laboratory in Fig. 11(b), on the way the robot will encounter unexpected obstacles. The robot will use the VFH+ method to avoid obstacles as shown in Fig. 15.

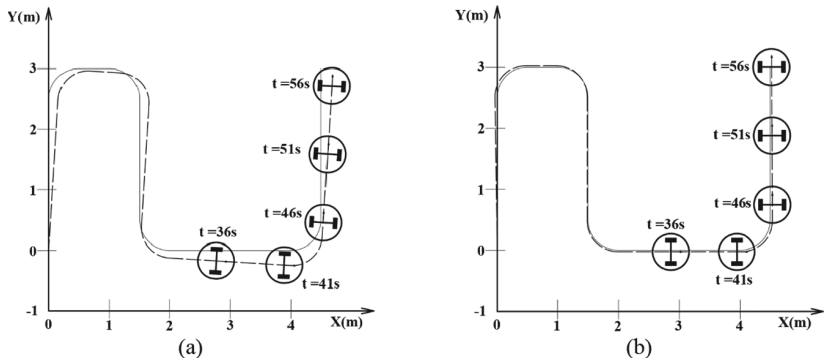


Fig. 14. The error of position over time of robot with reference trajectory in case: (a) using Encoder and (b) using all three sensors Encoder + IMU + Hall.

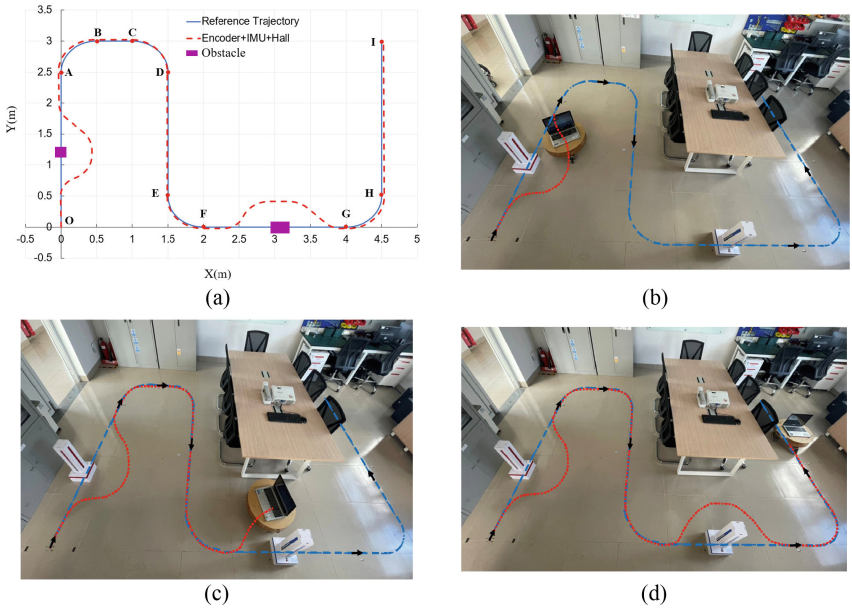


Fig. 15. Trajectory path and scene image of the robot.

5 Conclusions

The article presents the research process of guiding a mobile robot to operate autonomously through the stages of positioning, tracing the trajectory, and avoiding obstacles appearing unexpectedly on the way. Despite the existence of measurement noise in sensors such as the Encoder and IMU, the simulation results show that the robot follows the reference trajectory and clings to it successfully.

The basis of the reference trajectory is established, with a control law based on the Lyapunov function applied. Although the Lyapunov function method may require

a longer distance than the previous method, it will have a continuous trajectory, satisfying the angle of the robot at the destination. The software for controlling robotic obstacle avoidance by potential field method has also been developed with good results in local space. These research results can be developed for the research of navigation of autonomous vehicles in outdoor environments.

References

1. Meiling, W., Zhen, W., Yi, Y., Mengyin, F.: Model predictive control for UGV trajectory tracking based on dynamic model. In: Proceedings of the IEEE International Conference on Information and Automation Ningbo, pp. 1676–1681 (2016)
2. Thanh, V.C., Quan, N.N.A., Le Thang Dong, T., Hoang, T.T., Nguyen, M.T.: Fusion of inertial and magnetic sensors for autonomous vehicle navigation and freight in distinctive environment. In: Nguyen, D.C., Vu, N.P., Long, B.T., Puta, H., Sattler, K.-U. (eds.) ICERA 2021. LNNS, vol. 366, pp. 431–439. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-92574-1_45
3. Kim, D.H., Kim, S.B.: Path following control of automated guide vehicle using camera sensor. In: Zelinka, I., Brandstetter, P., Trong Dao, T., Hoang Duy, Vo., Kim, S.B. (eds.) AETA 2018. LNEE, vol. 554, pp. 932–938. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-14907-9_90
4. Hoang, T.T., Duong, P.M., Van, N.T.T., Viet, D.A., Vinh, T.Q.: Development of an EKF-based localization algorithm using compass sensor and LRF. In: Proceedings of IEEE 12th International Conference on Control, Automation, Robotics & Vision, ICARCV, pp. 341–346 (2013)
5. Dinh, T.H., Phung, M.D., Tran, T.H., Tran, Q.V.: Localization of a unicycle-like mobile robot using LRF and omni-directional camera. In: IEEE International Conference on Control System, Computing and Engineering, pp. 477–482 (2012)
6. Hoang, T.T., Duong, P.M., Van, N.T.T., Viet, D.A., Vinh, T.Q.: Multi-sensor perceptual system for mobile robot and sensor fusion - based localization. In: IEEE 1st International Conference on Control, Automation and Information Sciences (ICCAIS-2012), pp. 259–264 (2012)
7. Hoang, T.T., Hiep, D.T., Duong, P.M., Van, N.T.T., Duong, B.G., Vinh, T.Q.: Proposal of algorithms for navigation and obstacles avoidance of autonomous mobile robot. In: Proceedings of IEEE 8th Conference on Industrial Electronics and Applications (ICIEA-2013), pp. 1308–1313 (2013)
8. Thanh, V.C., Dong, T.L.T., Quan, N.N.A., Hoang, T.T.: Autonomous vehicle navigation using inertial sensors and virtual path. In: National Conference "High-Tech Application in Practice in 2021" (2021)
9. Yousuf, S., Kadri, M.B.: Sensor fusion of INS, odometer and GPS for robot localization. In: IEEE Conference on Systems, Process and Control (ICSPC 2016), pp. 118–123 (2016)
10. Zhang, M., Li, K., Hu, B., Meng, C.: Comparison of kalman filters for inertial integrated navigation. Sensors **19**, 1426 (2019)
11. Kumar, S., Suganthi, M.: Design of accurate navigation system by integrating INS and GPS using extended kalman filter. Int. J. Eng. Res. Technol. (IJERT) **4**(5), 803–808 (2015)
12. Anbu, N.A., Jayaprasanth, D.: Integration of inertial navigation system with global positioning system using extended kalman filter. In: IEEE Second International Conference on Smart Systems and Inventive Technology (ICSSIT 2019), pp. 789–794 (2019)
13. Klancar, G., Matko, D., Blazic, S.: Mobile robot control on a reference path. In: Proceedings of the 13th Mediterranean Conference on Control and Automation, Limassol, Cyprus, 27–29 June 2005, pp. 1343–1348 (2005)

14. Widyotriatmo, A., Hong, K.S., Prayudhi, L.H.: Robust stabilization of a wheeled vehicle: hybrid feedback control design and experimental validation. J. Mech. Sci. Technol. **24**(2), 513–520 (2010)
15. Tran, T.H., Phung, M.D., Van Nguyen, T.T., Tran, Q.V.: Stabilization control of the differential mobile robot using lyapunov function and extended kalaman filter. J. Sci. Technol. **50**(4), 441–452 (2012)
16. Aicardi, M., Casalino, G., Bicchi, A., Balestrino, A.: Closed loop steering of unicycle-like vehicles via Lyapunov techniques. IEEE Robot. Autom. Mag. **2**(1), 27–35 (1995)
17. Ryck, D.M., Mark, V., Debrouwere, F.: Automated guided vehicle systems, state-of-the-art control algorithms and techniques. J. Manuf. Syst. **54**, 152–173 (2020)
18. Secchi, H., Carelli, R., Mut, V.: An experience on stable control of mobile robots. J. Latin Am. Appl. Res. **33**(4), 379–385 (2003)
19. Fei, L.: Two-wheel Driven AGV Path Planning and Motion Control. Hebei University of Science and Technology (2016)
20. Son, T.A., et al.: Research and manufacture of automated guided vehicle for the service of storehouse. Sci. Technol. Dev. J. Eng. Technol. **1**(1), 5–12 (2018)
21. Nascimento, T.P., Dórea, C.E.T., Gonçalves, L.M.G.: Nonholonomic mobile robots' trajectory tracking model predictive control: a survey. Robotica **36**, 676–696 (2018)
22. Hasan, H.S., Abidin, M.S.Z., Mahmud, M.S.A., Mohd, F.M.S.: Automated guided vehicle routing: Static, dynamic and free range. J. Int. Eng. Adv. Technol. **8**(5C), 1–7 (2019)
23. Kim, S., Jin, H., Seo, M., Har, D.: Optimal path planning of automated guided vehicle using dijkstra algorithm under dynamic conditions. In: 7th International Conference on Robot Intelligence Technology and Applications (RiTA). IEEE (2019)
24. Moshayedi, A.J., Roy, A.S., Liao, L.: PID tuning method on AGV (automated guided vehicle) industrial robot. J. Simul. Anal. Novel Technol. Mech. Eng. **12**(4), 53–66 (2020)
25. Borenstein, J., Koren, Y.: The vector field histogram - fast obstacle avoidance for mobile robots. IEEE J. Robot. Autom. **7**(3), 278–288 (1991)
26. Ulrich, I., Borenstein, J.: VFH+: reliable obstacle avoidance for fast mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, pp. 1572–1577 (1998)
27. Diaz, D., Marın, L.: VFH+D: an improvement on the VFH+ algorithm for dynamic obstacle avoidance and local planning. IFAC PapersOnLine **53**(2), 9590–9595 (2020)