Computational evaluation of logistic potential fields in real-time obstacle navigation

Ihor Berizka[1,†], Ivan Karbovnyk[1,*,†]

[1] *Ivan Franko Lviv National University, Faculty of Electronics and Computer Technologies, Department of Radiophysics and Computer Technologies, Tarnavskoho 107 79017 Lviv, Ukraine*

Abstract

Obstacle avoidance is a critical capability for autonomous mobile robots, enabling safe operation in dynamic and unstructured environments. This paper proposes a novel approach based on the Artificial Potential Field (APF) method utilizing the Logistic function for obstacle avoidance problem. A complete mathematical formulation of the model is presented and analyzed. Validation was performed using a simulation framework built on ROS 2, the Gazebo simulator, and the TurtleBot3 Burger platform. Extensive simulations were conducted, including LiDAR-based environment sampling and visualizations of repulsive, attractive, and total potential field distributions. The results confirm the correctness of the method and demonstrate effective real-time navigation. RViz visualization further illustrated the robot's smooth trajectory and gradual heading changes across 28 navigation steps. Performance benchmarking in C++ showed that evaluating the logistic function incurs a computational cost approximately 28% higher than that of the Gauss function. However, this trade-off is justified by the logistic function's improved smoothness and earlier obstacle response, which enhance the robot's ability to anticipate and adjust to dynamic obstacles. Logistic function ensures smoother potential transitions and more stable path generation, making it well-suited for scenarios requiring responsive yet fluid motion planning.

Keywords

artificial potential field method, cyber-physical system, edge computing, information technologies, IoT concepts, obstacle avoidance, robotics

1. Introduction

Mobile robots constitute a prominent area of research and development within the field of robotics. These systems are designed to navigate independently and make real-time, context-aware decisions based on continuous sensory input, enabling operation without human intervention. Their adaptability to dynamic and unstructured environments has led to widespread deployment across various domains. In the hospitality sector, service robots autonomously deliver food and beverages, while in industrial settings, mobile transport robots efficiently handle the movement of goods. A particularly impactful application is that of autonomous vehicles, which leverage advanced sensor technologies and computational algorithms to perceive their surroundings and navigate complex traffic environments without human control. These examples underscore the broad applicability and transformative potential of mobile robots in sectors such as logistics, hospitality, and transportation. As such, the development and optimization of these systems continue to be a central focus in robotics research, with ongoing efforts directed at improving their performance, autonomy, and adaptability.

A core elements of autonomous mobile robot are software components responsible for path planning and obstacle avoidance. These components provide key functionalities such as autonomous parking, emergency maneuvers, and self-directed navigation. Path planning strategies are typically divided into two categories: local and global. Global path planning relies on information from

MoDaST 2025: Modern Data Science Technologies Doctoral Consortium, June, 15, 2025, Lviv, Ukraine

[1*] Corresponding author.

[†] These authors contributed equally.

ihor.berizka@lnu.edu.ua (I. Berizka); ivan.karbovnyk@lnu.edu.ua (I. Karbovnyk)

0009-0007-1111-1493 (I. Berizka); 0000-0000-0000-0000 (I. Karbovnyk); 0000-0002-3697-4902

Geographic Information Systems (GIS) and global localization techniques, requiring the robot to store a detailed, large-scale map of its environment. This enables long-range navigation tasks, such as urban traversal. In contrast, local path planning operates using the robot's relative position and real-time sensory input to perceive nearby obstacles. It is focused on short-range, reactive navigation in dynamic settings—such as avoiding pedestrians or maneuvering around moving vehicles. Local planning plays a crucial role in enabling safe, context-aware behavior in unpredictable and rapidly evolving environments.

Numerous methods have been developed to address the complexities inherent in global and local path planning. Each method presents distinct advantages and limitations, and selecting an appropriate algorithm is critical to ensuring the efficiency, robustness, and safety of autonomous navigation. The existing literature offers several comprehensive surveys that examine these algorithms in detail, including their underlying principles, performance characteristics, and suitability for various application domains [1, 2, 3, 4].

Obstacle detection and avoidance are fundamental components of local path planning algorithms, serving a vital role in ensuring the safety of both autonomous systems and their surrounding environments. Over the past few decades, this area has received considerable attention in the research community, leading to the development of a diverse set of approaches, many of which have demonstrated effectiveness in real-world deployments.

Effective collision avoidance requires that the robot is able to detect obstacles and dynamically replan its trajectory in real-time mode. Such responsiveness is critical for enabling safe and efficient navigation in complex and dynamically changing environments.

Obstacle avoidance generally starts with the acquisition of sensory data, where onboard sensors like LiDAR, sonar, or cameras are used to identify potential obstacles. Upon identifying an obstacle, the system must rapidly compute an alternative trajectory that ensures safe traversal around the object. This trajectory must be generated with minimal computational latency to support real-time motion adjustments, thereby preventing collisions while preserving smooth navigation.

The fundamental objective of obstacle avoidance is to enable the robot to reach a designated target location while continuously modifying its trajectory in response to obstacles encountered along the planned path. A schematic representation of the path planning and obstacle avoidance algorithm is presented in Fig. 1.

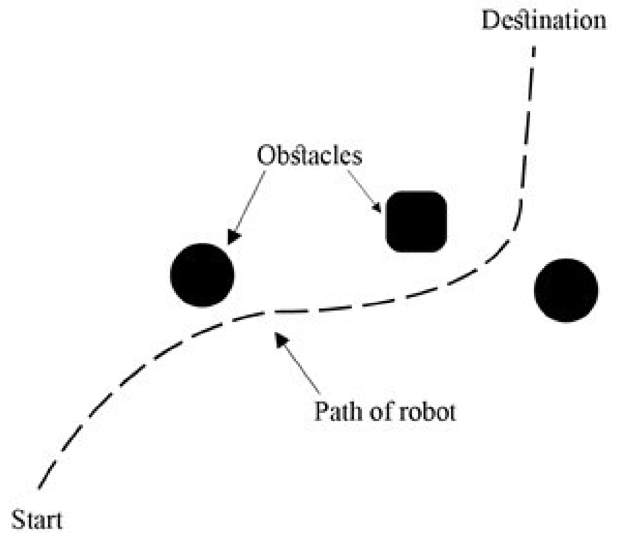

Figure 1: Schematic representation of the obstacle avoidance problem. The original problem is divided into two subtasks: calculating a new collision-free trajectory and sending commands to the actuators to move the robot along the new trajectory.

The process of obstacle detection and trajectory adjustment forms an iterative loop, wherein the robot continuously monitors its environment and performs real-time path modifications as needed.

This cycle persists until the robot successfully reaches its target destination. The iterative nature of this procedure underscores the dynamic characteristics of autonomous navigation and highlights the essential role of reliable obstacle detection and path replanning mechanisms in ensuring the safe, efficient, and adaptive operation of mobile robotic systems.

2. Literature Review

2.1. Mathematical Model of Logistic APFM

The Artificial Potential Field (APF) method is a widely recognized technique in robotics, particularly in the areas of trajectory planning and obstacle avoidance. Originally proposed by Khatib in 1984 [5], this method models the robot's environment using virtual attractive and repulsive forces to perform navigation.

In the APF method, a virtual potential field is defined such that the goal location produces an attractive force and obstacles produce repulsive forces. The autonomous agent is influenced by the resultant force obtained from the summation of these forces, guiding it toward the target while steering it away from collisions. This force-based navigation framework enables real-time path adjustment based on the robot's interactions with its environment [5]. The mathematical formulations of the attractive, repulsive, and total potential fields are presented in equations (1)–(3).

$$\boldsymbol{f}_{total} = \boldsymbol{f}_{att} + \boldsymbol{f}_{rep} \tag{1}$$

$$\boldsymbol{f}_{att} = k_{att} * \left(\frac{\boldsymbol{r}_{goal} - \boldsymbol{r}}{|\boldsymbol{r}_{goal} - \boldsymbol{r}|}\right) \tag{2}$$

$$\boldsymbol{f}_{rep} = \begin{cases} -k_{rep} * \sum_{i=1}^{n} \left(\frac{1}{d_i} - \frac{1}{d_{max}}\right) * \boldsymbol{s}_i, if\ d_i < d_{max} \\ 0 \end{cases} \tag{3}$$

Despite its advantages, the traditional APFM is subject to several well-documented limitations. A primary concern is the occurrence of local minima—situations in which the robot becomes trapped at a location in the workspace which is not the target, due to the equilibrium of forces. This can prevent the robot from progressing toward its target. Furthermore, the classical APF approach may lead to unreachable goal configurations or the generation of suboptimal, inefficient trajectories, particularly in complex or cluttered environments [6].

To overcome the limitations of the classical Artificial Potential Field (APF) method, various enhanced approaches were proposed. One such modification, introduced in [7], incorporates probabilistic elements into the traditional framework. Known as the ODG-PF method, this approach was specifically designed to enhance obstacle detection and estimate the probability of collisions with detected obstacles. It introduces novel formulations for both fields, along with an improved strategy for determining movement direction.

In [8], a more detailed review of the ODG-PF method was provided, along with proposals for future research areas in this domain. In particular, a mathematical model of the APFM was introduced, utilizing the Laplace function to represent the repulsive field. In [9] we performed computational evaluation of Laplace APFM modifications and in [10] we introduced and performed evaluation of Hyperbolic Secant APFM modification.

In this paper, we propose the use of the Logistic function to model the repulsive force within the artificial potential field framework.

$$f_k(\theta_i) = A_k * sech^2\left(\frac{\pi(\theta_k - \theta_i)}{2\sqrt{3} * \sigma_k}\right) \tag{4}$$

Where θ_k corresponds to the central angle of the k_{th} obstacle, σ_k is half of angle occupied by the k_{th} obstacle.

In the context of a sensor with 1-degree angular resolution, each index can be interpreted as a discrete representation of an angle within the 0° to 360° range. For each of these angular positions, a corresponding logistic function is used to model the influence of a detected obstacle. These individual contributions collectively form the repulsive field—a continuous vector field which guides the robot away from obstacles. This formulation enables fine-grained, direction-sensitive obstacle avoidance based on high-resolution sensor data.

A key parameter in this formulation is A_k - the scaling coefficient, which is carefully tuned to ensure that the function adequately captures the spatial extent of each obstacle. Proper adjustment of this coefficient is essential for accurately modeling the distribution of repulsive forces in the vicinity of obstacles. Its value directly influences the sharpness and spatial spread of the repulsive field, thereby affecting the robot's ability to react to nearby hazards and is obtained using equation (5).

$$A_k = \bar{d}_k * sech^2(\frac{\pi}{2\sqrt{3}}) \tag{5}$$

Where $\bar{d}_k = d_{max} - d_k$, d_{max} is sensor range distance.

To account for the influence of several obstacles, the overall repulsive field is computed as the superposition of the individual repulsive fields generated by each obstacle. Consequently, the repulsive potential is expressed as a function of the angular variable, reflecting the cumulative effect of obstacle-induced forces across all sensed directions.

$$f_{rep}(\theta_i) = \sum_{k=1}^{n} A_k * sech^2(\frac{\pi(\theta_k - \theta_i)}{2\sqrt{3} * \sigma_k}) \tag{6}$$

The subsequent stage involves the computation of the attractive field, as defined in equation (7). This field models the virtual force that attracts the robot toward the target direction of motion. When combined with the repulsive field, the attractive component contributes to the overall potential. The calculated trajectory shows a balance between avoiding obstacles and motion toward the goal, enabling the robot to navigate safely while continuously progressing toward the target location.

$$f_{att}(\theta_i) = \gamma|\theta_{goal} - \theta_i| \tag{7}$$

$$f_{total}(\theta_i) = f_{att}(\theta_i) + f_{rep}(\theta_i) \tag{8}$$

$$\theta_{dir} = argmin(f_{total}) \tag{9}$$

Parameter γ is selected experimentally and is set to 1.5. Equation (8) represents total field produced by the system, and the safe direction of robot movement is determined using (9).

2.2. Evaluation Framework

A plethora of software frameworks—commonly referred to as middleware—has been developed to support modularity and flexibility in robotic systems, enabling the integration of different hardware platforms and distributed processing capabilities [11].

Among these, the Robot Operating System, became de facto a standard, accelerating research and development efforts by offering reusable components, standardized communication, and extensive tooling [12]. However, ROS 1 was initially designed with a focus on research rather than commercial deployment, resulting in limited support for hard real-time capabilities, fault tolerance, network optimization, and robust security [13, 14].

Dozens of studies demonstrated critical vulnerabilities in ROS-based systems. For instance, publicly accessible ROS nodes have been shown to be susceptible to unauthorized access, and formal assessments have revealed multiple security weaknesses across the ROS communication architecture. Furthermore, the centralized ROS Master design introduces a single point of failure, limiting the system's scalability in multi-robot and heterogeneous environments [14]. As a result, many production-grade applications built upon ROS 1 required extensive customizations—such as real-time kernel patches, network encapsulation, or additional process isolation layers—to achieve the robustness needed for industrial deployment [13].

Recognizing these limitations, the robotics community initiated the development of next gen ROS framework – ROS 2. Built from the ground up, ROS 2 addresses critical requirements for modern robotic systems. The adoption of the Robot Operating System (ROS), and particularly its modern iteration ROS 2, offers numerous advantages for the developers in robotic domain. ROS 2 accelerates the engineering process by providing a comprehensive ecosystem of open-source algorithms, libraries and tools. Also, it supports the seamless integration of heterogeneous subsystems and promotes interoperability among software components—capabilities that are essential for building reliable, modular, and scalable robotic platforms. Moreover, the active global ROS community contributes continuous innovation, validation, and technical support across a wide array of robotic applications.

The integration of ROS 2 as a unified middleware framework also facilitates the transition from simulation to real-world deployment (sim-to-real), enabling researchers to evaluate and refine their solutions under realistic conditions beyond controlled laboratory settings. In this work, the proposed Logistic Artificial Potential Field Method (APFM) was implemented as a reusable C++ library, allowing for flexible integration into larger robotic software architectures. This modular implementation supports direct incorporation into the ROS 2 ecosystem, thereby enhancing the portability and extensibility of this approach.

To order to perform simulation, the Gazebo simulator was selected due to its optimal integration capabilities with ROS 2. Gazebo offers high-fidelity dynamic modeling, accurate sensor emulation, and native compatibility with the ROS ecosystem, making it a widely adopted platform for simulating robotic systems. As the reference simulator for ROS-based development, Gazebo enables physically realistic representations of robotic behavior, including multi-body dynamics, contact interactions, and the generation of sensor data such as depth images, LiDAR scans, and inertial measurements.

The Gazebo–ROS 2 interface facilitates seamless bi-directional communication through dedicated plugins, enabling synchronized control loops and consistent timing between simulated environments and real-world implementations. This integration supports rigorous validation of algorithms in simulation before physical deployment, thereby significantly reducing development time and risk.

Moreover, Gazebo's ability to model complex scenarios—including multi-robot systems and dynamic, unstructured environments—combined with ROS 2’s distributed architecture, provides a scalable and flexible framework for advanced robotic research and system-level verification. A more comprehensive review of Gazebo and alternative simulation platforms is provided in [9].

For the experimental platform, a ROS-compatible wheeled mobile robot was required. After evaluating available options, the TurtleBot series was selected due to its widespread use in academic research. As reported in [9], TurtleBot platforms appear in approximately 20% of publications related to mobile robot research, highlighting their suitability for scientific applications. Notably, TurtleBot3 offers native integration with ROS 2 and Gazebo, along with comprehensive documentation, simulation models, and readily available software packages.

TurtleBot3 was chosen for its modular design, affordability, and active community support. It features scalable hardware and a suite of essential sensors—including LiDAR, IMU, and wheel encoders—making it well-suited for implementing and testing autonomous navigation algorithms. Its compatibility with ROS 2 enables efficient development and validation of the proposed obstacle avoidance method in a realistic yet controlled environment.

3. Computational Evaluation

3.1. Simulation in Gazebo

To evaluate the proposed mathematical model of the Logistic Artificial Potential Field Method (APFM), we conducted simulations in a virtual environment containing obstacles, as illustrated in Fig. 2. The key parameters and their corresponding values required for the Logistic APFM are summarized in Table 1. The workspace shown on Fig. 2 depicts configuration employed in the Gazebo simulation environment to evaluate the proposed obstacle avoidance method. The experimental setup includes a TurtleBot3 Burger robot positioned in the middle of a room, depicted as a small cylinder marked with a white dot. The robot's physical parameters—including dimensions, mass, and sensor specifications—were sourced from the TurtleBot3 website and accurately modeled within the simulation to ensure realistic behavior. The environment consists of walls enclosing the workspace and multiple obstacles of varying geometries (cylindrical and rectangular), creating a complex and dynamic scenario for testing obstacle avoidance performance. Blue lines emanating out of the robot represent 1D LiDAR scan rays, with the sensor configured at a 1° angular resolution to achieve high precision. In this paper, LiDAR scan completes a full 360° rotation, delivering a comprehensive snapshot of the workspace.

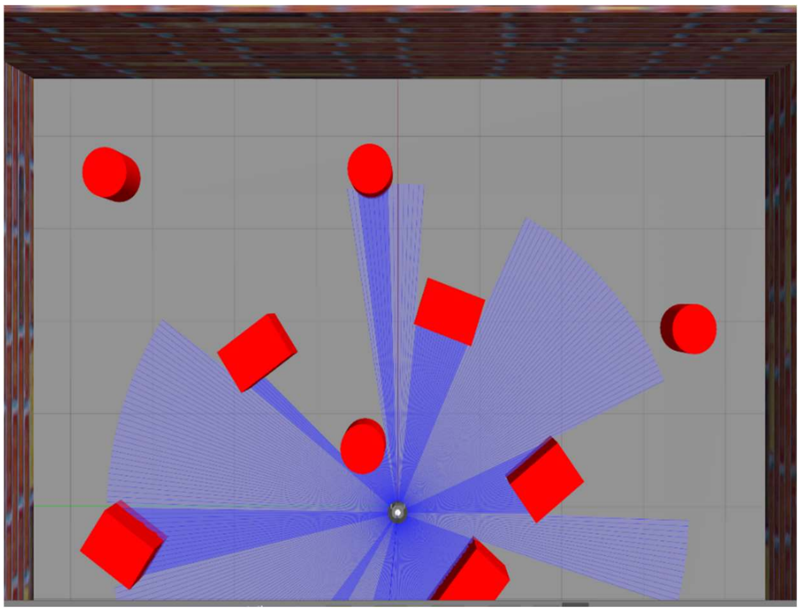

Figure 2: Workplace configuration in Gazebo simulator.

In Table 1 summarizes the key parameters and their corresponding values required for the Logistic APFM.

Table 1
Parameters and corresponding values of the Logistic APFM

Parameter	Value
Threshold distance	1 m
Robot diameter	0.2 m
LiDAR max range	6 m
LiDAR resolution	-179...180, step 1°
Gamma	1.5
Goal direction	0°
Robot linear speed	0.1 m/s

Under the specified configuration and parameter values, the robot navigates forward toward the target wall while dynamically avoiding collisions with obstacles in its path. The core obstacle avoidance behavior is governed by the Logistic APFM model, which calculates repulsive forces from nearby obstacles while maintaining attraction toward the goal position. Fig. 3 shows a complete single step of the algorithm is presented in the pseudocode.

```
Algorithm 1 General Robot Navigation Procedure Based on APFM
while the target location is not reached do
    Step 1: Environment Scanning
        Acquire 1D LiDAR sensor data
        Identify obstacles within the sensitivity zone
        Enlarge detected obstacles based on robot dimensions
    Step 2: Force Computation
        Compute repulsive forces using Eq. (6)
        Compute attractive force toward the target using Eq. (8)
        Determine the resultant force vector using Eq. (9)
    Step 3: Safe Direction Evaluation
        Calculate the safe movement angle according to Eq. (10)
    if the safe direction differs from the target direction then
        Rotate the robot to align with the computed safe angle
        Move forward for a predefined time interval (e.g., 1 second)
        Reorient the robot back toward the original target direction
    else
        Continue moving forward for the same time interval
    end if
end while
```

Figure 3: Pseudocode of a Robot Navigation Algorithm used in this study.

The following section examines in detail the first three steps of the proposed algorithm. At time t_1, the robot is situated at the geometric center of the room, as depicted in Fig. 2. Fig. 4 depicts 1D LiDAR data sample corresponding to this position.

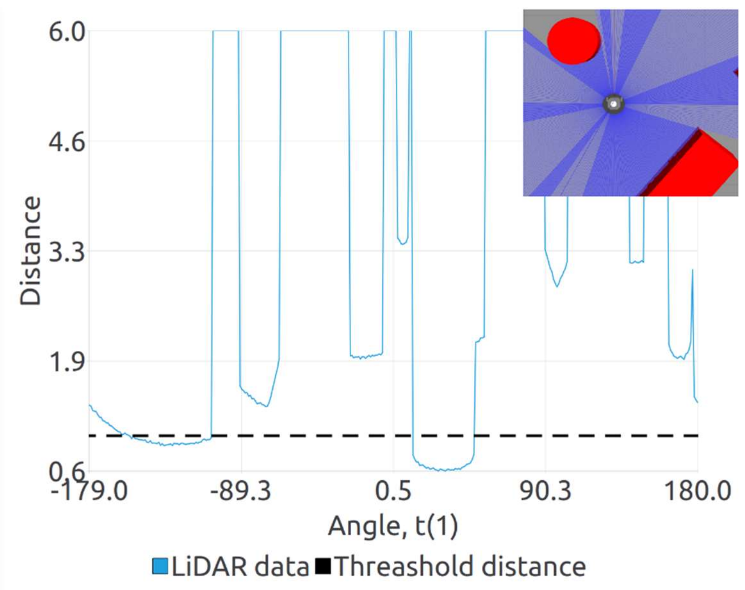

Figure 4: The 1D LiDAR sample at the initial step. The solid line represents data a single complete LiDAR scan. The dashed curve depicts the threshold distance; objects detected closer than this threshold are classified as obstacles. The picture in the top-right corner demonstrates the current robot position in the simulator.

Entities detected at distances shorter than a predefined threshold are classified as obstacles. Analyzing the LiDAR scan we observe two distinct, continuous regions with distance measurements below this threshold, located approximately within the angular intervals of [−170°, −100°] and [+10°, +45°]. Based on this observation, the APFM is expected to identify two obstacles within the specified angular sectors. The obstacles detected by the APFM, highlighted in red, correspond closely with the angular positions inferred from the LiDAR data, thereby confirming the accuracy of the obstacle detection process.

Plot of the repulsive, attractive, and total forces calculated by APFM at time step t_1 are presented on Fig. 5. The x-axis represents the angular direction relative to the robot's forward orientation, ranging from −179° to 180°. The y-axis denotes the magnitude of the corresponding force components.

The blue curve (repulsive force) exhibits two prominent peaks, indicating obstacles exerting significant influence on the robot's path planning. Those peaks correspond to the LiDAR data presented on Fig. 4. The attractive force (black dashed line) increases linearly with angle, steering the robot toward the goal direction. The total force (green curve), obtained by vector summation of the attractive and repulsive components, reaches a minimum at approximately −18°, which is identified as the safe angle for navigation at this time step. This minimum represents the direction in which the net potential field guides the robot, effectively balancing obstacle avoidance and goal-seeking behavior.

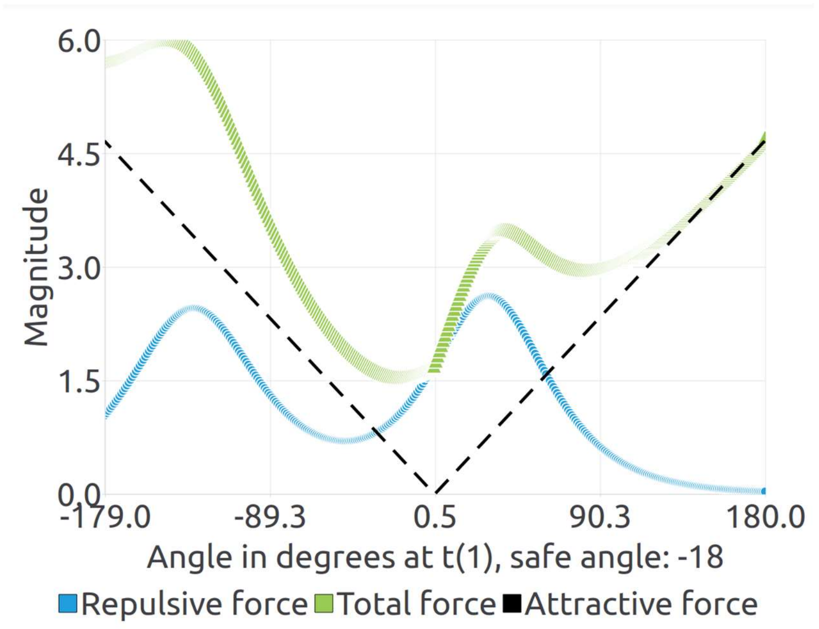

Figure 5: Repulsive, attractive, and total forces computed at the specified timestamp t_1.

As outlined in the navigation algorithm illustrated above, the robot first performs a rotational maneuver toward the computed safe direction. This reorientation allows the robot to avoid the obstacles identified in its immediate surroundings. Following this adjustment, the robot proceeds by moving forward for a fixed duration of one second, thereby making incremental progress toward the goal. Upon completing this forward motion, the robot executes a corrective rotation to realign its heading, allowing it to resume its intended trajectory. This sequence of operations is executed iteratively and continues systematically until the robot either successfully reaches the designated target position.

Under the current experimental setup, the robot successfully navigated the workspace and reached the designated target in 28 discrete steps. To further evaluate the reliability of the proposed method, a detailed analysis of two additional representative iterations is presented. The first case focuses on an intermediate location along the trajectory, corresponding to timestamp t_{16}. The second case examines the robot's final step to the target, recorded at timestamp t_{28}, where the goal is near the wall.

For both time steps, a comprehensive assessment of the potential field forces—repulsive, attractive, and resultant total—acting on the robot will be conducted. Additionally, the complete navigation trajectory, visualized using the RViz tool, will be analyzed to evaluate the smoothness and consistency of the obstacle avoidance behavior throughout the execution of the algorithm.

Analysis of the data presented in Fig. 6 reveals a single continuous region below the threshold line, spanning approximately the angular interval of [−110°, −45°]. Based on this observation, the Artificial Potential Field Method (APFM) is expected to identify a single obstacle within this angular sector. Moreover, the location of the obstacle in the simulation environment corresponds closely to the angular position inferred from the LiDAR scan, confirming the consistency of the detection process.

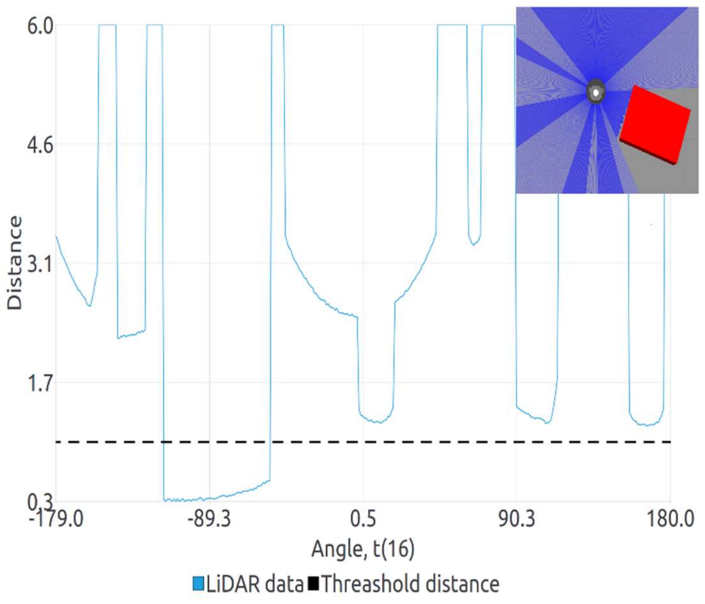

Figure 6: The 1D LiDAR sample at the t_{16} step. The solid line represents data a single complete LiDAR scan. The dashed curve depicts the threshold distance; objects detected closer than this threshold are classified as obstacles. The picture in the top-right corner demonstrates the current robot position in the simulator.

It is also noteworthy that the detected obstacle is located close to the robot, at an approximately 0.4m near the side of the robot. This results in a broader influence on the obstacle enlargement process, leading to the wide peak in the repulsive force generated by this obstacle. Such a peak significantly affects the total force distribution and, consequently, the robot's navigational decisions.

Fig. 7 presents the force profiles at a later time instance. In this case, the repulsive force has a single pronounced peak centered around −89°, indicating an isolated obstacle in that direction. The attractive force retains its characteristic V-shaped profile, with its minimum at 0°, corresponding to the goal direction. The total force also reaches its minimum at 0°, indicating that no significant obstacle obstructs the direct path. Consequently, the robot can proceed straight toward the target without needing to adjust its heading. This scenario illustrates the APFM's ability to sustain goal-directed motion when the environment allows.

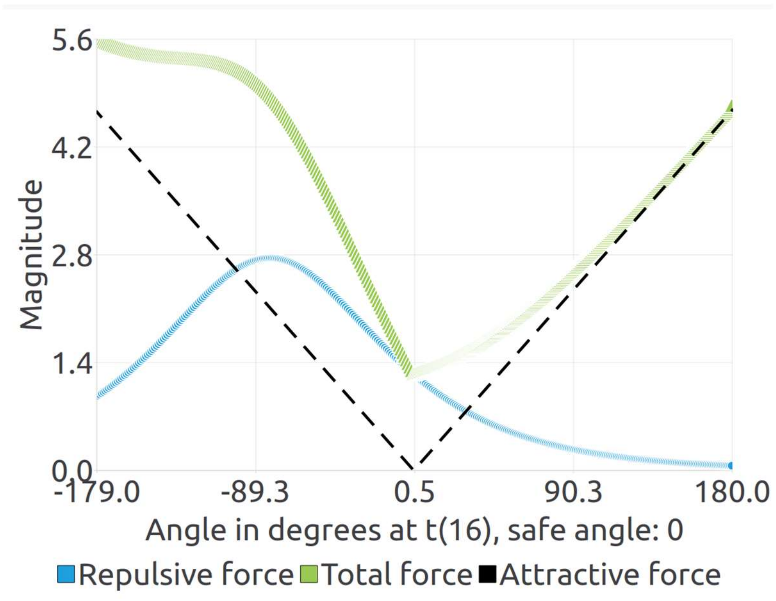

Figure 7: Repulsive, attractive, and total forces computed at the specified timestamp t_{16}.

The LiDAR sample obtained at the robot's final step, at the location near the wall is shown on Fig. 8.

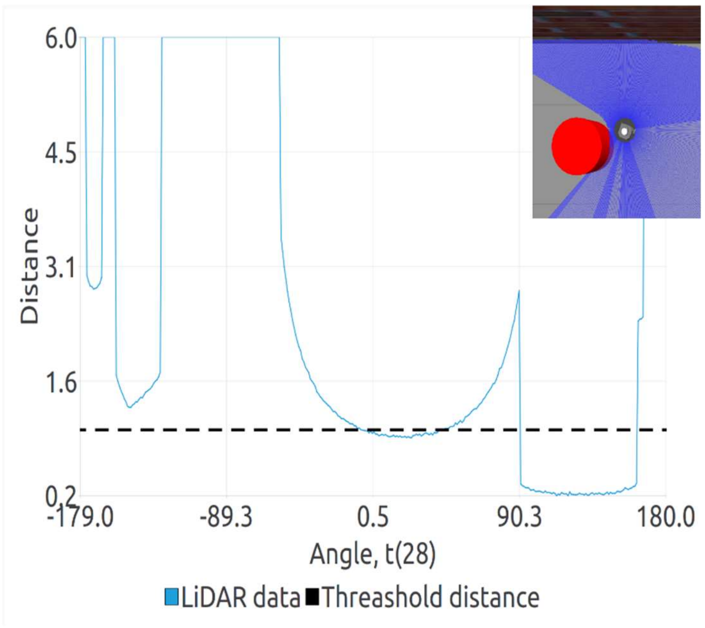

Figure 8: The 1D LiDAR sample at the t_{28} step. The solid line represents data a single complete LiDAR scan. The dashed curve depicts the threshold distance; objects detected closer than this threshold are classified as obstacles. The picture in the top-right corner demonstrates the current robot position in the simulator.

An analysis of the data presented in Fig. 8, reveals the presence of two continuous regions located below the threshold line, approximately within the angular ranges of [-10°, 60°] and [90°, 170°]. Subsequently, it is expected that the APFM should detect two obstacles. The first detected obstacle is a wall, as indicated by the geometric shape observed in the LiDAR sample. Specifically, the obstacle displays a smooth, circular-like shape—a signature feature typically associated with walls due to their uniform and continuous surfaces.

The second region suggests the existence of another distinct obstacle, whose angular location suggests it is positioned behind the robot. This conclusion is supported by the simulation environment visualization in the top-right corner of Figure 8.

Fig. 9 illustrates the force field structure at a subsequent stage. The repulsive force exhibits two prominent peaks near 0° and 150°, indicating multiple obstacles along the forward path. The attractive force maintains its canonical V-shaped profile, while the total force shows a minimum shifted to approximately −36°.

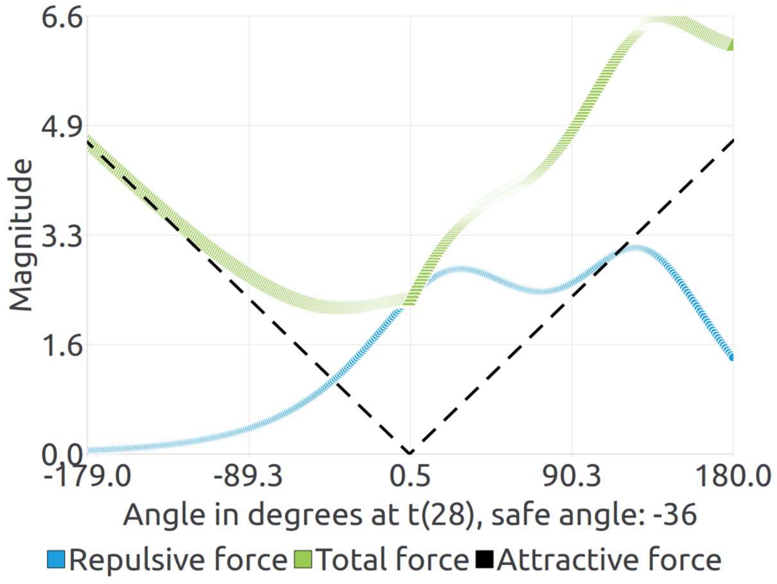

Figure 9: Repulsive, attractive, and total forces computed at the specified timestamp t_{28}.

This deviation indicates that the robot must adjust its heading by -36° to safely circumvent the obstacles. The resulting total force distribution underscores the dynamic balance between avoiding obstacles and moving toward the target, enabling robust and adaptive navigation. However, since the robot has already reached its target location, the computed safe direction is disregarded. This behavior reflects the algorithm's design principle of prioritizing target completion over continued navigation adjustments based on force computations once the goal has been attained.

Figure 10 displays the robot's navigation trajectory, as visualized in RViz, providing a spatial overview of the path taken during execution.

The RViz visualization demonstrates the operational efficacy of the implemented navigation system by highlighting three key elements of the obstacle avoidance process. Green vectors indicate real-time directional corrections where the calculated safe navigation angle deviates from the target trajectory due to detected obstacles, reflecting the system's dynamic response to environmental constraints. Red markers represent LiDAR-measured obstacle positions, forming a point cloud that quantifies the robot's perceptual field and correlates with the repulsive potential field generated by our APF implementation. The traversed path shows smooth deviations when encountering obstacles, confirming the proper application of repulsive forces. This visualization provides empirical validation of the algorithm's ability to maintain forward progress while executing collision-free path modifications.

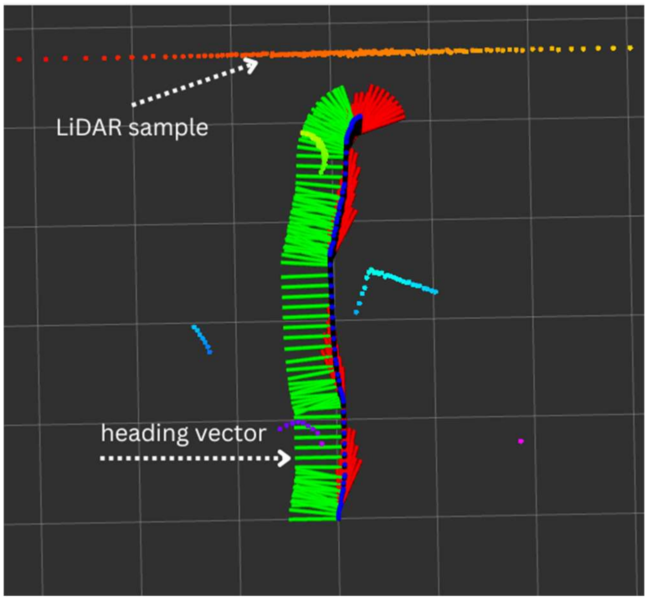

Figure 10: Traversed path visualization in RViz.

The results affirm the real-world applicability of the Logistic APFM modification in dynamic environments. Notably, this modification produced the smoothest path compared to three other variants.

3.2. Computational Efficiency

The computational performance of Hyperbolic Secant and Gaussian repulsive force models was evaluated through systematic benchmarking under controlled virtualized conditions. The host system featured an AMD Ryzen 7 3700X desktop processor (8C/16T) with 32GB DDR4-3200 MHz RAM and Samsung 970 EVO Plus 500 GB NVMe SSD, running Ubuntu 24.04.1 x64, using gcc 13.3.0.

Since the C++ standard library does not natively support the $\operatorname{sech}(x)$ function, two implementations of were used: one based on the equation (10) and another faster one based on the equation (11).

$$sech(x) = \frac{1}{\cosh(x)} \tag{10}$$

$$sech(x) = \frac{2}{e^x + e^{-x}}, \qquad with\ e^{-x} = \frac{1}{e^x} \tag{11}$$

Each benchmark run consisted of 1,000 iterations, with 10 million evaluations per iteration to ensure consistent timing resolution. The logistic function was calculated with $\sigma = 1, \mu = 0$. Execution times for each variant are presented in Table 2.

Table 2
Computational performance comparison (10^7 evaluations over 10^3 runs)

Function	Median time (s)	StdDev (s)
Gauss	0.05429	0.00035
Logistic (equation 10)	0.08429	0.00045
Logistic (equation 11)	0.06962	0.00028

Results show that the Gaussian calculation is the fastest baseline. The standard Logistic (equation 10) is about 55% slower than the Gaussian, while the Logistic (equation 11) implementation reduces this overhead to approximately 28% slower. These findings highlight the performance trade-offs when choosing between accuracy and computational efficiency in logistic function evaluations.

Conclusion

The Logistic APFM represents a notable advancement in obstacle avoidance algorithms, offering distinct benefits in path planning quality. The inherent characteristics of the logistic function—particularly its smooth gradient transitions and broader, more gradual peaks compared to other evaluated modifications—enable more natural and fluid navigation behavior. This results in visibly smoother trajectories with fewer abrupt corrections, especially in environments featuring sharp-edged obstacles or complex geometries. The method's increased sensitivity to nearby obstacles facilitates earlier and more gradual course adjustments, effectively reducing unnecessary path oscillations. These improvements stem directly from the mathematical formulation, without reliance on additional systems or sensors. While the approach demonstrates limitations in computational efficiency, it consistently yields the smoothest path among the tested variants, making it well-suited for low-speed robotic platforms.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Debnath, D., Vanegas, F., Sandino, J., Hawary, A. F., & Gonzalez, F. (2024). A Re-view of UAV Path-Planning Algorithms and Obstacle Avoidance Methods for Re-mote Sensing Applications. Remote Sensing, 16(21), 4019. https://doi.org/10.3390/rs16214019.

[2] Berizka I, Path planning and obstacle avoidance methods for autonomous mobile robots, ISSN 2224-087X. Electronics and information technologies. 2024. Issue 28. P. 123–142, https://doi.org/10.30970/eli.28.11.

[3] Katona, K.; Neamah, H.A.; Korondi, P. Obstacle Avoidance and Path Planning Methods for Autonomous Navigation of Mobile Robot. Sensors 2024, 24, 3573. https://doi.org/10.3390/s24113573.

[4] Merei, A., Mcheick, H., Ghaddar, A., & Rebaine, D. (2025). A Survey on Obstacle Detection and Avoidance Methods for UAVs. Drones, 9(3), 203.

[5] Khatib O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. The International Journal of Robotics Research. 1986;5(1):90-98. https://doi.org/10.1177/027836498600500106.

[6] Xiaojing Fan, Yinjing Guo, Hui Liu, Bowen Wei, Wenhong Lyu. (2020 Apr). Improved Artificial Potential Field Method Applied for AUV Path Planning. Mathematical Problems in Engineering. Mathematical Problems in Engineering. [Online]. Available: https://doi.org/10.1155/2020/6523158.

[7] Jang-Ho Cho, Dong-Sung Pae, Myo-Taeg Lim, Tae-Koo Kang. (2018 Aug). A Real-Time Obstacle Avoidance Method for Autonomous Vehicles Using an Obstacle-Dependent Gaussian Potential Field. Journal of Advanced Transportation. [Online]. Available: https://doi.org/10.1155/2018/5041401J.

[8] Berizka, I. and Karbovnyk, I. 2024. MATHEMATICAL MODEL OF MODIFIED REAL-TIME OBSTACLE AVOIDANCE METHOD BASED ON LAPLACE ARTIFICIAL POTENTIAL FIELD. Applied Problems of Computer Science, Security and Mathematics. 3 (Sep. 2024), 12–22, https://apcssm.vnu.edu.ua/index.php/Journalone/article/view/123

[9] Berizka I, Karbovnyk I, Computational Evaluation of Laplace Artificial Potential Field Methods for Real-Time Obstacle Avoidance in Gazebo, ISSN: 2524-0382 (print), 2707-0069 (online). 2025.

Advances in Cyber-Physical Systems (ACPS). Volume 10, Number 1. P. 1-9, https://doi.org/10.23939/acps2025.01.001 .

[10] I. Berizka, I. Karbovnyk, "Simulation-Based Evaluation of Hyperbolic Secant Potential Field for Real-Time Obstacle Avoidance", to be published in Ukrainian Journal of Information Technology, accepted for publication, June 2025.

[11] Elkady, Ayssam, Sobh, Tarek, Robotics Middleware: A Comprehensive Literature Survey and Attribute-Based Bibliography, Journal of Robotics, 2012, 959013, 15 pages, 2012. https://doi.org/10.1155/2012/959013 .

[12] Pablo Estefo, Jocelyn Simmonds, Romain Robbes, Johan Fabry, The Robot Operating System: Package reuse and community dynamics, Journal of Systems and Software, Volume 151, 2019, Pages 226-242, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2019.02.024 .

[13] Mayoral-Vilches, V., Pinzger, M., Rass, S., Dieber, B., & Gil-Uriarte, E. (2020). Can ros be used securely in industry? red teaming ros-industrial. arXiv preprint arXiv:2009.08211.

[14] DeMarinis, N., Tellex, S., Kemerlis, V. P., Konidaris, G., & Fonseca, R. (2019, May). Scanning the internet for ros: A view of security in robotics research. In 2019 International Conference on Robotics and Automation (ICRA) (pp. 8514-8521). IEEE.