IJRCS

International Journal of Robotics and Control Systems
Vol. 6, No. 1, 2026, pp. 324-341
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

ASCEE

Dynamic Evasion Point System for Mobile Robots

Salvador Santos-Romero [a,1], Virgilio López-Morales [a,2,*], Julio C. Ramos-Fernández [b,3], Manuel A. Ojeda-Misses [a,4]

[a] Information Technologies and Systems Research Center, Universidad Autónoma del Estado de Hidalgo, Carr. Tulancingo-Pachuca Km. 4.5, Pachuca de Soto, 42184, HGO., México
[b] LANAVEX, Universidad Politécnica de Pachuca, Carr. Pachuca-Cd. Sahagún Km. 20, Rancho Luna, Ex-Hacienda de Sta. Bárbara, Zempoala, 43830, HGO., México
[1] sa513025@uaeh.edu.mx; [2] virgilio@uaeh.edu.mx; [3] jramos@upp.edu.mx; [4] manuel_ojeda@uaeh.edu.mx
[*] Corresponding Author

ARTICLE INFO

Article History
Received September 23, 2025
Revised December 09, 2025
Accepted January 29, 2026

Keywords
Multi-Obstacle Avoidance;
Normalized Cost Function;
Dynamic Evasion Points;
Pose Tracking Control;
Differential Drive Robot;
2D LiDAR

ABSTRACT

In this paper, a novel local path planner based on a complete Obstacle Avoidance Algorithm (OAA) and a Proportional-Derivative (PD) controller is developed for Mobile Robots (MR). The main contribution of this work is a local planner that generates global evasion points instead of just angles, and uses a normalized cost function that weights between the total Euclidean route distance and the orientation turning cost, separating the evasion algorithm from the pose tracking controller. "Individual obstacles detected by the LiDAR (Light Detection and Ranging sensor) are classified by measuring the distance between their corners. The OAA determines whether the mobile robot can drive through them or considers them as a single obstacle. A target point (an evasion point or the final goal) is selected by minimizing a normalized cost function. Various real experiments were carried out comparing the VPH and our method to illustrate the advantages and efficiency in both scenarios, achieving successful evasion in complex environments, such as 'U' structures, and maintaining stable CPU usage. Our method uses a PD controller to drive both wheels and allows the mobile robot to reach its destination point in a smooth and efficient manner.

1. Introduction

For Mobile Robots operating in well-structured environments, obstacle avoidance algorithms can be relatively straightforwardly applied. However, in dynamic scenarios such as precision agriculture, greenhouses, outdoor autonomous navigation, etc., robust Real-Time Obstacle Avoidance (R-TOA) algorithms are essential for solving their random and involved environments. Furthermore, various solutions already exist, including complex and expensive computers and signal processors to implement autonomous navigation. Computers and signal processors have a high cost for autonomous navigation, due to the sophisticated hardware required for perception, localization, and path planning, which processes vast amounts of sensor data in real-time. Therefore, the aim of this paper is to achieve a solid balance between a cost-effective and robust autonomous navigation system.

Global path planning requires prior knowledge of the environment, specifically a map that indicates the position of obstacles. This may generate additional computational costs and poses a chal-

lenge in dynamic environments. An alternative is local path planning, where one function drives the robot toward a final goal or waypoints, while another algorithm detects obstacles and reorients the robot to avoid them. Therefore, in [8]–[11], [62] 2D LiDAR sensors are used to detect obstacles.

Several studies have addressed different strategies for autonomous navigation and obstacle avoidance in mobile robots, using light-based ranging (LiDAR) sensors and path planning algorithms.

Real-time local path planning, based on 2D LiDAR sensors, is a pillar in mobile robotics [1], [8], [10], [62]. These methods process the sensor data set to identify obstacles and generate immediate evasion commands. However, it is common that many purely reactive approaches, although fast, can result in suboptimal trajectories or jerky movements, as observed in [5], [49]. This occurs because the algorithm focuses on the temporary deviation from the obstacle, without a clear strategy for optimizing the remaining route.

For tracking these trajectories, classic PID or PD controllers are widely used due to their simplicity and proven effectiveness [2], [4] or controllers based on fuzzy systems as in [6]–[9], [19]–[21]. However, a controller's performance depends on the quality and stability of the target points it receives. Therefore, it is essential to develop a method that considers the generation of a smooth trajectory.

A widely used approach in local path planning is the VPH (Vector Polar Histogram) family of algorithms [11], [60], which share similarities with our method. These algorithms analyze the LiDAR data to create a polar histogram that identifies "valleys" (free passageways). Then, they select the optimal direction within the "valley" based on a cost function that compares the target direction and the current orientation [60]. However, the VPH and VPH+ cost functions are not usually normalized and primarily optimize the next steering angle.

The main weakness of these methods is that selecting a better angle does not guarantee a better route. The original VPH cost function, for example, is very sensitive to weights and does not consider the total Euclidean path length. This can lead the robot to choose a passageway that, although angularly optimal at that instant, may result in a much longer overall trajectory.

Additionally, "free segments" or "turning point" strategies [34] are also highlighted, which can be effective when tracing routes through an environment with obstacles, but can generate abrupt orientation changes.

In this paper, we propose a local path-planning and obstacle-avoidance algorithm for a differential mobile robot. Using 2D LiDAR measurements, the algorithm generates global coordinates in real time and selects the one that minimizes the cost of both remaining distance to the goal and the change steering effort. The computation is repeated in a loop and, consequently, the mobile robot effectively traces a path around the obstacle geometry. Unlike static-map or multi-sensor methods (which increase hardware and computational cost), our approach uses a PD controller. Experimental results show feasibility in real-world settings and advantages in simplicity and computation speed over full mapping techniques. Experimental results show feasibility in real-world settings and advantages in simplicity and computation speed over full mapping techniques.
The main contributions of this work are the following:
1) An evasion algorithm optimizing the total Euclidean distance of the trajectory (not just the angle or the safety distance) using a normalized cost function, allowing for an intuitive adjustment between smoothness and a shortest path.
2) Generation of global target points (intermediate points or final point), which separates the evasion task from the control task and then allows the general algorithm to be compatible with any points tracking controller (PD, difuso, etc.).
3) A method considering the displacement of the LiDAR sensor with respect to the robot's coordinate axis, allowing a flexible application in various hardware configurations.

The paper is organized as follows. Section 2 describes the experimental platform and develops the differential robot's kinematic model, including the PD controller design for navigation and reorientation. Section 3 describes the 6 steps that constitute the proposed algorithm. Section 4 presents the experimental results, divided into the validation of the proposed algorithm in a complex environment and a comparison with VPH; and Section 5 summarizes conclusions and future work.

2. Method

In the following, mobile robot's characteristics are presented, followed by its kinematic model and then a PD controller for the robot navigation is proposed.

In this paper, the navigation to a point is the core for the subsequent development of a dynamic path planning system for obstacle avoidance. The implementation of an odometry method is described to calculate the robot's position and orientation at each instant, crucial for real-time control.

2.1. Description of the Experimental Platform

A square differential mobile robot (DMR) with 80 cm sides was built and used to carried out all the tests. The DMR is equipped with incremental encoders to measure the position and angular velocity of both the left and right wheels. This robot incorporates an RPLIDAR-A1 sensor mounted on the front, 35 cm from the center of the drive wheel axle, which is considered the origin of the robot's local coordinate system (cf. Fig. 1).

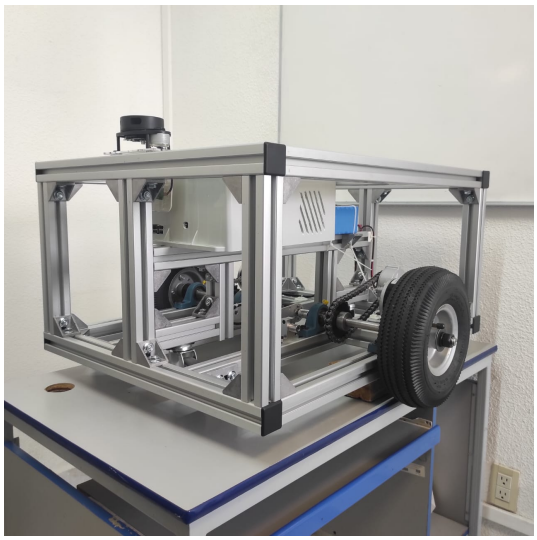

Fig. 1. Differential-drive robot with a front-mounted LiDAR sensor.

The SLAMTEC RPLIDAR-A1 is a 2D LiDAR sensor with a 360° scanning angle, a range of 12 meters, and a sampling rate of 2000 Hz. This sensor communicates via a serial port at a transmission rate of 115200 bps with a Raspberry Pi 5, which has a quad-core ARM Cortex-A76 processor running at 2.4 GHz and 8 GB of RAM. The Raspberry Pi 5, in turn, communicates serially with an ESP32 microcontroller. The ESP32 is responsible for acquiring data from the encoders, calculating odometry variables, and controlling the speed of the differential robot's motors. This device receives wheel speed commands from the Raspberry Pi 5 and sends odometry data in real time via serial communication.

2.2. Kinematic Model of the Robot

Our differential mobile robot has two individual drive wheels mounted on a single axle at the rear of the chassis and two caster wheels (one at the front and one at the rear) to balance the robot

during its movement. The two independent drive wheels, driven by two actuators, allow the robot to move linearly and angularly.

In general, the configuration of a rover moving on a plane is described by its generalized coordinates (cf. Eq. (1)), where the configuration space C describes all possible positions and orientations that the system can adopt. R^2 represents the plane where the robot or rover moves, defined by the coordinates (x,y). S represents the orientation space, indicating that θ is an angle in the range of $[0,2\pi)$ (cf. [18]).

$$q = (x,y,\theta) \in C : C \subset \mathbb{R}^2 \times S. \tag{1}$$

A differential robot is a non-holonomic system since its motion is restricted such that it cannot move laterally instantaneously.

For a differential robot configuration (cf. Fig. 2), the angular velocity ω can be defined by Eq. (2) where ω is formed from the Instantaneous Center of Curvature (CCI) and the horizontal axis. When V_R and V_L are equal, the angular velocity $\omega = 0$, i.e., the robot has a straight line displacement. Where In consequence:

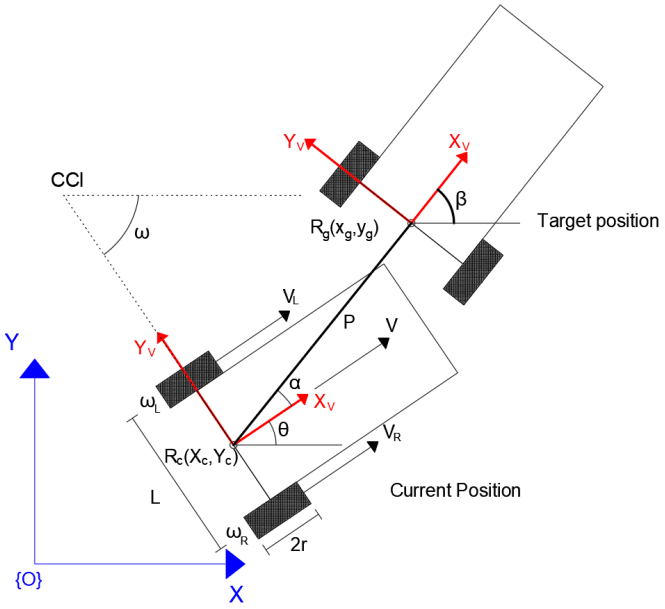

Fig. 2. Configuration of a differential robot

L : Distance between the two wheels (m).	r : Radius of the wheel (m)
ω : Angular velocity (rad/s).	ω_L : Angular velocity at the left wheel (rad/s).
ω_R : Angular velocity at the right wheel (rad/s).	V : Linear velocity (m/s).
V_R : Linear velocity at the right wheel (m/s).	V_L : Linear velocity at the left wheel (m/s).
θ : Orientation of the robot (rad).	CCI : Instantaneous Center of Curvature.

$$\omega = \frac{r(\omega_R - \omega_L)}{L}. \tag{2}$$

The linear velocity of the robot depends on the angular velocity and the radius of each of the wheels, as shown by Eq. (3).

$$V = \frac{r(\omega_R + \omega_L)}{2}. \tag{3}$$

The mobile robot's kinematic model, is described by the nonlinear equations (4).

$$\dot{x} = \frac{r(\omega_R - \omega_L)}{2}\cos\theta; \quad \dot{y} = \frac{r(\omega_R - \omega_L)}{2}\sin\theta; \quad \dot{\theta} = \frac{r(\omega_R - \omega_L)}{L}, \tag{4}$$

Where the rate of change of $\dot{x}$ is on the displacement in x, of $\dot{y}$ is on the displacement in y and of $\dot{\theta}$ is on the angular displacement in θ are represented respectively.

2.3. Design of the Proportional-Derivative Controller

Since motors' control is mandatory to move the robot from one point to another, a computationally simple Proportional-Derivative (PD) controller was chosen to perform position and orientation control tasks. Therefore, the real-time execution required for local trajectory planning is ideal for platforms with limited computational resources and provides a solid foundation for algorithm validation, a goal of this work.

2.3.1. PD Position Controller

The PD position controller aims to minimize the position error between the current position and the target, while simultaneously correcting the orientation error. This is achieved through a constant linear advancement combined with an orientation adjustment, allowing the robot to move directly toward the target. The distance from the robot's current position to the target is calculated as follows:

$$P = \sqrt{(X_c - X_g)^2 + (Y_c - Y_g)^2}, \tag{5}$$

Where X_c and Y_c represent the current coordinates of the robot in the global reference frame, and X_g and Y_g denote the coordinates of the target point. Note that while the PD controller can be optimally tuned for a linear system, in our case the differential robot is a nonlinear system, which presents a slightly more complicated challenge.

The orientation error, defined as $\alpha = \beta - \theta$, where β is the orientation angle of the robot's distance to the target and θ is the robot's current orientation angle.

The distance of the position error P and the orientation error α, defined with respect to a target point from the current robot's position is shown in Fig. 2. These quantities are calculated from the location of the robot's local coordinate axis, which is located at the center of the drive wheel axle.

To minimize orientation and position errors, a condition is used in which the robot maintains a constant linear velocity V_{com} and a control variable ω_{com} individually modifies the speed of the wheels. Consequently, the resulting control velocities V_R and V_L are obtained, as indicated in (6).

$$V_R = V_{com} - \omega_{com}, \quad V_L = V_{com} + \omega_{com}, \tag{6}$$

Where ω_{com} is obtained from the PD controller, as given in (7).

$$\omega_{com} = k_P e + k_D \dot{e}, \tag{7}$$

Where k_P and k_D are the proportional and derivative gains, respectively. These values were adjusted during experimental tests via exhaustive trial-error tests, to obtain a stable and fast response. While obtaining the optimal gains for nonlinear systems, as this one, is a complex challenge, the experimental adjustment proved sufficient to meet our objectives.
Since the orientation error is $e = \alpha$, therefore:

$$e = \beta - \theta, \tag{8}$$

Where β is defined as the angle between the positive horizontal axis and the displacement vector from the robot's current position (X_c, Y_c) to the goal position (X_g, Y_g):

$$\beta = \tan^{-1}\left(\frac{Y_g - Y_c}{X_g - X_c}\right). \tag{9}$$

2.3.2. PD Orientation Controller

Once the position error is practically zero, the orientation controller will steer the robot towards the target orientation. Similar to Eq. (6), Eq. (10) is obtained as follows:

$$V_R = V_{com}(0) - \omega_{com}, \quad V_L = V_{com}(0) + \omega_{com}. \tag{10}$$

When the robot reaches the target point, the orientation error can be defined by Eq. (8), where θ is the current orientation (i.e., the orientation with which it reached the target point) and β is the desired target orientation. Since β is constant, its derivative is equal to zero, and therefore ω_{com} can be expressed by Eq. (11).

$$\omega_{com} = k_P(\beta_g - \theta) - k_D\,\omega. \tag{11}$$

2.4. Odometry

The robot measures its location at each instant and its position estimation is performed using odometry, specifically, by the Dead Reckoning method, which is based on the previous position and orientation at the time $t = t_k$. Applying the Euler approximation to the differential robot's kinematic model, the following Eq. (12) is obtained:

$$\begin{aligned} x_{k+1} &= x_k + \frac{\pi D}{2}(\Delta L + \Delta R)\cos(\theta_k), \\ y_{k+1} &= y_k + \frac{\pi D}{2}(\Delta L + \Delta R)\sin(\theta_k), \\ \theta_{k+1} &= \theta_k + \frac{\pi D}{L}(\Delta R - \Delta L), \end{aligned} \tag{12}$$

Where:

- x_k, y_k, θ_k represent the position and orientation of the robot at time step k, respectively.
- D is the wheel diameter.
- L is the distance between the wheels (robot axle).
- ΔL and ΔR are the changes at the left and right wheel encoders, respectively.

The time interval between samples is defined as follows:

$$\Delta T = T_{k+1} - T_k. \tag{13}$$

The odometry calculation is carried out in the ESP32 microcontroller from an initial position (x_k, y_k) and an initial orientation θ_k. From this reference, the position and orientation are updated at each time interval ΔT based on the changes detected in the encoders. This method can introduce errors, such as those caused by wheel slippage, which accumulate as the distance traveled increases. However, it remains effective at low speeds or in short trips.

3. Local Path Planning

To achieve reactive navigation in environments with multiple obstacles, a local path planning algorithm was developed that uses real-time LiDAR sensor readings. The goal is to generate reliable intermediate points to avoid obstacles within the detection radius and select the most suitable point based on a cost function that considers both distance and change in orientation. The algorithm follows a logical sequence to transform the sensor readings into an immediate (intermediate or final) target point. The algorithm runs cyclically and consists of the following steps:

- Detection: LiDAR data points that indicate a possible collision within the detection radius are identified.

- Clustering: These collision points are grouped together to distinguish individual obstacles.
- Segment validation: The free segments (corridors) among the detected obstacles are detected and checked if they are wide enough for the robot to pass through them.
- Global evasion points generation: Global coordinates are calculated for evasion points within safe corridors.
- Target visibility check: Determines whether the final target (goal) is directly accessible without colliding.
- Target point selection: The optimal point to follow (whether an evasion point or the final point) is chosen using a normalized cost function that takes into account the total distance of the route and the necessary change of direction.

Each of these steps is detailed below.

3.1. Detection (Collision Vectors)

The first step processes the distance vector $D = \{d_1, d_2, ..., d_{359}\}$, obtained from the LiDAR sensor. The indices corresponding to the readings that represent an obstacle within a defined detection radius d_{max} are identified. The set of collision indices, $V_{collision}$, is defined as in Eq. (14).

$$V_{collision} = \{i \in \{1, ..., 359\} \mid 1 < d_i \leq d_{max}\}, \tag{14}$$

Where d_i is the distance measured at the angular index i. If $V_{collision}$ is empty, it means that no nearby obstacles are detected and the robot proceeds directly to the final target point.

3.2. Clustering

If $V_{collision}$ is not empty, its elements are sorted in ascending order. Then, a clustering, based on angular proximity, is performed to differentiate individual obstacles. A maximum angular difference threshold, Δ_{max}, is then defined. The set $V_{collision}$ is divided into a set of subsets $O = \{O_1, O_2, ..., O_n\}$, where each O_j represents an obstacle. A new obstacle O_{j+1} is created if the index $v_m \in V_{collision}$, together with the previous index v_{m-1}, is greater than the threshold assigned by Eq. (15).

$$v_m - v_{m-1} > \Delta_{max}. \tag{15}$$

At the end of this step, a list of detected obstacles is obtained.

3.3. Segment Validation

This step aims to obtain those corridors (segments) that are wide enough for the robot to pass safely.

The edges of each obstacle O_j are identified and the initial index, to the right of the obstacle, is named as $i_{R_j} = \min(O_j)$, and the final index, to the left of the obstacle, is $i_{L_j} = \max(O_j)$. The straight line segments S_j connecting the final edge of one obstacle O_j to the initial edge of the next obstacle O_{j+1} (including the final segment between O_n and O_1) are calculated (cf. Fig. 3). The length of each segment is calculated using the law of cosines, as in Eq. (16).

$$S_j = \sqrt{d_{i_{L_j}}^2 + d_{i_{R_{j+1}}}^2 - 2 d_{i_{L_j}} d_{i_{R_{j+1}}} \cos(\gamma_j)}, \tag{16}$$

Where d_i is the distance measured at index i, and γ_j is the absolute angle between rays i_{L_j} and $i_{R_{j+1}}$.

A segment is considered navigable only if its length S_j is greater than or equal to the minimum width required by the robot as indicated by Eq. (17).

$$C_{min} = L + 2\varepsilon. \tag{17}$$

The result is a set of index pairs as in (18), where each pair represents the edges of a safe segment (corridor).

$$S_{valids} = \{(i_{L_j}, i_{R_{j+1}}) \mid S_j \geq C_{min}\}. \tag{18}$$

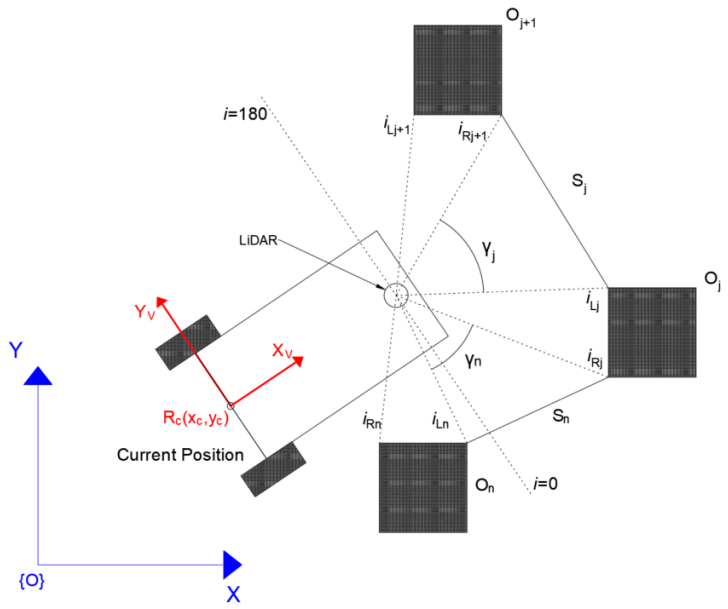

Fig. 3. Location of lateral indices and detection of segments between obstacles

3.4. Global Evasion Points Generation

For each valid segment $(i_L, i_R) \in S_{valids}$, two avoidance points are calculated in global coordinates (P_L, P_R). These points are located at a lateral distance $d_{IntPoint} = L/2 + \varepsilon$ from the edges of the obstacle. The position of each point (P_x, P_y) is obtained through a coordinate transformation that considers the current position of the robot (x_c, y_c, θ_c), the distance from the LiDAR to the robot axis d_{Lidar}, the measured distance d_i, and the angle α_i measured with respect to the horizontal. The distance to the intermediate point h_i and the lateral angle β_i are obtained through Eqs. (19) and (20) respectively (cf. Fig. 4).

$$h_i = \sqrt{d_i^2 + d_{IntPoint}^2}, \tag{19}$$

$$\beta_i = \arctan 2(d_{IntPoint}, d_i). \tag{20}$$

The global coordinates are calculated using Eq.s (21)-(22).

$$P_{L_x} = x_c + d_{Lidar}\cos(\theta_c) + h_{i_L} C_{\theta+\alpha+\beta}, \quad P_{L_y} = y_c + d_{Lidar}\sin(\theta_c) + h_{i_L} S_{\theta_c+\alpha_L+\beta_L}, \tag{21}$$

$$P_{R_x} = x_c + d_{Lidar}\cos(\theta_c) + h_{i_R} C_{\theta_c+\alpha_R-\beta_R}, \quad P_{R_y} = y_c + d_{Lidar}\sin(\theta_c) + h_{i_R} S_{\theta_c+\alpha_R-\beta_R}. \tag{22}$$

Where $C_{\theta_c+\alpha_L+\beta_L} = \cos(\theta_c + \alpha_{i_L} + \beta_{i_L})$, $S_{\theta_c+\alpha_L+\beta_L} = \sin(\theta_c + \alpha_{i_L} + \beta_{i_L})$, $C_{\theta_c+\alpha_R-\beta_R} = \cos(\theta_c + \alpha_{i_R} - \beta_{i_R})$ and $S_{\theta_c+\alpha_R-\beta_R} = \sin(\theta_c + \alpha_{i_R} - \beta_{i_R})$.
The result is a set of evasion point pairs shown by Eq. (23).

$$P_{evasion} = \{(P_{L_j}, P_{R_j}) \mid \forall (i_{L_j}, i_{R_j}) \in S_{valids}\} \tag{23}$$

3.5. Target visibility check

Before selecting the next point, it is checked whether the final destination point $P_g = (x_g, y_g)$ is directly accessible. The final destination point is considered visible if it meets one of these two conditions:

- This is closer than any obstacle detected: $d(P_c, P_g) < \min(d_i), \quad i \in V_{collision}$.

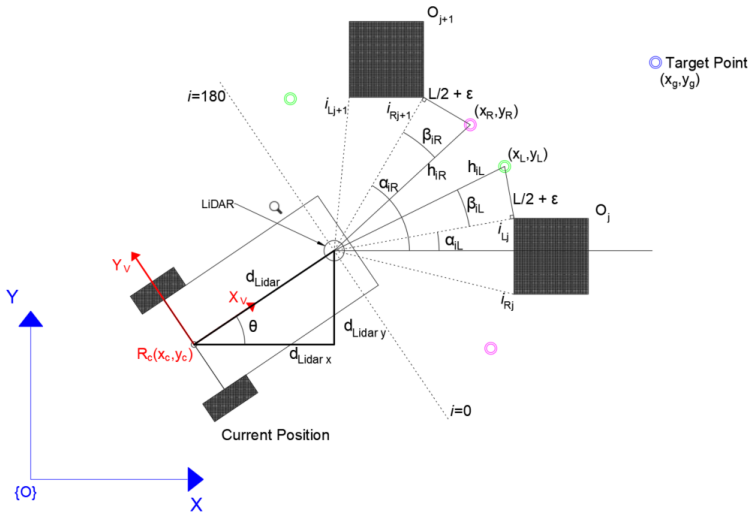

Fig. 4. Intermediate points for all lateral indices in valid segments

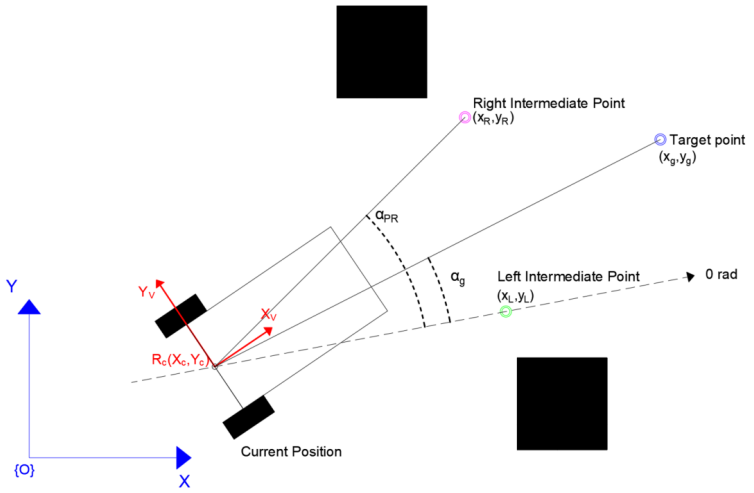

Fig. 5. Validation when a target point is within one of the corridors (rotation from a new 0 axis)

- From a certain angle, the target point is located within one of the navigable corridors defined by the evasion points P_L, P_R (cf. Fig. 5).

Condition two is achieved by performing angular rotations around a new axis 0, drawn from point P_c to point P_L and $P_g(visible)$ if $0 < \alpha_g < \alpha_{P_R}$.

3.6. Target Point Selection (Using the Cost Function)

In this step, we use a cost function inspired by local route planning methods such as VPH [60] and VPH+ [11]. A set of candidate points C is defined by Eq. (24). This set includes all calculated avoidance points (P_L and P_R for each valid corridor) and, in addition, the final destination point P_{goal} if it is visible.

$$C = \{P \mid (P_L, P_R) \in P_{avoidance}\} \cup \{P_{goal} \mid \text{Visible}\}. \tag{24}$$

For each point $P \in C$, a normalized cost function $J(P)$ is calculated considering the total distance of the route and the required change of orientation as in Eq. (25).

$$J(P) = w_{dist} J_{dist\_norm}(P) + w_{angle} J_{angle\_norm}(P), \tag{25}$$

Where w_{dist} and w_{angle} are the respective weighting parameters. Parameters will be defined within the algorithm and w_{dist} is greater than w_{angle}, prioritizing the shortest path. The parameter w_{angle} allows the robot to move along a path and avoid abrupt changes between points. The normalization for $J_{dist\_norm}(P)$ is shown by Eq. (26) and for $J_{angle\_norm}(P)$ by Eq. (27).

$$J_{dist\_norm}(P) = \frac{d(P_c,P) + d(P,P_g)}{2d_{max}}, \tag{26}$$

$$J_{\text{angle-norm}}(P) = \frac{|\text{atan2}(y_P - y_c, x_P - x_c) - \theta_c|}{\pi}. \tag{27}$$

The final target point P_{target} is the one that minimizes the cost function given by Eq. (28). This cost function differs from the VPH and VPH+ methods because while VPH optimizes almost exclusively the angular cost (a comparison between the direction to the target and the current direction), and VPH+ optimizes the safe distance, our approach optimizes the total Euclidean distance of the path. Furthermore, the use of normalized costs ensures that the weights w_{dist} and w_{angle} provide a balance between minimizing distance and penalizing sharp turns, a behavior that is difficult to calibrate in previous VPH methods.

$$P_{target} = \min J(P), \quad P \in C. \tag{28}$$

This point P_{target} is sent to the PD controller as the immediate target.

4. Experimental Design

Several experimental tests were performed on the differential robot described in Section 2.1. The test set was divided into two groups:

1. Validation of the proposed algorithm: The behavior of the algorithm was evaluated in an environment designed to include three common obstacle scenarios in local route planning: a U-shaped structure, a narrow corridor, and a section with multiple obstacles near the final goal.
2. Comparison with VPH: A quantitative comparison of the performance of our algorithm was performed against the VPH method [60] in a specific scenario.

The configuration and results of these tests are detailed below.

4.1. Navigation To a Destination Point With Multiple Obstacles

For this test, the robot starts from the coordinate $(x,y) = (0,0)$ with an orientation of $\theta = 0$ rad. The final target is located at position $(x,y) = (-85,750)$ cms., with a desired orientation of $\theta = 0$ rad. The robot's pose (position and orientation) is estimated using odometry calculated by the ESP32 microcontroller, as described in Section 2.4. On the Raspberry Pi 5, the LiDAR data is processed and an immediate target point (either an intermediate evasion point or the final target) is generated using the local path planning algorithm detailed in Section 3. This point is tracked using the PD controller described in Section 2.3.

The experimental environment for this test is a complex scenario made up of several sections: the robot starts inside a U-shaped structure, navigates through an L-shaped corridor and, in the final stretch towards the target, must manage the simultaneous detection of at least three obstacles.

The parameters used in this test were established by exhaustive trial-error: for the PD controller, the gains were $k_p = 1.1$ and $k_d = 0.5$, with a linear command speed $V_{com} = 1.5$ rad/s. For the evasion algorithm, a detection radius $d_{max} = 200$ cm and a safety distance $\varepsilon = 35$ cm were set. The weights for the normalized cost function were $w_{dist} = 1$ and $w_{angle} = 0.5$. Fig. 6 illustrates the trajectory followed by the robot from the start to the final target, superimposed with the set of all evasion points calculated during the journey. Finally, Fig. 7 presents the same trajectory, but highlighting only the target points selected by the cost function at each instant.

4.2. Comparison Against the Vph Method

To evaluate the relative performance of the proposed algorithm, a comparison was made against the VPH method in a specific navigation environment. An environment consisting of a corridor was set up, where the robot must navigate from the initial coordinate $(x,y) = (0,0)$ to the final target point $(x,y) = (700,-450)$.

The parameters used in our algorithm correspond to those described in Section 4.1. For the VPH algorithm, it was necessary to define its key parameters, including the safety distance (D_f), the minimum window width (W_{max}), and the weights of its cost function (K_1, which prioritizes maintaining the current direction, and K_2, which prioritizes the direction toward the target). It should be noted that the cost function of the original VPH is sensitive to the ratio between K_1 and K_2, where a slightly higher value of K_2 compared to K_1 tends to select angles very close to the target direction, even if these are near the edges of an obstacle. To mitigate this behavior in our comparative implementation of VPH, an additional clipping was applied to the candidate valleys, eliminating directions closest to the detected edges of the obstacle.

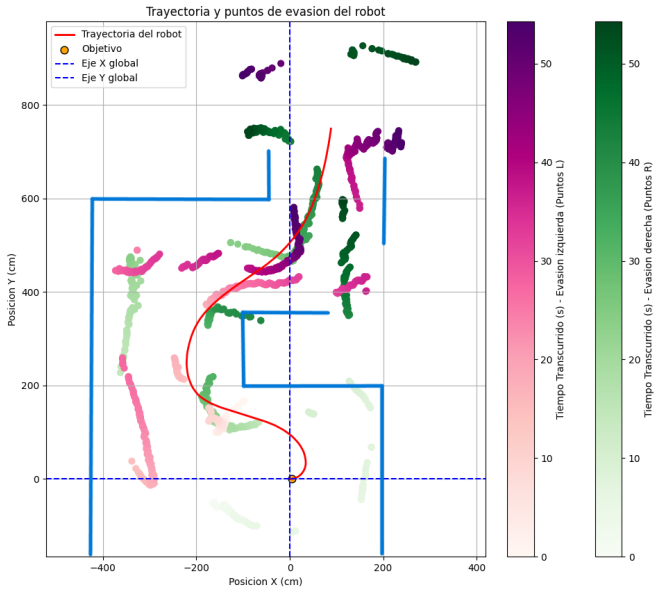

Fig. 6. Robot's path (red line) coincides with avoidance points (green and pink/purple points)

Fig. 8 shows the resulting trajectory when running our algorithm. Fluid movement and constant forward speed are observed, completing the course in approximately 53 seconds. The figure also shows the target points selected by the cost function at each instant. In contrast, Fig. 9 shows the robot's behavior using the VPH algorithm (with the clipping modification). It initially moves at a higher speed on the straight sections of the corridor and it struggles when approaching the front walls, exhibiting a drastic reduction in its speed and less smooth turns. On the final curve, the VPH algorithm failed to reach the target, taking more than 800 seconds.

Finally, Table 1 shows the CPU usage of the algorithms. The proposed algorithm has a slightly lower average (9.9%) compared to VPH (10.4%), while the typical usage (median) is identical (10.0%). However, VPH reached a considerably high peak (85.2%), while the proposed algorithm remained below this (30.1%). This suggests greater stability in CPU usage for our algorithm. Regarding memory usage, both algorithms showed identical and stable behavior, with a constant average of 111.0

MB during the tests.

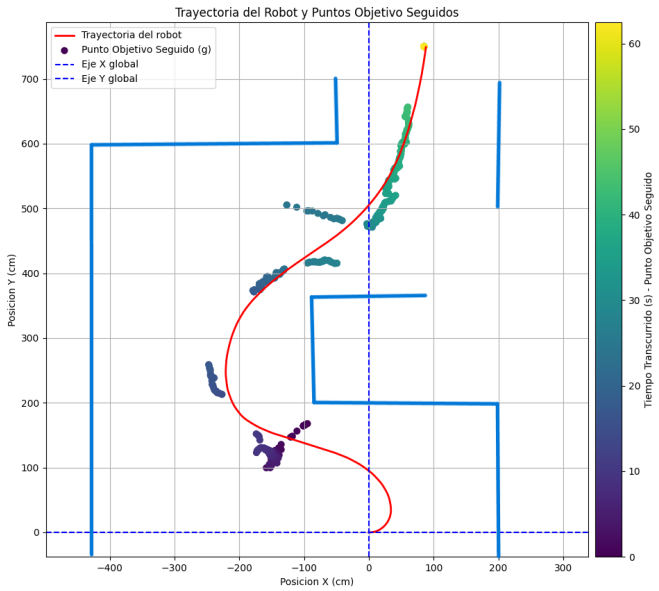

Fig. 7. Robot's path (red line) and target points selected by the cost function (points with color according to the elapsed time)

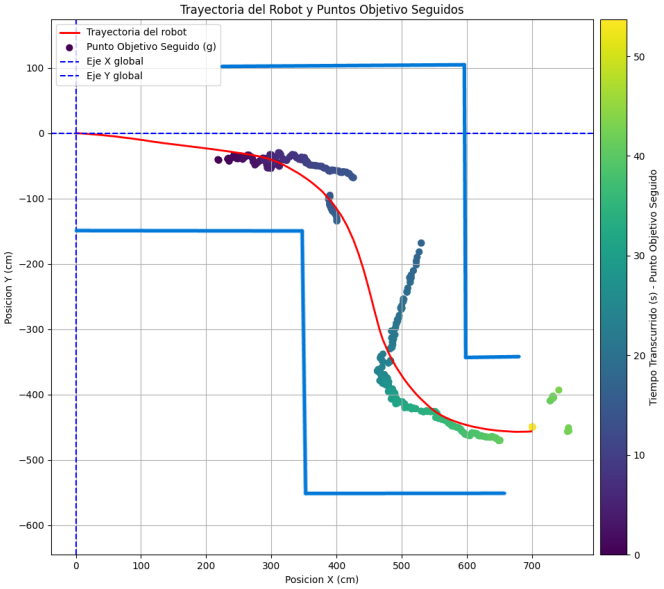

Fig. 8. Robot trajectory using the proposed algorithm. The route followed (red line) and the target points selected by the cost function are shown

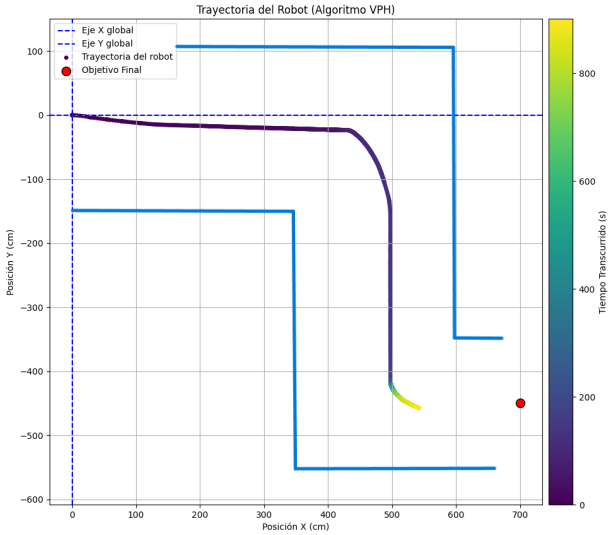

Fig. 9. Robot trajectory using the VPH algorithm (with clipping modification). The trajectory color varies according to the elapsed time

Table 1. Comparison of average computational performance

Metric	VPH Algorithm	Proposed Algorithm
CPU Usage (%)		
Average	10.4 %	9.9 %
Median	10.0 %	10.0 %
Maximum	85.2 %	30.1 %
Memory usage (MB)		
Median	111.0 MB	111.0 MB

5. Conclusions

In this article, a local path planning algorithm was presented for a differential mobile robot, capable of navigating multi-obstacle environments. The proposed method is structured into six real-time steps operating on 2D LiDAR readings and utilizing a PD controller for pose tracking.

The algorithm focus on precision, calculating evasion points at a safe distance ($\varepsilon + L/2$) within geometrically validated corridors. The objective selection is based on a normalized cost function that enables a smooth trajectory, prioritizing the shortest route while penalizing abrupt turns, achieving a balance between efficiency and stability.

The experimental results validated the algorithm's efficacy in a complex environment, successfully bypassing a "U" shaped structure and an "L" shaped corridor, and managing multiple obstacles. It was observed that the robot navigates smoothly, and the cost function correctly discriminates among multiple evasion points and the final goal when the latter is visible.

In comparison with a VPH algorithm with window trimming, the trajectory provided by our method was notably more stable. Although the proposed algorithm requires more geometric validation steps, sequential filtering ensures that the cost function is only applied to a reduced set of safe points. As demonstrated, the computational performance is more stable, with less variability and lower peak CPU usage.

Finally, this paper provides an evasion algorithm optimizing the total Euclidean distance of the

route by using a normalized cost function and allowing for an intuitive adjustment between smoothness and the shortest route. The generation of global target points allows separating evasion from control, which is essential for the algorithm to be compatible with any pose tracking controller. Finally, a method was presented considering the displacement of the LiDAR sensor relative to the robot's coordinate axis and enabling its flexible application in various hardware configurations.

Future works will focus on robustness for handling concave obstacles with a closed line of sight, dynamic obstacles, and mitigating the accumulation of odometry errors in long trajectories through positioning evaluation via image analysis.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] G. Csaba, L. Somlyai and Z. Vámossy, "Mobil robot navigation using 2D LIDAR," *2018 IEEE 16th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pp. 000143-000148, 2018, https://doi.org/10.1109/SAMI.2018.8324002.

[2] S. Ahmed, *Navigation and obstacle avoidance control of an autonomous differential wheeled robot using PID controller in MATLAB simulation*, Books on Demand, 2014, https://books.google.co.id/books?id=yjehDwAAQBAJ&hl=id&source=gbs_navlinks_s.

[3] F. A. Macias *et al*., "VANET network simulation for greenhouse monitoring utilizing two mobile differential robot nodes," *International Journal of Combinatorial Optimization Problems & Informatics*, vol. 15, no. 5, 2024, https://doi.org/10.61467/2007.1558.2024.v15i5.558.

[4] A. J. Moshayedi, A. Abbasi, L. Liao and S. Li, "Path planning and trajectroy tracking of a mobile robot using bio-inspired optimization algorithms and PID control," *2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pp. 1-6, 2019, https://doi.org/10.1109/CIVEMSA45640.2019.9071596.

[5] P. T.-T. Nguyen, S.-W. Yan, J.-F. Liao, and C.-H. Kuo, "Autonomous mobile robot navigation in sparse LiDAR feature environments," *Applied Sciences*, vol. 11, no. 13, p. 5963, 2021, https://doi.org/10.3390/app11135963.

[6] Aqeel-Ur-Rehman and C. Cai, "Autonomous Mobile Robot Obstacle Avoidance Using Fuzzy-PID Controller in Robot's Varying Dynamics," *2020 39th Chinese Control Conference (CCC)*, pp. 2182-2186, 2020, https://doi.org/10.23919/CCC50068.2020.9188467.

[7] S. M. Saidi, R. Mellah, A. Fekik, and A. T. Azar, "Real-Time Fuzzy-PID for Mobile Robot Control and Vision-Based Obstacle Avoidance," *International Journal of Service Science, Management, Engineering, and Technology (IJSSMET)*, vol. 13, no. 1, pp. 1–32, 2022, https://doi.org/10.4018/IJSSMET.304818.

[8] M. Gao, J. Tang, Y. Yang, Z. He and Y. Zeng, "An Obstacle Detection and Avoidance System for Mobile Robot with a Laser Radar," *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 63-68, 2019, https://doi.org/10.1109/ICNSC.2019.8743288.

[9] P. Wu, S. Xie, H. Liu, J. Luo and Q. Li, "A novel algorithm of autonomous obstacle-avoidance for mobile robot based on LIDAR data," *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2377-2382, 2015, https://doi.org/10.1109/ROBIO.2015.7419694.

[10] D. Ghorpade, A. D. Thakare and S. Doiphode, "Obstacle Detection and Avoidance Algorithm for Autonomous Mobile Robot using 2D LiDAR," *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, pp. 1-6, 2017, https://doi.org/10.1109/ICCUBEA.2017.8463846.

[11] J. Gong, Y. Duan, Y. Man, and G. Xiong, "VPH+: An enhanced vector polar histogram method for mobile robot obstacle avoidance," in *2007 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1384–1389, 2007, https://doi.org/10.1109/ICMA.2007.4304000.

[12] S. Gatesichapakorn, J. Takamatsu and M. Ruchanurucks, "ROS based Autonomous Mobile Robot Navigation using 2D LiDAR and RGB-D Camera," *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, pp. 151-154, 2019, https://doi.org/10.1109/ICA-SYMP.2019.8645984.

[13] S. Jiang, S. Wang, Z. Yi, M. Zhang, and X. Lv, "Autonomous navigation system of greenhouse mobile robot based on 3D LiDAR and 2D LiDAR SLAM," *Frontiers in Plant Science*, vol. 13, 2022, https://doi.org/10.3389/fpls.2022.815218.

[14] H. Teng, Y. Wang, D. Chatziparaschis, and K. Karydis, "Adaptive LiDAR odometry and mapping for autonomous agricultural mobile robots in unmanned farms," *Computers and Electronics in Agriculture*, vol. 232, p. 110023, 2025, https://doi.org/10.1016/j.compag.2025.110023.

[15] Y. Cheng and G. Y. Wang, "Mobile robot navigation based on LiDAR," in *2018 Chinese Control and Decision Conference (CCDC)*, pp. 1243–1246, 2018, https://doi.org/10.1109/CCDC.2018.8407319.

[16] M. H. Haider *et al.*, "Robust mobile robot navigation in cluttered environments based on hybrid adaptive neuro-fuzzy inference and sensor fusion," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 9060–9070, 2022, https://doi.org/10.1016/j.jksuci.2022.08.031.

[17] K. Samsudin, F. A. Ahmad, and S. Mashohor, "A highly interpretable fuzzy rule base using ordinal structure for obstacle avoidance of mobile robot," *Applied Soft Computing*, vol. 11, no. 2, pp. 1631–1643, 2011, https://doi.org/10.1016/j.asoc.2010.05.002.

[18] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, Springer, 2011, https://doi.org/10.1007/978-3-642-20144-8.

[19] S. Louda, N. Karkar, F. Seghir, and O. Boutalbi, "Fuzzy Dynamic Feedback Linearization for Efficient Mobile Robot Trajectory Tracking and Obstacle Avoidance in Autonomous Navigation," *International Journal of Robotics and Control Systems*, vol. 5, no. 2, pp. 881–901, 2025, https://doi.org/10.31763/ijrcs.v5i2.1780.

[20] S. Hanana, N. Krichen, M. S. Masmoudi and M. Masmoudi, "Fuzzy Logic Obstacle Avoidance Controller for a Mobile Robot using LiDAR," *2024 IEEE 7th International Conference on Advanced Technologies, Signal and Image Processing (ATSIP)*, pp. 387-390, 2024, https://doi.org/10.1109/ATSIP62566.2024.10638842.

[21] F. M. Suárez, S. Velásquez, E. Díaz, and J. Velásquez, "HYBRID PID-DIFFUSED CONTROL IN ROBOT NON-HOLONOMIC LINE FOLLOWER," *Universidad, Ciencia y Tecnología*, vol. 22, no. 88, 2018, https://web.archive.org/web/20210813055900/https://www.uctunexpo.autanabooks.com/index.php/uct/article/download/130/126.

[22] J. Nuñez, N. Rivas and L. Vinces, "A design of an autonomous mobile robot base with omnidirectional wheels and plane-based navigation with Lidar sensor," *2022 Congreso Internacional de Innovación y Tendencias en Ingeniería (CONIITI)*, pp. 1-4, 2022, https://doi.org/10.1109/CONIITI57704.2022.9953628.

[23] Q. Li *et al*., "Edge computing for mobile robots: Multi-robot feature-based LiDAR odometry with FPGAs," in *2019 12th International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, pp. 1–2, 2019, https://doi.org/10.23919/ICMU48249.2019.9006646.

[24] J. Park, J. Y. Kim, B. Kim and S. Kim, "Global Map Generation using LiDAR and Stereo Camera for Initial Positioning of Mobile Robot," *2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT)*, pp. 1-4, 2018, https://doi.org/10.1109/ICT-ROBOT.2018.8549897.

[25] V. L. Popov, S. A. Ahmed, N. G. Shakev and A. V. Topalov, "Detection and Following of Moving Targets by an Indoor Mobile Robot using Microsoft Kinect and 2D Lidar Data," *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 280-285, 2018, https://doi.org/10.1109/ICARCV.2018.8581231.

[26] M. Matli, "Development of a Mecanum-wheeled mobile robot for dynamic- and static-obstacle avoidance based on laser range sensor," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 20, no. 3, pp. 188–200, 2020, https://doi.org/10.5391/IJFIS.2020.20.3.188.

[27] X. Feng, Y. Xue, H. Dai and L. Wang, "Development and Analysis of Patrol Robot based on Fusion of Dual LIDAR Data," *2023 7th International Conference on Electrical, Mechanical and Computer Engineering (ICEMCE)*, pp. 902-906, 2023, https://doi.org/10.1109/ICEMCE60359.2023.10490961.

[28] Q. Liang, "Autonomous aerial obstacle avoidance using LiDAR sensor fusion," *PLoS ONE*, vol. 18, no. 6, p. e0287177, 2023, https://doi.org/10.1371/journal.pone.0287177.

[29] P. D. Ren Yee, N. Pinrath and N. Matsuhira, "Autonomous Mobile Robot Navigation Using 2D LiDAR and Inclined Laser Rangefinder to Avoid a Lower Object," *2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1404-1409, 2020, https://doi.org/10.23919/SICE48898.2020.9240417.

[30] J. -H. Choi, S. -H. Bae, Y. -C. An and T. -Y. Kuc, "Development of an Advanced Navigation System for Autonomous Mobile Robots for Logistics Environments," *2023 23rd International Conference on Control, Automation and Systems (ICCAS)*, pp. 1286-1291, 2023, https://doi.org/10.23919/ICCAS59377.2023.10316962.

[31] B. Kiran, S. Karthikeyan, M. A. S. Pasha, K. N. Manjunatha, S. M. Kumar and S. V. Moras, "Design and Development of Autonomous Mobile Robot for Mapping and Navigation System," *2022 IEEE Pune Section International Conference (PuneCon)*, pp. 1-5, 2022, https://doi.org/10.1109/PuneCon55413.2022.10014944.

[32] L. Sui and L. Lin, "Design of Household Cleaning Robot Based on Low-cost 2D LIDAR SLAM," *2020 International Symposium on Autonomous Systems (ISAS)*, pp. 223-227, 2020, https://doi.org/10.1109/ISAS49493.2020.9378863.

[33] M. Takahashi, K. Kobayashi, K. Watanabe and T. Kinoshita, "Development of prediction based emergency obstacle avoidance module by using LIDAR for mobile robot," *2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, pp. 561-564, 2014, https://doi.org/10.1109/SCIS-ISIS.2014.7044725.

[34] I. Hassani, I. Ergui, and C. Rekik, "Turning point and free segments strategies for navigation of wheeled mobile robot," *International Journal of Robotics & Control Systems*, vol. 2, no. 1, pp. 172–186, 2022, https://doi.org/10.31763/ijrcs.v2i1.586.

[35] Y. Yang, G. Yang, Y. Tian, T. Zheng, L. Li and Z. Wang, "A robust and accurate SLAM algorithm for omni-directional mobile robots based on a novel 2.5D lidar device," *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 2123-2127, 2018, https://doi.org/10.1109/ICIEA.2018.8398060.

[36] C. Pang, X. Zhong, H. Hu, J. Tan, X. Peng, and J. Zeng, "Adaptive Obstacle Detection for Mobile Robots in Urban Environments Using Downward-Looking 2D LiDAR," *Sensors*, vol. 18, no. 6, p. 1749, 2018, https://doi.org/10.3390/s18061749.

[37] H. Fu, Q. Chen, Z. Chen and S. Wen, "Costmap Construction and Pseudo-Lidar Conversion Method of Mobile Robot Based on Monocular Camera," *2021 33rd Chinese Control and Decision Conference (CCDC)*, pp. 3163-3168, 2021, https://doi.org/10.1109/CCDC52312.2021.9602346.

[38] R. Uchida, K. Kobayashi, T. Ohkubo, K. Watanabe, N. J. Sebi and K. C. Cheok, "Development of A Raindrop-aware Environment Detection Algorithm for 3D-LiDAR Based Outdoor Mobile Robot Navigation," *2021 60th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1482-1487, 2021, https://ieeexplore.ieee.org/abstract/document/9555336.

[39] A. N. Kravtsov and B. S. Goryachkin, "Integration of Large Language Models for Autonomous Navigation of a Mobile Robot," *2025 7th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)*, pp. 1-6, 2025, https://doi.org/10.1109/REEPE63962.2025.10971154.

[40] M. A. Nursyeha, M. Rivai, D. Purwanto and Tukadi, "LiDAR Equipped Robot Navigation on Behavior-based Formation Control for Gas Leak Localization," *2020 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pp. 89-94, 2020, https://doi.org/10.1109/ISITIA49792.2020.9163758.

[41] X. Ou, Z. You, and X. He, "Local Path Planner for Mobile Robot Considering Future Positions of Obstacles," *Processes*, vol. 12, no. 5, p. 984, 2024, https://doi.org/10.3390/pr12050984.

[42] Y. Liu, "Obstacle Avoidance for Autonomous Mobile Robots in Unstructured Human Environments," *2021 6th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pp. 28-32, 2021, https://doi.org/10.1109/CACRE52464.2021.9501301.

[43] M. -A. Chung and C. -W. Lin, "Pose Detection of a Mobile Robot Based on LiDAR Data," *2021 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*, pp. 1-5, 2021, https://doi.org/10.1109/iFUZZY53132.2021.9605092.

[44] K. Horichi, T. Kinoshita, T. Yoshida and K. Kobayashi, "Spot-based LIDAR profile estimation algorithm for mobile robots in motion," *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 237-242, 2016, https://doi.org/10.1109/SICE.2016.7749237.

[45] S. T. Padgett and A. F. Browne, "Vector-based robot obstacle avoidance using LIDAR and mecanum drive," *SoutheastCon 2017*, pp. 1-5, 2017, https://doi.org/10.1109/SECON.2017.7925312.

[46] C. Han *et al.*, "Visual Navigation and Obstacle Avoidance Control for Agricultural Robots via LiDAR and Camera," *Remote Sensing*, vol. 15, no. 22, p. 5402, 2023, https://doi.org/10.3390/rs15225402.

[47] F. Paciolla, N. Pace, G. Barile, P. Patimisco, F. Valenza and S. Pascuzzi, "Autonomous navigation simulation of an agricultural robot during soil fertilization in open fields," *2023 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, pp. 640-645, 2023, https://doi.org/10.1109/MetroAgriFor58484.2023.10424176.

[48] M. J. Santiago-Vásquez, D. Carmona-Ruiz, K. I. Salado-Chávez and O. D. Ramírez-Cárdenas, "Development of an Autonomous Omnidirectional Robot for Obstacle Avoidance Using Potential Fields," *2024 XXVI Robotics Mexican Congress (COMRob)*, pp. 79-84, 2024, https://doi.org/10.1109/COMRob64055.2024.10777460.

[49] H. Xiao, Y. Song, Q. Lu, S. Zhou, and Y. He, "Reinforcement learning-driven dynamic obstacle avoidance for mobile robot trajectory tracking," *Knowledge-Based Systems*, vol. 297, p. 111974, 2024, https://doi.org/10.1016/j.knosys.2024.111974.

[50] M. G. Mohanan and A. Salgaonkar, "Robotic motion planning in dynamic environments and its applications," *International Journal of Robotics and Control Systems*, vol. 2, no. 3, pp. 497–509, 2022, https://doi.org/10.31763/ijrcs.v2i4.816.

[51] D. Hutabarat, M. Rivai, D. Purwanto and H. Hutomo, "Lidar-based Obstacle Avoidance for the Autonomous Mobile Robot," *2019 12th International Conference on Information & Communication Technology and System (ICTS)*, pp. 197-202, 2019, https://doi.org/10.1109/ICTS.2019.8850952.

[52] S. Shrestha, S. Parajuli, P. Shah, B. Shrestha and M. K. Guragai, "Navigation of Mobile Robot with Nav2 and SLAM Using LiDAR," *2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL)*, pp. 579-585, 2025, https://doi.org/10.1109/ICSADL65848.2025.10933029.

[53] D. Shen, Y. Huang, Y. Wang and C. Zhao, "Research and Implementation of SLAM Based on LIDAR for Four-Wheeled Mobile Robot," *2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE)*, pp. 19-23, 2018, https://doi.org/10.1109/IRCE.2018.8492968.

[54] S. Wang, H. Pang and X. Huang, "Research on Indoor Mobile Robot Mapping based on the Fusion of LiDAR and Depth Camera," *2025 2nd International Conference on Artificial Intelligence and Digital Technology (ICAIDT)*, pp. 246-251, 2025, https://doi.org/10.1109/ICAIDT66272.2025.00053.

[55] R. Qiu, H. Wang, H. Qiu, L. Huang, L. Yang and H. Qiu, "Research on the Control System of Dual Lidar Mobile Robot," *2021 4th International Conference on Robotics, Control and Automation Engineering (RCAE)*, pp. 289-294, 2021, https://doi.org/10.1109/RCAE53607.2021.9638944.

[56] J. Kim, H. Jeong, and D. Lee, "Single 2D lidar based follow-me of mobile robot on hilly terrains," *Journal of Mechanical Science and Technology*, vol. 34, no. 12, pp. 5171–5180, 2020, https://doi.org/10.1007/s12206-020-0835-7.

[57] Z. Xuexi, L. Guokun, F. Genping, X. Dongliang and L. Shiliu, "SLAM Algorithm Analysis of Mobile Robot Based on Lidar," *2019 Chinese Control Conference (CCC)*, pp. 4739-4745, 2019, https://doi.org/10.23919/ChiCC.2019.8866200.

[58] J. de Heuvel, X. Zeng, W. Shi, T. Sethuraman and M. Bennewitz, "Spatiotemporal Attention Enhances Lidar-Based Robot Navigation in Dynamic Environments," in *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4202-4209, 2024, https://doi.org/10.1109/LRA.2024.3373988.

[59] J. Chen, J. Cao, Z. Cheng and S. Jiang, "Towards Efficient Distributed Collision Avoidance for Heterogeneous Mobile Robots," in *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 3605-3619, 2024, https://doi.org/10.1109/TMC.2023.3279906.

[60] Dong An and H. Wang, "VPH: a new laser radar based obstacle avoidance method for intelligent mobile robots," *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No.04EX788)*, pp. 4681-4685 Vol.5, 2004, https://doi.org/10.1109/WCICA.2004.1342407.

[61] J. S. Saputro, S. Setyawan, and S. Susanto, "Collision avoidance in mini autonomous electric vehicles using artificial potential fields for outdoor environment," *International Journal of Robotics & Control Systems*, vol. 5, no. 2, pp. 1139–1155, 2024, https://doi.org/10.31763/ijrcs.v5i2.1708.

[62] H. Dong, C. -Y. Weng, C. Guo, H. Yu and I. -M. Chen, "Real-Time Avoidance Strategy of Dynamic Obstacles via Half Model-Free Detection and Tracking With 2D Lidar for Mobile Robots," in *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 4, pp. 2215-2225, 2021, https://doi.org/10.1109/TMECH.2020.3034982.