Transactions on Energy Systems and Engineering Applications

Article

Adaptive stochastic gradient descent with least angle regression enhanced navigation: intelligent path planning in cluttered environments for autonomous robots

Abhishek Thakur[1,*], Subhranil Das[2], Sudhansu Kumar Mishra[1] and Subrat Kumar Swain[1]

[1] Department of EEE, Birla Institute of Technology, Mesra, Ranchi, India.
[2] School of Computer Science, UPES, Dehradun, India.
* Correspondence: abhishekthakur9396@gmail.com

Received: 17 December 2023; Accepted: 09 June 2025; Published: 15 September 2025

Abstract: In the dynamic realm of Autonomous Mobile Robots (AMRs), ensuring smooth navigation among obstacles is critical, especially as they become increasingly integral to industries such as manufacturing and transportation. Recent advances have introduced several learning models to aid in obstacle avoidance, but many face computational challenges. This research introduces the Adaptive Stochastic Gradient Descent with Least Angle Regression (ASGD-LARS) algorithm, specifically designed to enhance the navigation of AMRs. By carefully considering obstacle orientations, it facilitates quicker decision-making for direction changes. When compared with well-established algorithms like KNN, XG Boost, Naive Bayes, and Logistic Regression, ASGD-LARS consistently performs better in terms of accuracy, computational efficiency, and reliability. This study lays the foundation for the deployment of smarter and more efficient AMRs across diverse industries.

1. Introduction

In the evolving realm of robotics, navigating cluttered spaces presents both a challenge and an opportunity for innovation. As industries move towards automation, the need for Autonomous Mobile Robots (AMRs) that can efficiently navigate diverse environments becomes paramount. This article delves into these critical facets, providing a holistic perspective on AMR navigation. The ability of AMRs to traverse cluttered environments is crucial, particularly in industrial and urban settings. These spaces are often unpredictable, with static and dynamic obstacles that can impede the robot's path. Understanding the intricacies of such environments and the complexities they introduce to navigation is fundamental to improving AMR performance. Beyond mere navigation, the route an AMR takes plays a significant role

How to cite this article: Thakur, Abhishek; Das, Subhranil; Mishra, Sudhansu; Swain, Subrat. Adaptive stochastic gradient descent with least angle regression enhanced navigation: intelligent path planning in cluttered environments for autonomous robots. *Transactions on Energy Systems and Engineering Applications*, 6(2): 602, 2025. DOI:10.32397/tesea.vol6.n2.602

in its efficiency and safety. Intelligent path planning goes beyond conventional navigation, ensuring that the robot not only reaches its destination but does so in the most optimal manner. This involves assessing multiple routes, predicting potential challenges, and choosing a path that balances efficiency with safety. At the heart of effective navigation in cluttered spaces is the robot's ability to identify and avoid obstacles. This necessitates a robust algorithm that can quickly process environmental data, identify potential hazards, and adjust the robot's path in real-time. The effectiveness of this algorithm directly impacts the AMR's safety and efficiency. AMRs stand at the forefront of this discussion, representing a fusion of advanced robotics, artificial intelligence, and sensor technology. Their autonomous nature requires them to make split-second decisions, adapt to changing environments, and operate without human intervention. As such, improving their navigation capabilities in cluttered spaces is of paramount importance. In this paper, we embark on an in-depth exploration of these topics, introducing the Adaptive Stochastic Gradient Descent with Least Angle Regression (ASGD-LARS) algorithm and evaluating its performance against established algorithms such as, as KNN, XG Boost, Naive Bayes, and Logistic Regression in the realm of AMR navigation. The forthcoming sections will also draw upon existing research and models, underscoring the contributions and limitations present in the current landscape of AMR navigation and obstacle avoidance. Through a meticulous exploration of the ASGD-LARS model, this paper endeavors to cast light on potential pathways toward enhancing AMR navigation, fostering a focus on future research trajectories in this domain.

1.1. Related Works

Navigating cluttered environments using Autonomous Mobile Robots (AMRs) has become a pressing concern in various sectors, including industry, transportation, and manufacturing. A key challenge in these spaces is the efficient avoidance of obstacles, especially in diverse structured, unstructured, and hybrid surroundings. Despite the development of various proficient learning models for this purpose, many still grapple with computational efficiency issues. Obstacle avoidance has emerged as a pivotal research area in robotics. This technology underpins a multitude of applications ranging from automation in industries, such as precise positioning of goods, to surveillance, assisting passengers in airports, and several service industry operations [1–3]. Modern AI techniques have been employed to enhance trajectory tracking amidst obstacles, a theme extensively discussed in this paper. Kumar and Parhi [4] pioneered an AI-infused regression navigational controller designed for the navigation of single and multiple humanoids in congested spaces. This method leverages the Genetic Algorithm (GA) to pinpoint the controller's optimal parameters, ensuring smooth navigation amidst multiple humanoids. However, the use of an intermediate Advancing Angle (AA) restricts the humanoid's range. On a similar note, Zafar et al. blended Grey Wolf Optimization (GWO) with the Artificial Potential Field (APF) to steer mobile robots to their destination safely [5]. This dual-phase approach first ascertains a Focus Region (FR) devoid of obstacles, then utilizes GWO algorithm to determine the shortest path by reducing the Artificial Potential Field (APF) in a cluttered environment. Nonetheless, GWO's convergence rate was deemed subpar for pinpointing optimal paths in densely cluttered zones. Sezer introduced the Follow the Gap Method (FGM), which traces a global plan of successive waypoints [6]. Alongside this, a Look Ahead Distance (LAD) function optimized tracking and facilitated obstacle avoidance. A limitation, however, was the oversight of obstacle orientation. Das and Mishra recently postulated a novel Machine Learning algorithm named Adaptive Stochastic Gradient Descent Linear Regression (ASGDLR). This model guides AMR around obstacles based on their orientation, demonstrating efficacy across varying AMR speeds and ensuring efficient obstacle circumvention [7]. Furthermore, Mary et al. outlined an array of sensors that capture real-time velocity and distance metrics through IoT [8]. In the domain of dynamic obstacle avoidance, Kashyap and Parhi integrated Regression Analysis (RA), Cell Decomposition (CD), and Whale Optimization Algorithm (WOA) to compute optimal steering angles in intricate environments [9]. Another noteworthy research

introduced a dynamic recurrent neuro-fuzzy approach to mobile robot obstacle avoidance, emphasizing short memory utilization [10]. Both studies signify advancements in dynamic obstacle detection and avoidance. Path planning is integral to robotics, acting in tandem with obstacle avoidance, especially in cluttered settings [11]. Recent innovations like Durakli and Nabiyev's method based on Bezier Curves [12] and nature-inspired optimization techniques, such as the Reformative Bat Algorithm (RBA) [13], whale optimization [14], Owl Search Algorithm (OWA) [15], and Particle Swarm Optimization (PSO) [16], have showcased their potential in mobile robot applications, ensuring efficient path planning with minimal computational overhead. Table 1 provides a comparative overview of the merits of the proposed optimization algorithms vis-à-vis existing state-of-the-art methodologies.

Table 1. Comparative Analysis of Path Planning Algorithms for AMRs.

Path Planning Algorithms	Approach	Online/Offline	Key Features/Comments
Improved A* [17]	Search-Based	Offline	1. Prunes redundant neighboring nodes. 2. Minimizes memory usage. 3. Evaluates optimal nodes until final path determination.
DFPA [18]	–	Online	1. Utilizes Voronoi points for prioritizing nodes. 2. Extracts obstacle edges for enhanced avoidance.
APF [19]	Potential Field	Online	1. Designates a Focus Region for unobstructed movements. 2. Finds the shortest path by minimizing the APF value.
GVF [20]	–	Online	1. Tracks target paths amidst unforeseen disturbances. 2. Accounts for nonholonomic constraints. 3. Features low path error convergence criteria.
NFC [21]	Fuzzy-Based	Offline	1. Taps into precise navigational control benefits. 2. Fuses neural networks for decision-making.
ASGDLR [7]	Regression-Based	Online	1. Employs a fully-trained adaptive algorithm for optimal decisions. 2. Skillfully avoids obstacles in intricate settings given sufficient training.

1.2. Contributions to existing research

To address the shortcomings of the previously discussed algorithms, this paper presents the Adaptive Stochastic Gradient Descent–Least Angle Regression (ASGD-LARS) method. This approach aims to classify the movements of AMRs into three distinct binary categories. Six datasets, representing low, medium, and high-speed AMRs, have been leveraged to validate the proposed algorithm, capturing real-time distances and velocities using three IoT sensors. Key contributions of this paper include:

i. While the study in [8] lacked depth in movement categorization, this research distinctly classifies AMR movements into three binary groups: Left Turn (LT) vs Right Turn (RT), No Turn (NT) vs Right Turn (RT), and No Turn (NT) vs Left Turn (LT) using the ASGD-LARS model.
ii. Our algorithm delves into six unique speeds, selecting two pairs from each speed category (low, medium, and high). LARS method coefficients are continuously updated for each distance segment covered by the AMR via the Adaptive Stochastic Gradient Descent (SGD) optimization.
iii. Additionally, the accuracy of the Region of Operating (ROC) curves is demonstrated for all six speeds to assess the method, creating a real-time clustered environment for its evaluation.
iv. We have compared the performance of the ASGD-LARS model with established AI algorithms, including K-Nearest Neighbour (K-NN), XG-Boost, Naïve Bayes, and Logistic Regression (LR). This comparison encompasses the percentage of obstacle avoidance and the representation of all pertinent ROC curves.
v. Our approach was subjected to rigorous testing in three diverse clustered settings, where its efficiency was benchmarked against leading methodologies like A*, VFH, FLC, and ASGDLR in terms of path lengths and computational duration.

The structure of this paper is organized as follows: Section 2 delves into the essential hardware requirements. Section 3 elucidates the proposed methodology, followed by problem definition and an exploration of the mathematical formulations for obstacle avoidance in Section 4. Section 5 provides insights into two distinct simulation tasks, one involving a single obstacle and the other multiple obstacles. Section 6 highlights the limitations and future directions of research. Finally, research findings and conclusions are drawn in Section 7.

2. Hardware Prerequisites

In our research, we employed distance-sensitive Ultrasonic Sensors (US) and velocity-sensitive Infrared Sensors (IR) strategically placed at the front of the AMR's chassis. These sensors relay real-time data to the NodeMCU controller, which in turn modulates the velocities of the AMR's two wheels based on proximity to obstacles. Additionally, each wheel of the AMR is equipped with power driver modules (L298N DC Motor) to ensure optimal power delivery. For stability, a front wheel is positioned at the front end of the setup.

The IR sensors, situated on the encoder discs, are tasked with gauging the velocity of each wheel. In contrast, the US sensor, centered on the AMR chassis, measures real-time distances. Signals pertaining to distance and velocity from these sensors are processed by the NodeMCU controller, which then adjusts the left and right wheel velocities for effective obstacle avoidance, steering the AMR either to the right or the left as needed.

Furthermore, our system harnesses Internet of Things (IoT) sensors to transmit real-time data packets for visualization purposes. A comprehensive depiction of the AMR's hardware configuration can be found in Figure 1. A laptop, equipped with Arduino Software, interfaces with the AMR via data transfer channels. On the AMR chassis, Ultrasonic and Infrared Sensors measure real-time distances and velocities, respectively. These sensors feed data to the NodeMCU ESP8266 controller, which in turn manages the AMR's motion based on obstacle proximity.

Figure 2 explains the schematics for drawing the connections present in the NodeMCU ESP 12E Controller. In this figure, four terminals of DC motor driver module are connected to four points of the controller via electronic wires which are further connected to the wheels of AMR.

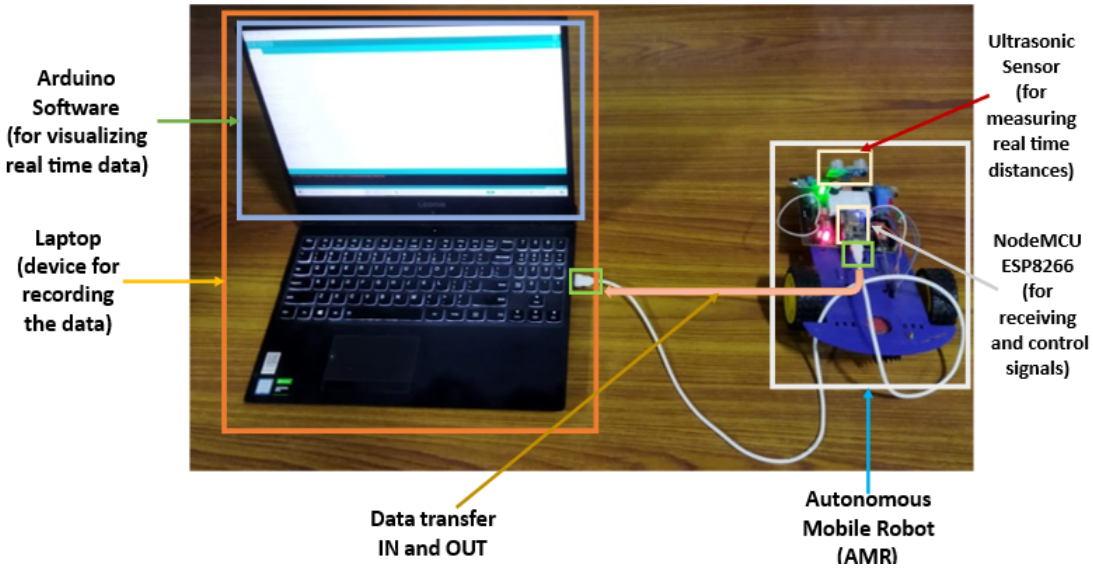

Figure 1. Hardware Setup for AMR.

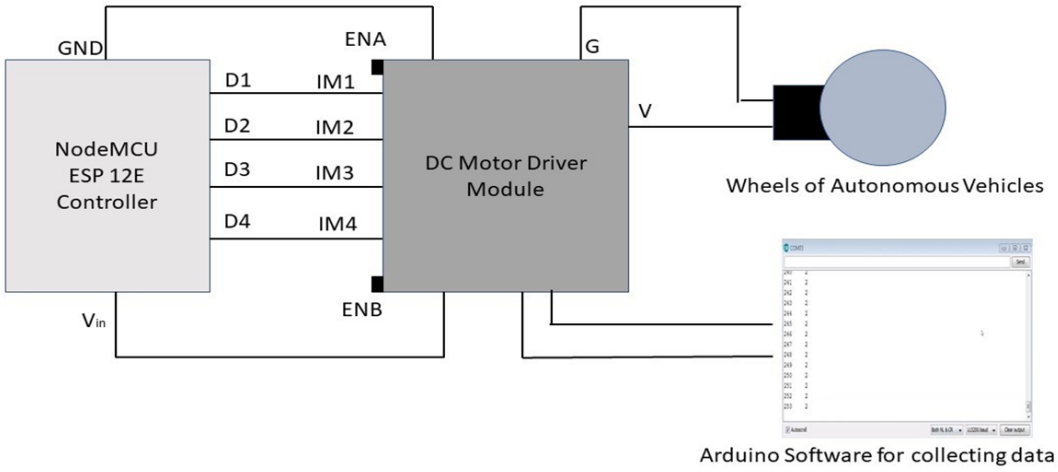

Figure 2. Schematic diagram for connections in NodeMCU ESP 12E Controller.

2.1. Experimental Set up

The experiment was designed to rigorously test the performance of the Adaptive Stochastic Gradient Descent with Least Angle Regression (ASGD-LARS) algorithm in three different cluttered environments. This setup has allowed for collecting reliable data which ensures that the results are not influenced by unpredictable environmental variables. The area of 390×150 unit has been considered for experimentation purposes with a 30×30 cm grid layout for facilitating positioning and measurement of obstacle avoidance performance. The layout of the structured grid layout ensures the standardized path planning in different testing conditions. For each environment, the AMR obstacle avoidance has been carefully designed to challenge, path planning, and its adaptability. The three different types of the cluttered environments have been explained below.

i. **Type-1 (large obstacle at 45°Tilt):** In this set up, a single large obstacle has been placed at a 45-degree angle. In this configuration, AMR has been configured to detect the obstacle around an inclined structure where the real-world obstacles like fallen objects or slanted barriers have been considered. The angle-wise adjustment of the ASGD-LARS algorithm plays an important role while optimizing the trajectory of AMR. By adjusting its movement dynamically based on the obstacle's tilt, the algorithm enabled the AMR to make precise directional changes which ensure the smoother and more efficient path around the obstacle.

ii. **Type-2 (Equidistant obstacles allowing multiple optimal paths):** This environment was designed to test path efficiency, where multiple paths existed for the AMR to navigate. The AMR had to dynamically select the shortest and safest route using the ASGD-LARS algorithm. Since multiple pathways were available, the decision-making process relied on the algorithm's ability to balance speed, efficiency, and safety. The distance-wise component of the algorithm helped in prioritizing paths based on real-time sensor feedback, reducing computational load while ensuring efficient obstacle avoidance. This allowed the AMR to choose the most optimal route with minimal deviation from the desired trajectory.

iii. **Type-3 (Horizontally and Vertically aligned obstacles with two tilted ones):** This environment contains obstacles both horizontally and vertically aligned, with the additional challenge of two uniquely tilted obstacles. This required the AMR to frequently adjust its path and orientation while navigating through constrained spaces. The presence of tilted obstacles introduced additional complexity, demanding a higher level of adaptability from the AMR's navigation system. The integration of both angle-wise and distance-wise optimization ensured that the AMR could efficiently adapt to rapidly changing obstacle configurations. By continuously analyzing the spatial arrangement of obstacles, the algorithm facilitated real-time adjustments, leading to a more reliable and effective path-planning strategy. Table 2 explains this environment schema of AMR.

Table 2. Specification of AMR.

S.No.	Features	Dimensions
1	Length	20 cm
2	Breadth	15 cm
3	Ground Clearance	4 cm
4	Wheel Radius	2.5 cm
5	Wheel Thickness	1.2 cm
6	Weight	1 Kg
7	Moment of Inertia	0.5 Kg/cm^2
8	Battery Voltage	7.4 V
9	Battery Capacity	6000 mAH
10	Running Capacity	1 hr

3. Proposed Methodology

3.1. Motivation

In recent years, crafting efficient navigational technologies for automated vehicles amidst static and dynamic obstacles has emerged as a significant hurdle across various sectors, including industrial, manufacturing, and service industries. The adoption of Artificial Intelligence (AI)-based navigation

technologies in the automobile sector, addressing distinct challenges, has been particularly prominent. It plays a vital role in navigating Automated Mobile Robots (AMRs) by making complex decisions regarding their direction and velocity to adeptly avoid collisions [22]. Globally, car manufacturers like Tesla Motors, Hyundai, and Google have engineered advanced AI-based navigation systems into their vehicles. These systems incorporate numerous sophisticated features into Advanced Driving Assistance Systems (ADAS) to ensure safe trajectory tracking, even when obstacles are present on the road. For instance, Tesla's Autopilot software comes integrated with a myriad of collision prevention features to augment vehicle safety, a strategy also pursued by other manufacturers developing their own collision avoidance systems. These systems heavily depend on lasers, sensors, and cameras to preemptively detect obstacles, alongside existing internal technology associated with vehicle navigation. Nevertheless, cars fitted with such systems have been armed with low-level AI technology, which has occasionally resulted in accidents [23]. To address the limitations of low-level AI implementations, this research proposes a recent AI-based regression technique, proficiently enabling the avoidance of any type of static obstacle by leveraging information on distance and velocity from various sensors.

3.2. Advantages of Angle-Wise over Distance-Wise Approach

The ASGD-LARS algorithm significantly benefits from its angle-wise and distance-wise considerations, making it superior in real-time navigation and computational efficiency. For angle-wise adjustment, precise directional changes are enabled based on obstacle alignment. Also, it reduces unnecessary turns and has been reduced which lead to smoother navigation. The prediction in the trajectory has been improved by considering relative positions of obstacles. For distance-wise adjustment, adaptive speed control can be ensured which depends on obstacle proximity. Collision probability has been reduced by making dynamic real-time adjustments. Computational efficiency has been increased by prioritizing obstacles within the decision boundary. By integrating both strategies, ASGD-LARS achieves higher obstacle avoidance accuracy with lower computational overhead which ensure efficient, safe, and adaptive navigation for AMRs in complex environments.

3.3. Problem Formulation

The Autonomous Mobile Robot (AMR) relies on data procured from infrared (IR) and ultrasonic (US) sensors for effective collision avoidance. This sensor data informs the robot's movement, which can be quantified through the following mathematical expressions:

$$Yi = f(\text{VLW}, D), \tag{1}$$

$$YJ = f(\text{VRW}, D). \tag{2}$$

Here, f represents a non-linear function influenced by wheel speed and proximity to obstacles.

The limitation of this approach is that the velocity of the obstacle, i.e., $V_o = 0$, which means that the difference in the position of the obstacles has been considered as zero. As a result, the environments on which the proposed approach would work are applicable to static environments only. On the other hand, if the positions of the obstacles have changed, then $V_o \neq 0$, which means that there is a certain velocity of the obstacles. The proposed approach will be accordingly modified to account for dynamic environments.

Specifically, Yi denotes the output based on the right wheel's velocity (VRW) and distance (D) to potential obstructions. Conversely, YJ is determined by the left wheel's velocity (VLW) and the same distance (D). We've categorized the velocity into three classifications for clarity:

i. Low speed: 0 to 10 cm/s
ii. Medium speed: 10.1 to 15 cm/s

iii. High speed: 15.1 to 20 cm/s

The term D measures the distance from the furthest detectable obstacle, whereas Dc specifies the essential distance to that obstacle. $Dc1$ illustrates the space between the AMR's decision-making threshold and the obstacle. Static obstructions have dimensions characterized by W_j (width) and B_j (length), where j can vary from 1 to P. Positioned centrally on the AMR chassis, the US sensor calculates the Euclidean distance to any static barrier.

As the robot approaches the vital decision boundary, it either chooses a left or right trajectory. This decision is influenced by the speed parameters: low, medium, or high. Rightward rotations of the AMR are represented through angles of $\theta_{L1}, \theta_{L2}, \theta_{M1}, \theta_{M2}, \theta_{H1}, \theta_{H2}$ in radians. The opposite, leftward movements, are signified by $-\theta_{L1}, -\theta_{L2}, -\theta_{M1}, -\theta_{M2}, -\theta_{H1}, -\theta_{H2}$ in radians. These angles correspond to the six speed variants, as depicted in Fig. **??**.

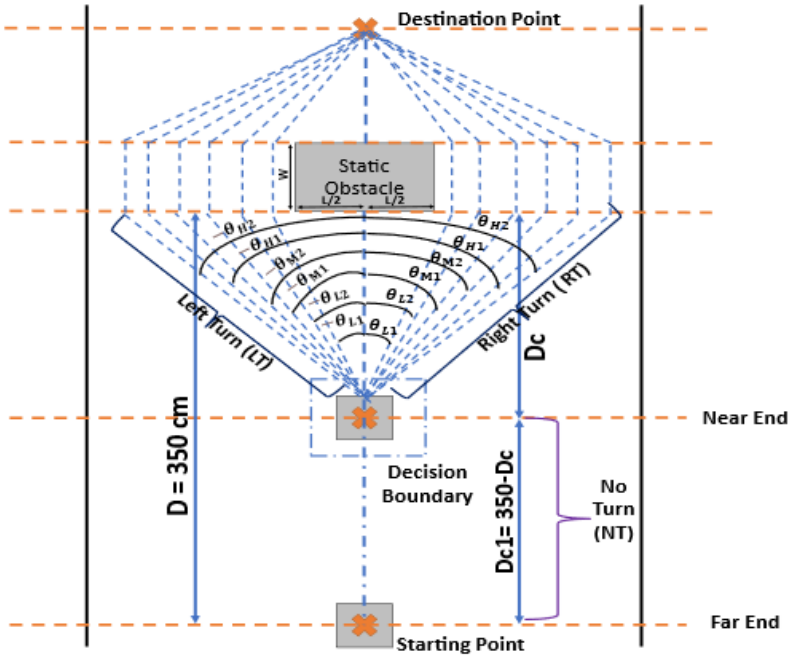

Figure 3. Mathematical Model for AMR's collision evasion.

Figure 3 provides a visual representation of the AMR's collision avoidance model. It highlights the robot's varying turn angles at different speeds when faced with an obstruction, emphasizing the integral roles of speed, angle, and distance in decision-making. For further clarity,

For low speed, i.e., 2.77 cm/s, the angle θ_{L1} ranges from:

$$\theta_{L1} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{L}{8}\right)}{D_c}\right). \tag{3}$$

For low speed, i.e., 3.77 cm/s, the angle θ_{L2} ranges from:

$$\theta_{L2} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{L}{4}\right)}{D_c}\right). \tag{4}$$

For medium speed, i.e., 11 cm/s, the angle θ_{M1} ranges from:

$$\theta_{M1} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{3L}{8}\right)}{D_c}\right). \tag{5}$$

For medium speed, i.e., 13.5 cm/s, the angle θ_{M2} ranges from:

$$\theta_{M2} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{L}{2}\right)}{D_c}\right). \tag{6}$$

For high speed, i.e., 19.4 cm/s, the angle θ_{H1} ranges from:

$$\theta_{H1} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{5L}{8}\right)}{D_c}\right). \tag{7}$$

For high speed, i.e., 23.8 cm/s, the angle θ_{H2} ranges from:

$$\theta_{H2} \sim \tan^{-1}\left(\frac{\frac{L}{2} + \max\left(0, \frac{5L}{8}\right)}{D_c}\right). \tag{8}$$

Table 3 offers a breakdown of the mathematical notations used in the ASGD-LARS model, detailing the speed, angle, values, and distances considered during training.

Table 3. Summary of mathematical notations used for ASGD-LARS model.

Speed	Angle	Value	Distance Covered	Angle	Value
LS1	θ_{L1}	[0°, 10°]	$\sqrt{D_c^2 + \left(\frac{5L}{8}\right)^2}$	$-\theta_{L1}$	[0°, 10°]
LS2	θ_{L2}	[10°, 17°]	$\sqrt{D_c^2 + \left(\frac{3L}{4}\right)^2}$	$-\theta_{L2}$	[10°, 17°]
MS1	θ_{M1}	[17°, 22°]	$\sqrt{D_c^2 + \left(\frac{7L}{8}\right)^2}$	$-\theta_{M1}$	[17°, 22°]
MS2	θ_{M2}	[22°, 30°]	$\sqrt{D_c^2 + L^2}$	$-\theta_{M2}$	[22°, 30°]
HS1	θ_{H1}	[30°, 37°]	$\sqrt{D_c^2 + \left(\frac{9L}{8}\right)^2}$	$-\theta_{H1}$	[30°, 37°]
HS2	θ_{H2}	[37°, 42°]	$\sqrt{D_c^2 + \left(\frac{3L}{4}\right)^2}$	$-\theta_{H2}$	[37°, 42°]

3.4. Algorithm Design

The design of the AMR's navigation system to adeptly maneuver around static obstacles can be broken down into a sequence of four core stages, as illustrated in Figure 4.

i. **Data Collection and Initial Processing:** At the outset, the Ultrasonic Sensor (US) gauges the distance to static obstructions while the Infrared Sensor (IR) captures wheel velocities. Both sensors, situated at the forefront of the AMR, are critical for successful navigation. The data harvested from the US and IRs is then processed using Arduino software. Once processed, it's stored in the CPU in a ".dat" file format.

ii. **Wheel Velocity Refinement and Strategy Formation:** The NodeMCU Controller, equipped with the collected data, fine-tunes the wheel velocities using the ASGD-LARS model. As the AMR nears an obstruction, its navigation strategy is formed by referencing datasets associated with different wheel speeds: low, medium, and high. This phase wraps up once the robot successfully navigates past the obstacle.

iii. **Trajectory Quantification and Duration Measurement:** The next phase focuses on metrics. It computes the exact path followed by the AMR and the time taken to maneuver around each obstacle.

iv. **Recalibration and Optimal Path Formation:** If another obstacle is detected, the AMR recalibrates. It harnesses fresh data from the US and IRs to plot the most efficient navigation course.

For a deeper understanding of the algorithmic components, Table 4 provides a comprehensive overview of the mathematical symbols vital to the ASGD-LARS algorithm.

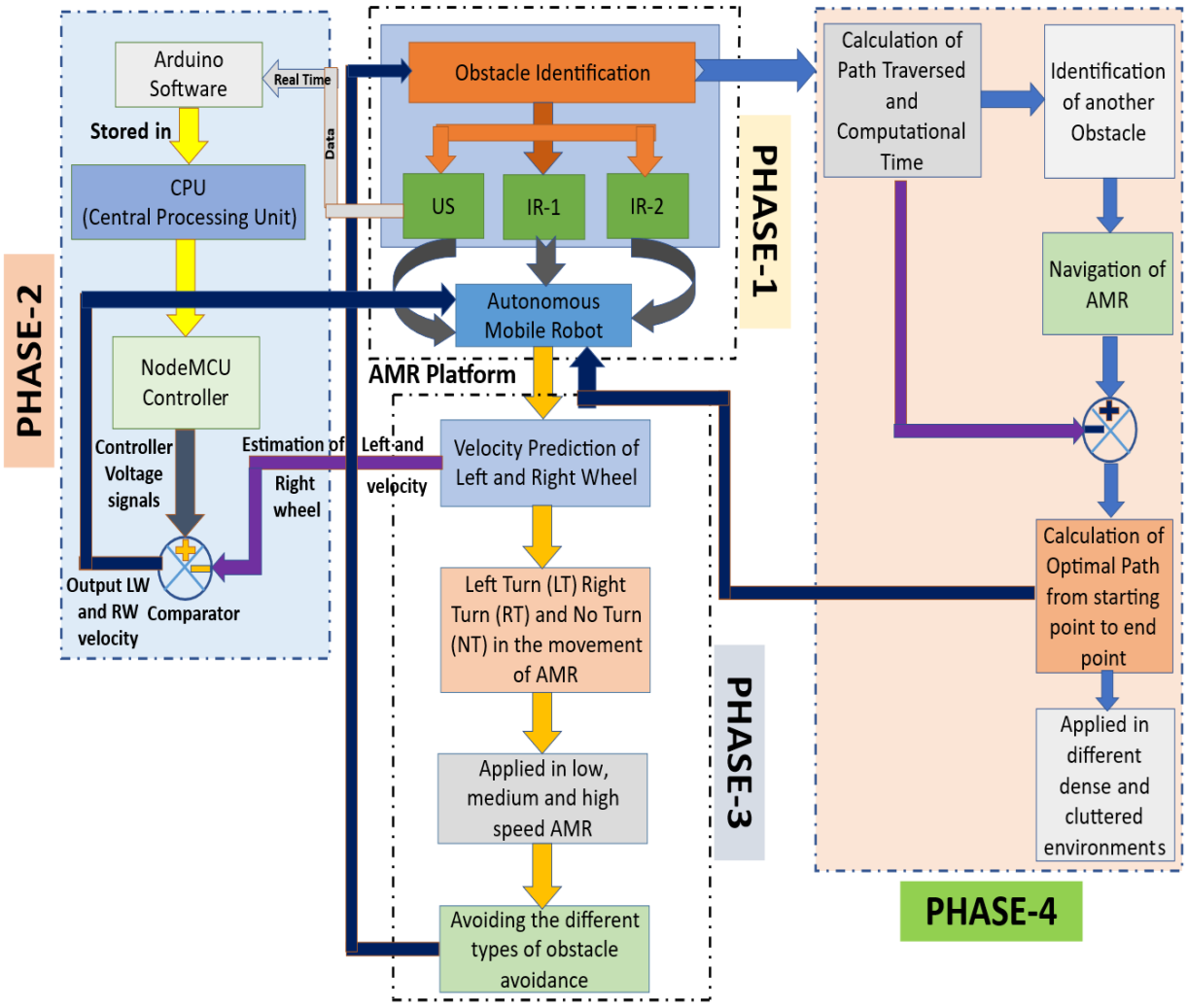

Figure 4. Outline of different phases of obstacle avoidance and path planning.

Table 4. Summary of mathematical notations used for ASGD-LARS model.

Symbols	Description
$\pm\theta_{L1}, \pm\theta_{L2}$	Steering angle for low speeds when making left or right turn
$\pm\theta_{M1}, \pm\theta_{M2}$	Steering angle for medium speeds when making left or right turn
$\pm\theta_{H1}, \pm\theta_{H2}$	Steering angle for high speeds when making left or right turn
$V_{\mathrm{LW}}, V_{\mathrm{RW}}$	Velocity of left and right wheel
$w_{K(L1)}, w_{K(L2)}$	Weights of the intercepts of low speeds
$w_{K(M1)}, w_{K(M2)}$	Weights of the intercepts of medium speeds
$w_{K(H1)}, w_{K(H2)}$	Weights of the intercepts of high speeds
D_c	Critical distance from the obstacle
D_{c1}	Distance from the decision boundary from the obstacle
$\hat{\alpha}$	Measured value of distance vector from the US
$S(\hat{\alpha}_K)$	Total square loss of the measured distances from the observed distances
$\mathrm{sign}(\hat{c}_j)$	Value of the co-relation current step vector

4. Mathematical Approach for Obstacle Avoidance

4.1. Formulation of ASGD-LARS Model

Specific assumptions are established to accurately simulate the proposed algorithm for an AMR, which are as follows:

i. **Symmetrical Differential Dynamics:** The AMR's differential dynamics are hypothesized to be symmetrically aligned with the line of sight (LoS) at the midpoint of the chassis. This implies both the left and right wheels share equivalent rotational speed and power, allowing for a straight trajectory.
ii. **Linear Trajectory:** The AMR is postulated to maintain a linear trajectory until encountering a static obstacle. This suggests a path free from turns or bends, optimizing obstacle detection and evasion.
iii. **Consistent Velocity:** The AMR is believed to travel at a consistent speed along a set track, excluding the possibility of acceleration. This standardization simplifies the processes of obstacle detection and path strategizing.
iv. **Even Terrain:** The terrain is considered even, ensuring minimal static friction between the AMR's wheels and the ground. This results in a seamless motion, ensuring optimal obstacle detection without the risk of slippage.
v. **Standard Climatic Conditions:** The simulation operates under typical weather conditions, ensuring consistent algorithm performance in path planning and obstacle evasion.

Least Angle Regression (LARS) is a statistical method for analysing high-dimensional data, developed by Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani [24]. Serving as a resolution to the least squares challenges associated with a plethora of predictors, the LARS algorithm emphasizes an equiangular direction between the predictors exhibiting the highest correlation with the response during its every phase. The pseudo code of LARS algorithm has been shown in the below snippet.

Algorithm 1 Pseudocode for existing LARS algorithm.

INITIALIZE:
for each j from 1 to N **do**
  $\beta_j \leftarrow 0$
end for
Set residual $r \leftarrow y - X\beta$
BEGIN ALGORITHM:
while a predictor exists that is correlated with r **do**
  1. SET $x_j \leftarrow$ predictor most correlated with r
  2. DO
  Move β_j in direction of correlation of x_j with r
  Set $x_k \leftarrow$ predictor with highest correlation with current residual
  while x_k's correlation with residual $\neq x_j$'s **do**
    continue updating x_k
  end while
  END DO
  3. DO
  Move jointly in direction defined by x_j and x_k
  Set $x_l \leftarrow$ predictor with highest correlation with current residual
  while x_l's correlation $\neq x_j$ and x_k correlation **do**
    continue updating x_l
  end while
  END DO
  4. CONTINUE movement in joint direction of x_j, x_k, and x_l until all predictors are in the model
  5. REPEAT
  for each predictor j **do**
    Compute $\gamma =$ smallest correlation value among all predictors
    Update $\beta_j \leftarrow \beta_j + \gamma \cdot \text{sign}(c_j)$
  end for
  UNTIL β reaches full least squares solution
end while
END ALGORITHM

This study proposes an augmentation to the conventional LARS methodology by integrating the Bagging technique. This enhancement seeks to mitigate the model's prediction variance. Bagging (Bootstrap Aggregating) is a strategy that involves generating numerous bootstrap samples from the foundational data and fitting a unique model (in this scenario, the LARS model) to each [25]. The culmination of this process is an averaged prediction, derived from the collective models. The Bagging with LARS pseudo-code has been illustrated in the following snippet.

Algorithm 2 Pseudocode for Bagging with LARS algorithm.

INITIALIZE:
DEFINE B (number of bootstrap samples)
DEFINE n (size of each bootstrap sample)
CREATE an empty array `predictions_array` to store predictions
BEGIN ALGORITHM:
for i FROM 1 TO B **do**
  Draw a bootstrap sample of size n from original data (with replacement)
  Fit a LARS model to the bootstrap sample
  Store the LARS model predictions on original data in `predictions_array`
end for
COMPUTE final prediction as the average of predictions in `predictions_array`
END ALGORITHM

4.2. Adaptive Stochastic Gradient Descent (ASGD) Optimization

In the LARS regression, the weights corresponding to the coefficients for the low, medium, and high speeds of both the right and left wheels are dynamically updated after every distance the AMR traverses. This is achieved through the application of the Adaptive Stochastic Gradient Descent (ASGD) algorithm, as illustrated in Figure 5.

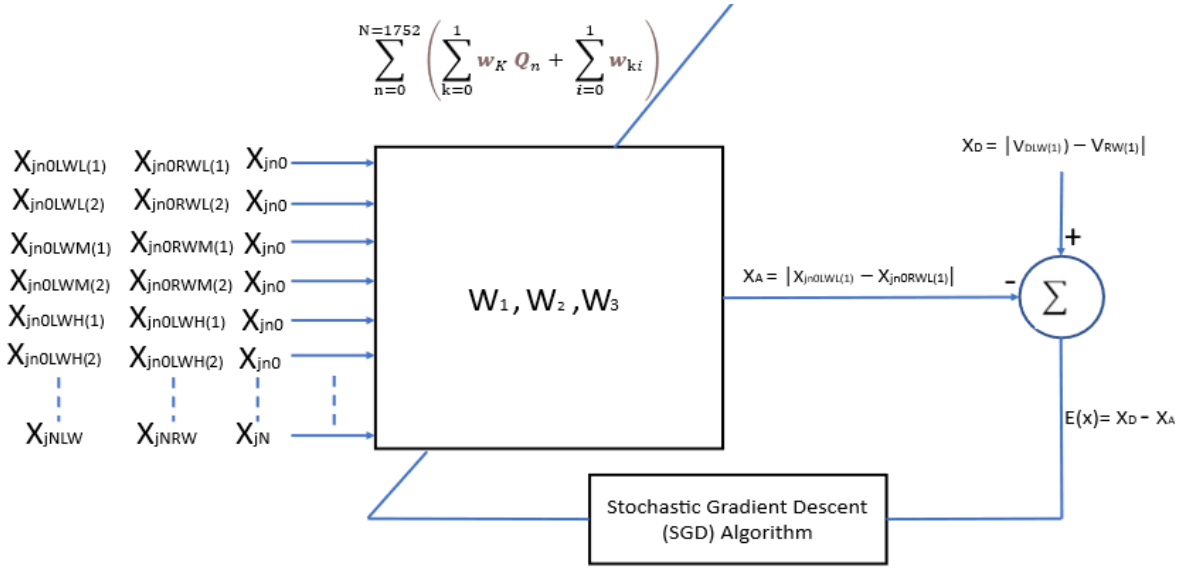

Figure 5. Updation of the weights of the coefficients of the proposed ASGD-LARS algorithm.

A closer look at Figure 5 reveals that the predictors for both the left and right wheels are determined after every centimeter the AMR covers. These predictors subsequently serve as input to the adaptive algorithm. Within this context, XA denotes the computed predictor, which represents the variance between the predictors of the two wheels. Conversely, XD signifies the intended predictor, equating to the actual estimated velocities for the left and right wheels. The difference between XD and XA results in $E(x)$, the error signal for the SGD optimization algorithm.

Iteratively, the coefficient weights of the LARS algorithm are fine-tuned until the desired value is achieved for each of the six speed categories. The unified weight vector is represented as:

$$Q_n = [\text{XjNLW} \quad \text{XjNRW} \quad \text{XjN}].$$

In the dataset employed, there are six distinct speed ranges categorized from low to high. Each of these speed categories contains 292 observations. The Bagging technique introduces 10 bootstrap samples for each speed category. Therefore, the dataset comprises 17,520 data points (292 observations $\times$ 10 bootstrap samples $\times$ 6 speed categories).

For the purposes of our study, training data is designated as 14,016 data points, which accounts for eighty percent of the dataset, while the test data is considered as the remaining 3,504 data points. The steps for the ASGD-LARS method are further detailed in the provided snippet below.

Algorithm 3 Advanced Stochastic Gradient Descent with Least Angle Regression (ASGD-LARS).

INPUT:
Dataset with 6 distinct speed categories
Each speed category contains 292 observations
Total data points: 17,520
Training data points: 14,016
Testing data points: 3,504
Learning Rate (LR): $\{\eta_i\}_{i=1}^{N}$
Scalar Function: $\varnothing$

PARAMETERS:
$[\![w_{ki}]\!]_1$, $[\![w_{ki}]\!]_2$, Y, α

INITIALIZATION:
Cost Function $J(w) = J([\![w_{ki}]\!]_1, [\![w_{ki}]\!]_2)$
$\mu^{(k+1)} = \mu^k + w_k \cdot \text{sign}(\hat{c}_j) \cdot \sum_{m\in[\text{LW,RW}],\ p\in[\text{L,R}]} X_{\text{imp}}$
Set $\mu^0 = 0$

TRAINING PROCESS:
Partition dataset: 80% training and 20% testing
for each speed category **do**
  Execute ASGD-LARS training procedure (as detailed previously)
  Update weights and compute cost
end for

TESTING PROCESS:
for each speed category in the testing set **do**
  Execute ASGD-LARS testing procedure (as detailed previously)
end for

OUTPUT:
Model performance metrics, weights, and predictions

5. Results and Discussion

The Autonomous Mobile Robot (AMR) was tested using the ASGD-LARS algorithm on a NodeMCU ESP8266 controller. The primary objective was to understand the robot's capability in navigating environments with obstructions of varying sizes and undisclosed positions. Python IDE was used for simulations while `MATLAB2022b` software was utilized for executing various path-finding algorithms in intricate settings. The computer setup seems adequately powerful for such testing. The computer specifications used for these tests boasted a 16GB RAM, a 1TB hard drive, ran on the Windows 11 Operating System, and was powered by an Intel® Core i7–9th Generation processor with a 2.60 GHz frequency.

5.1. Case 1: Avoidance of Single Obstacle

The AMR is tasked with avoiding a singular static obstacle as shown in Figure 6. The diagram displays a safety zone around the obstacle, limited to about 1 cm due to environmental factors like floor friction, weather, and model uncertainties such as voltage approximations and sensor data inconsistencies.

Figure 6. Formulation for No Turn Problem.

The data for low, medium, and high-speed conditions is collected, with 80% used for training and 20% for testing. A Confusion Matrix (CM), as depicted in Table 5, is generated to compute parameters like Classification Accuracy, Sensitivity, Specificity, and F1-Score.

Table 5. Dataset's CM.

Value Predicted	**LT (1) / RT (2)**	**NT (0)**
Value of Actual Scenario		
LT (1) / RT (2)	True Positive (TP)	False Positive (FP)
NT (0)	False Negative (FN)	True Negative (TN)

Table 5 illustrates a 2x2 matrix where TP and TN indicate correct classifications, and FN and FP point to incorrect classifications. Values of higher TP and TN indicate better prediction accuracy. The values of elements present in the CM reflect the values of CA, SENS, SPEC, PREC, REC, and F1-Score as represented mathematically as:

$$\text{SENS} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{3}$$

$$\text{SPEC} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \tag{4}$$

$$\text{PREC} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{5}$$

$$\text{CA} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}, \tag{6}$$

$$\text{F1-score} = 2 \cdot \frac{P \cdot R}{P + R}. \tag{7}$$

The values of the different parameters of the CM of the proposed ASGD-LARS algorithm have been compared with K-Nearest Neighbor (K-NN), Naïve Bayes (NB), XG-Boost, and Logistic Regression (LR), as mentioned in Table 6.

Three performance indices, namely Mean Absolute Error (MAE), Mean Square Error (MSE), and Lars Coefficient (LC), are used to evaluate the effectiveness of the suggested algorithm, as indicated below.

Table 6. Values of the CM parameters of all algorithms in different speed cases.

Parameters	Speed	K-NN	XG-Boost	Naïve Bayes	Logistic Regression	Proposed ASGD-LARS
Classification	2.77 cm/s	98.64	98.19	98.74	99.65	99.67
Accuracy		98.14	97.74	98.01	99.54	99.53
Sensitivity		99.34	99.17	99.11	100	100
Specificity		99.65	98.56	99.42	100	100
Precision		99.65	98.34	99.42	100	100
Recall		98.12	94.18	95.29	99.54	99.58
F1 score		1.9122	1.8236	1.8473	1.9711	1.9812
Classification	3.77 cm/s	97.72	96.12	96.74	98.97	98.98
Accuracy		97.26	98.34	96.21	98.69	98.72
Sensitivity		97.26	98.34	96.21	98.69	98.72
Specificity		99.65	98.88	99.54	100	100
Precision		99.65	98.12	99.52	100	100
Recall		97.67	97.45	95.29	98.69	98.83
F1 score		1.9342	1.8864	1.8643	1.9546	1.9552
Classification	11 cm/s	96.23	96.16	95.24	97.26	97.37
Accuracy		96.12	95.37	96.12	97.8	96.24
Sensitivity		96.12	95.37	96.12	97.8	96.24
Specificity		92.68	90.74	91.21	95.31	95.35
Precision		96.58	93.56	97.31	98.67	97.37
Recall		96.53	91.51	95.31	97.81	98.12
F1 score		0.9655	0.9712	0.9837	0.9762	0.9842
Classification	13.8 cm/s	94.31	94.12	95.12	96.23	96.34
Accuracy		95.32	93.35	94.52	96.46	96.12
Sensitivity		91.35	90.11	89.31	95.45	94.34
Specificity		97.46	93.77	97.43	98.64	96.78
Precision		95.42	92.26	94.31	96.46	97.77
Recall		95.42	92.26	94.31	96.46	97.77
F1 score		0.9548	0.8781	0.8838	0.9754	0.9635
Classification	16.67 cm/s	92.06	93.74	94.34	94.68	95.35
Accuracy		94.82	91.18	92.15	95.98	97.74
Sensitivity		89.82	87.67	86.42	91.17	93.38
Specificity		95.48	90.75	92.31	97.28	96.67
Precision		91.98	93.37	94.66	95.98	95.28
Recall		90.53	91.51	93.71	98.13	98.12
F1 score		0.9442	0.8772	0.8347	0.9662	0.8912
Classification	19.8 cm/s	90.97	93.74	91.22	94.52	94.67
Accuracy		93.65	91.51	88.32	94.68	95.51
Sensitivity		92.68	91.41	88.82	95.31	95.51
Specificity		96.87	93.12	91.77	98.01	95.51
Precision		92.67	91.87	93.21	97.28	96.67
Recall		91.65	91.53	91.11	96.52	96.68
F1 score		0.9608	0.9169	0.9332	0.9684	0.9612

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |X_{\mathrm{jmpr}} - X_{\mathrm{jmpal}}|, \tag{14}$$

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(X_{\text{jmpr}} - X_{\text{jmpal}})^2, \tag{15}$$

where, X_{jmpr}, X_{jmpal}, and N denote the predicted velocity, actual velocity of both wheels, and the total number of data points respectively.

The effectiveness of an Autonomous Mobile Robot (AMR) in navigating obstacles is evaluated using performance metrics including Mean Absolute Error (MAE), Mean Square Error (MSE), and the LARS Coefficient (LC). When the AMR operated at speeds of 2.77 cm/s and 3.77 cm/s, it demonstrates a high obstacle avoidance percentage, consistently achieving over 98%. However, as the robot's speed increases, the coefficients and MAE, MSE values also show a corresponding rise, indicating a higher risk of collisions at such speeds.

In the testing phase, as the speed increases from 3.77 cm/s to 13.8 cm/s, the LC shows a slight increase, reaching a value of 0.02. This increment in coefficient values is associated with a decrease in the obstacle avoidance percentage. At even higher speeds, specifically 16.67 cm/s and 19.4 cm/s, the LARS coefficients climb further to 0.03. Such increases in coefficients resulted in more pronounced values for MAE and MSE, underscoring the potential for the AMR to collide with obstacles when operating at high speeds. To visualize these findings, Figure 7 provides a depiction of obstacle avoidance percentages for different algorithms across three speed categories: low, medium, and high speeds.

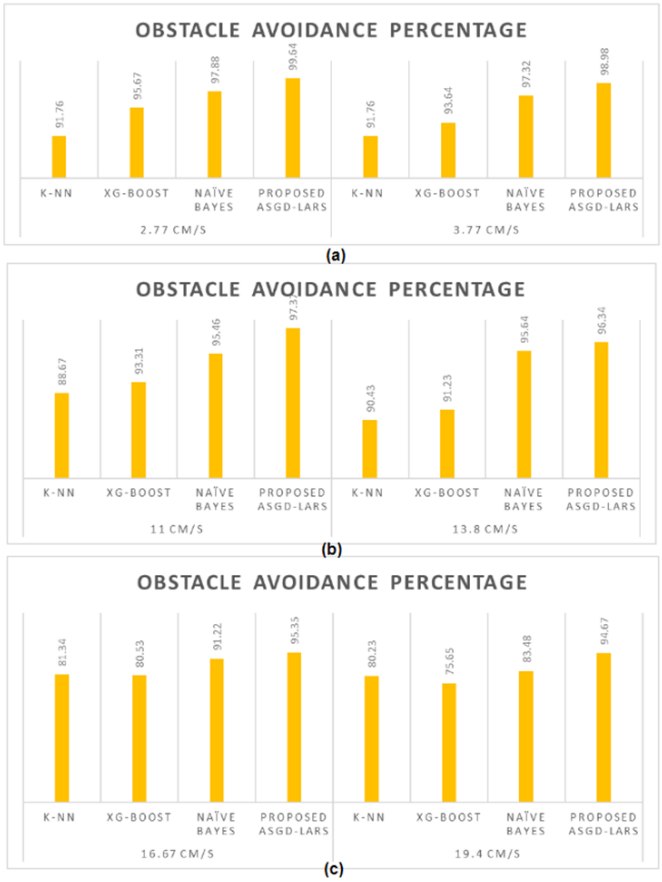

Figure 7. Obstacle avoidance percentages for all algorithms (a) Low Speed (b) Medium Speed (c) High Speed.

In a bid to create a realistic testing environment for the AMR, a 390×150 unit arena is constructed. Within this arena, a uniform 30 x 30 cm grid is laid out, with static obstacles placed strategically at specific points. The starting point of the AMR's route is denoted by a green star, while the end or goal point is marked by a red star. This setup can be observed in Figure 8, which showcases the AMR detecting distant obstacles, recognizing decision boundaries at closer proximities, making turns to avoid obstacles, adhering to designated trajectories, actively navigating around obstacles, and finally reaching the end point.

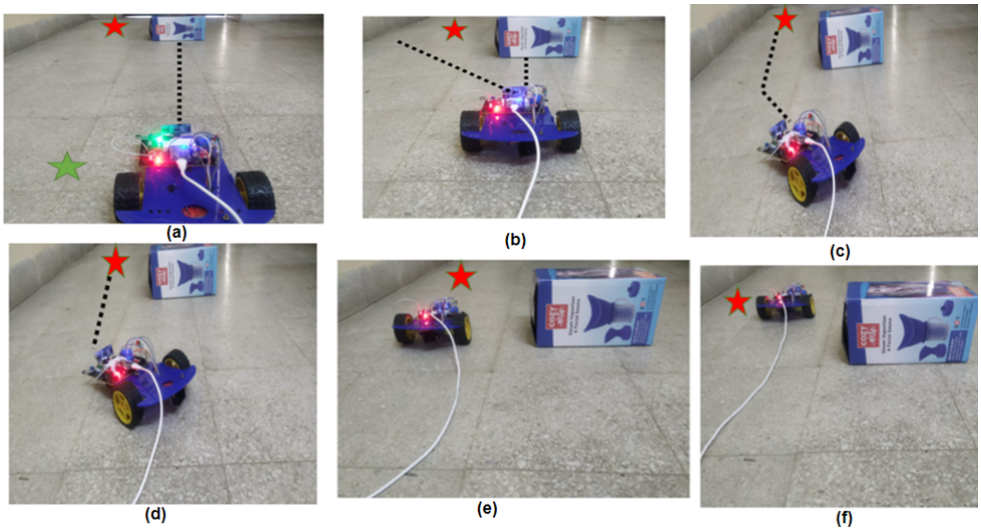

Figure 8. Demonstration of the experiment (a) Detection of obstacle at far end (b) Decision boundary at near end (c) AMR takes turn while avoiding the obstacle (d) AMR follows the trajectory line (e) AMR avoids the obstacle (f) Goal point is reached.

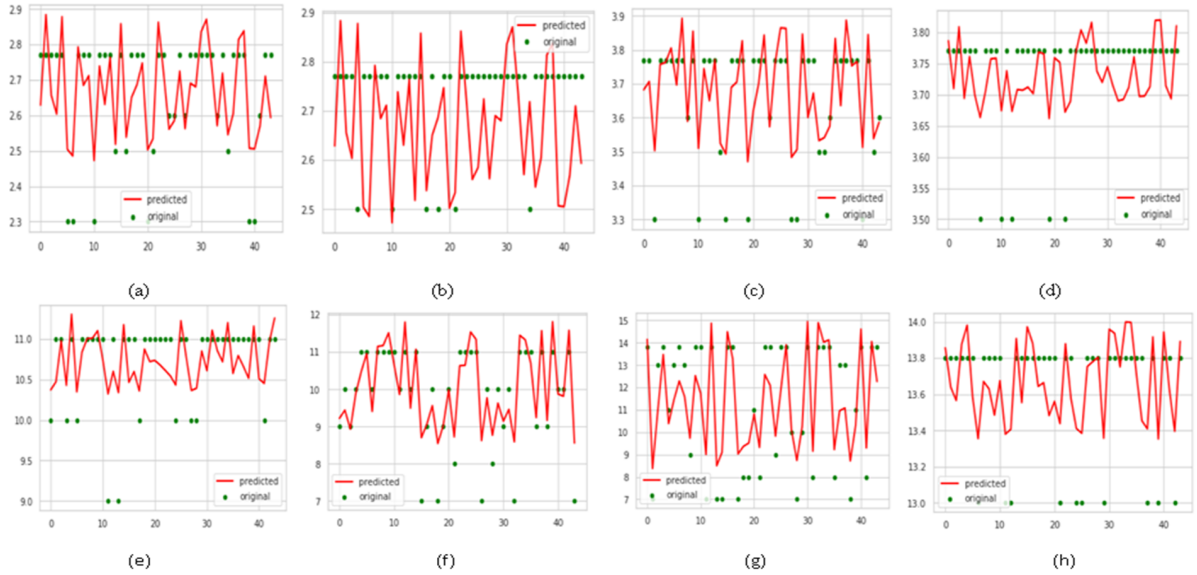

Figure 9. Simulation graphs for the left and right wheel velocities for testing dataset (a)-(d) low speed (e)-(h) medium speed.

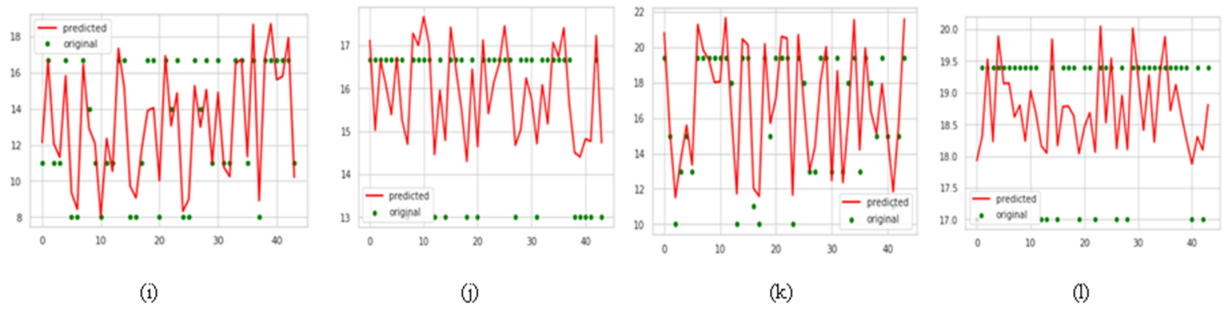

Figure 10. Simulation graphs for the left and right wheel velocities for testing dataset at high speed.

In addition to the aforementioned tests, wheel velocities of the AMR are also analyzed under varying speeds. Figures 9 and 10 displays simulation graphs for these velocities. Within these graphs, the green dots represent the computed values, whereas the continuous red line reflects the predicted data points. The graph provides insights into the AMR's wheel velocities at low, medium, and high speeds.

The study assesses algorithms including A*, FLC, VFH, ASGDLR, and the new ASGD-LARS, examining parameters like Computational Time, Path Length, and AMR Speed. At 2.77 cm/s, A* had a computational time of 3.04 minutes and covered 505.34 cm. As AMR speed rises, A*'s time decreases but its path length grows. In contrast, FLC achieves shorter paths (495.65 cm at 2.77 cm/s) with less time (2.98 minutes). VFH and ASGDLR improve on these, with VFH covering 470.42 cm in 2.83 minutes and ASGDLR, 461.52 cm in 2.77 minutes. Yet, ASGD-LARS consistently tops the chart, taking 2.72 minutes and covering 452.34 cm at 2.77 cm/s. As AMR speed changes, all algorithms display reduced computational times, but paths vary in length. ASGD-LARS's consistent performance highlights the importance of efficient algorithms in AMR path planning.

5.2. Case 2: Multi-tasking in Environments with Multiple Obstacles

Different experiments are conducted where the behavior of the proposed ASGD-LARS algorithm is exploited by investigating its performance in avoiding various sizes of obstacles in its path. The performance of this algorithm is now compared with A*, VFH, FLC, ASGDLR as four conventional path planning algorithms. Computational time and traversed path are calculated for six distinct speeds of the AMR. Three cluttered environments are created on the MATLAB 2022a platform according to varying levels of complexity. In these environments, the AMR identifies the optimal path from start to end. Every square in the environment spans 30×30 cm. These scenarios facilitate the simulation of multiple path planning algorithms, as depicted in Figure 11.

From Figure 11, static obstacles of four different sizes populate three unique environments. These are set up to evaluate the proposed ASGD-LARS algorithm against the four traditional methods. Each environment labels the starting point as SP and the endpoint as EP. In Type-1, obstacles challenge the efficiency of algorithms, with a notable large obstacle tilted at 45°. Type-2 spaces out obstacles equidistantly, paving the way for multiple optimal paths. Type-3, on the other hand, arranges obstacles both horizontally and vertically, supplemented by two uniquely tilted obstacles. All the algorithms undergo 20 simulation runs, with their average path lengths and computational durations presented in Tables 7 to 9.

Table 7 illustrates the A* algorithm navigating the most extended paths in Type-1 due to interference from the 3D obstacle. Employing fuzzy rules, the FLC algorithm trims paths by approximately 13% across all settings. VFH further reduces the journey by around 9.5% owing to diminished oscillations. While the ASGDLR algorithm curtails path lengths slightly more than VFH, ASGD-LARS stands out. It exploits the gaps between obstacles and the AMR to achieve the most concise paths with a mean computational time

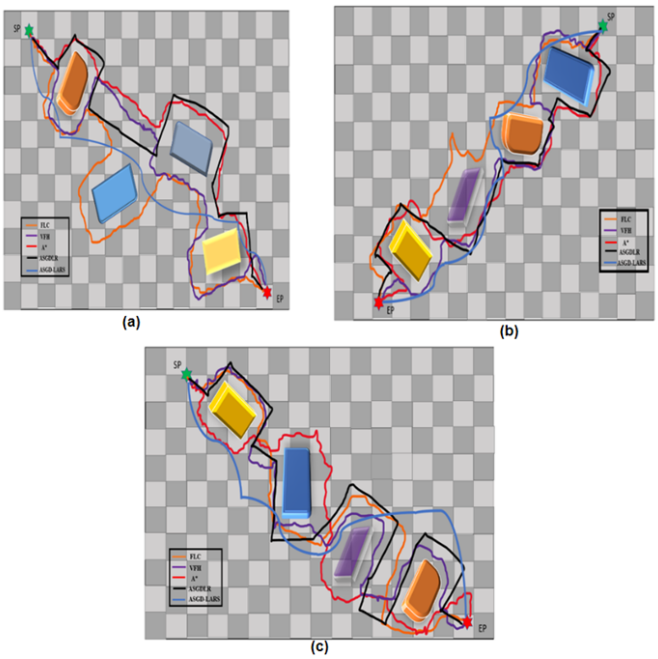

Figure 11. Simulation results for FLC, VFH, A*, ASGDLR, ASGD-LARS path planning algorithms on different clustered environments (a) Type-1 (b) Type-2 (c) Type-3.

of 3.62 minutes. According to Table 8, A* consistently charts the longest routes, notably within Type-3 environments at 13.8 cm/s. ASGDLR further shortens the paths, and ASGD-LARS emerges superior, producing the briefest path lengths with computational times of 0.83 and 0.79 minutes at 11 and 13 cm/s, highlighting exceptional obstacle navigation capabilities.

The ASGD-LARS algorithm holds promise for optimizing neural network weights. This research methodology offers insights for refining neural network weights, amplifying performance across various use-cases. Its applicability extends to challenging landscapes, narrower pathways, and slower autonomous systems. Table 9 specifically examines higher AMR speeds (16.67 cm/s and 19.4 cm/s) across the three environments. Results show that ASGD-LARS persistently outperforms other algorithms in both computational time and path length, underscoring its effectiveness. Notably, even in denser environments and higher speeds, the ASGD-LARS algorithm maintains its edge, indicating its promise for real-world applications and its adaptability for more complex scenarios, including UAVs navigating in 3D spaces.

Table 7. Simulation Results for different path planning algorithms for low speed AMR.

Exp. No.	Speed of AMR	Cluttered Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
31	2.77	Type-1	A*	5.28	877.75
			FLC	4.71	783.92
			VFH	4.04	672.67
			ASGDLR	3.44	573.35
			ASGD-LARS	3.07	510.39

Continued on next page

Table 7 – *Continued from previous page*

Exp. No.	Speed of AMR	Cluttered Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
			A*	5.47	910.23
			FLC	4.83	803.64
32	2.77	Type-2	VFH	3.83	637.67
			ASGDLR	3.68	612.35
			ASGD-LARS	3.52	585.63
			A*	5.68	944.26
			FLC	4.68	779.39
33	2.77	Type-3	VFH	4.21	700.78
			ASGDLR	3.96	658.71
			ASGD-LARS	3.67	610.31
			A*	3.94	891.78
			FLC	3.41	772.68
34	3.77	Type-1	VFH	2.91	658.89
			ASGDLR	2.69	610.46
			ASGD-LARS	2.27	513.54
			A*	3.98	901.22
			FLC	3.58	810.25
35	3.77	Type-2	VFH	3.19	722.25
			ASGDLR	3.01	681.35
			ASGD-LARS	2.35	531.93
			A*	4.21	952.38
			FLC	3.66	828.54
36	3.77	Type-3	VFH	3.14	710.47
			ASGDLR	2.96	671.68
			ASGD-LARS	2.64	598.72
			A*	3.96	907.16
			FLC	3.21	749.32

Table 8. Simulation Results for different path planning algorithms for medium speed AMR.

Exp. No.	Speed of AMR	Dense Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
37	11	Type-1	A*	1.31	870.73
			FLC	1.17	775.86

Continued on next page

Table 8 – *Continued from previous page*

Exp. No.	Speed of AMR	Dense Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
			VFH	1.00	665.84
			ASGDLR	0.91	603.31
			ASGD-LARS	0.79	524.43
			A*	1.39	922.81
			FLC	1.25	825.96
38	11	Type-2	VFH	1.07	712.74
			ASGDLR	1.01	672.27
			ASGD-LARS	0.83	550.72
			A*	1.45	957.84
			FLC	1.24	822.31
39	11	Type-3	VFH	1.07	710.72
			ASGDLR	0.99	659.26
			ASGD-LARS	0.89	590.75
			A*	1.06	882.61
			FLC	0.93	775.88
40	13.8	Type-1	VFH	0.81	670.31
			ASGDLR	0.72	603.27
			ASGD-LARS	0.63	522.31
			A*	1.09	910.31
			FLC	0.96	802.78
41	13.8	Type-2	VFH	0.87	725.58
			ASGDLR	0.83	691.77
			ASGD-LARS	0.71	581.82
			A*	1.16	961.89
			FLC	1.01	829.74
42	13.8	Type-3	VFH	0.88	733.94
			ASGDLR	0.82	680.51
			ASGD-LARS	0.75	622.85

Table 9. Simulation Results for different path planning algorithms for high speed AMR.

Exp. No.	Speed of AMR	Dense Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
43	16.67	Type-1	A*	0.88	881.53

Continued on next page

Table 9 – *Continued from previous page*

Exp. No.	Speed of AMR	Dense Environment	Path Planning Algorithms	Average Computational Time for 20 runs (min)	Average Path Length Traversed for 20 runs (cm)
			FLC	0.77	770.45
			VFH	0.67	671.84
			ASGDLR	0.61	610.73
			ASGD-LARS	0.53	531.37
			A*	0.91	911.95
			FLC	0.86	868.46
44	16.67	Type-2	VFH	0.72	722.58
			ASGDLR	0.68	681.39
			ASGD-LARS	0.54	549.28
			A*	0.96	963.42
			FLC	0.85	852.68
45	16.67	Type-3	VFH	0.74	746.81
			ASGDLR	0.68	683.36
			ASGD-LARS	0.61	600.41
			A*	0.76	891.59
			FLC	0.66	771.27
46	19.4	Type-1	VFH	0.57	673.94
			ASGDLR	0.52	612.98
			ASGD-LARS	0.43	511.84
			A*	0.78	915.68
			FLC	0.69	802.78
47	19.4	Type-2	VFH	0.63	733.92
			ASGDLR	0.59	690.57
			ASGD-LARS	0.51	591.76
			A*	0.83	967.72
			FLC	0.73	852.36
48	19.4	Type-3	VFH	0.64	752.84
			ASGDLR	0.61	701.32
			ASGD-LARS	0.56	652.83

The simulation results are significantly improved due to the following key novel contributions of the ASGD-LARS approach. For Adaptive coefficient update mechanism, ASGD-LARS continuously refines its decision-making parameters through an adaptive stochastic gradient descent process which ensures real-time adjustments that improve obstacle avoidance. By incorporating LARS, the algorithm effectively prioritizes the most relevant features influencing AMR movement which lead to a more efficient path selection process. The combined angle-wise and distance-wise strategy enhances the robot's ability to navigate complex environments by dynamically adjusting steering angles and velocities based on real-time sensor inputs. The iterative optimization of regression coefficients minimizes processing power requirements that enable faster decision-making with improved accuracy. ASGD-LARS ensures better

trajectory prediction with a higher success rate in obstacle avoidance when compared to conventional ML algorithms such as KNN and Logistic Regression. As a result, there is 12% reduction in computational time and a 5-10% improvement in obstacle avoidance accuracy in real-world environments. Experimental results indicate that ASGD-LARS outperforms traditional methods in terms of computational efficiency and path planning accuracy. The primary benefits of ASGD-LARS include reduced computational load by updating coefficients dynamically where the algorithm minimizes the processing power required for real-time path planning. The algorithm can effectively handle different cluttered environments, adjusting navigation paths accordingly. Optimized path planning reduces unnecessary movements which leads to more efficient battery usage in AMRs.

6. Limitations and Future Directions

The ASGD-LARS approach has a significant improvement over conventional methods by combining stochastic gradient descent with least angle regression for avoiding obstacles in real time. Unlike conventional methods such as Genetic Algorithm (GA), Grey Wolf Optimization (GWO), and Artificial Potential Field (APF), the proposed approach dynamically updates its regression coefficients to optimize the robot's path efficiently. Another key novelty of ASGD-LARS is its ability to adaptively fine-tune the decision-making process by considering both obstacle distance and orientation. Experimental results indicate that ASGD-LARS outperforms traditional methods in terms of computational efficiency and path planning accuracy. The primary benefits of ASGD-LARS include reduced computational load by achieving 5-10% higher accuracy than conventional ML-based models in avoiding obstacles. The algorithm can effectively handle different cluttered environments, adjusting navigation paths accordingly. While ASGD-LARS demonstrates superior performance in cluttered environments, further advancements can be explored.

For the future directions, the proposed approach can be refined to more dynamic environments by changing the position of the obstacles which is defined in the Eqn. 1 and 2 where the velocity of the obstacle has been considered as 0. As a result, new equations will be framed which hold good in the changing environments. The modified algorithm could be tested in the different scenarios while considering the moving obstacles in front of AMR. Also, the proposed model could be extended on 3 dimensional aerial autonomous systems which have number of moving obstacles at large.

7. Conclusions

In this study, we present a newly formulated ML model, ASGD-LARS, designed to guide AMR through single and multiple obstacles across three dense environments. We assessed the model's efficacy using Mean Absolute Error (MAE), Mean Square Error (MSE), and Lars Coefficient (LC), with the outcomes being promising. Moreover, we update the LARS coefficients using the ASGD optimization method, effectively predicting the probabilities for both wheels. Relative to other established ML algorithms, our model demonstrates enhanced performance. Simulations indicate that our model suggests an optimal path for the AMR, requiring the least distance and time, especially when compared with other leading algorithms like A*, FLC, VFH, and ASGDLR under various speed conditions.

Funding: This research received no external funding.

Author contributions: Abhishek Thakur has contributed to the conception and design. Material preparation, data collection and analysis were performed by Abhishek Thakur and Subhranil Das. The first draft of the manuscript was written by Abhishek Thakur and all authors commented on previous versions of the

manuscript. Sudhansu Kumar Mishra, Abhishek Thakur and Subrat Kumar Swain read and approved the final manuscript.

Disclosure statement: The authors declare that they have no competing interest among themselves or others for this work.

References

[1] Yudie Hu, Yuqi Wang, Kaixiong Hu, and Weidong Li. Adaptive obstacle avoidance in path planning of collaborative robots for dynamic manufacturing. *Journal of Intelligent Manufacturing*, 34(2):789–807, August 2021.

[2] Francisco J. Perez-Grau, J. Ramiro Martinez-de Dios, Julio L. Paneque, J. Joaquin Acevedo, Arturo Torres-González, Antidio Viguria, Juan R. Astorga, and Anibal Ollero. Introducing autonomous aerial robots in industrial manufacturing. *Journal of Manufacturing Systems*, 60:312–324, July 2021.

[3] Mingzhang Pan, Jing Li, Xiuze Yang, Shuo Wang, Lei Pan, Tiecheng Su, Yuke Wang, Qiye Yang, and Ke Liang. Collision risk assessment and automatic obstacle avoidance strategy for teleoperation robots. *Computers amp; Industrial Engineering*, 169:108275, July 2022.

[4] Priyadarshi Biplab Kumar and Dayal R. Parhi. Intelligent hybridization of regression technique with genetic algorithm for navigation of humanoids in complex environments. *Robotica*, 38(4):565–581, June 2019.

[5] Mohd. Nayab Zafar, J. C. Mohanta, and Anupam Keshari. Gwo-potential field method for mobile robot path planning and navigation control. *Arabian Journal for Science and Engineering*, 46(8):8087–8104, June 2021.

[6] Volkan Sezer. An optimized path tracking approach considering obstacle avoidance and comfort. *Journal of Intelligent amp; Robotic Systems*, 105(1), May 2022.

[7] Subhranil Das and Sudhansu Kumar Mishra. A machine learning approach for collision avoidance and path planning of mobile robot under dense and cluttered environments. *Computers and Electrical Engineering*, 103:108376, October 2022.

[8] Mary B. Alatise and Gerhard P. Hancke. A review on challenges of autonomous mobile robot and sensor fusion methods. *IEEE Access*, 8:39830–39846, 2020.

[9] Abhishek Kumar Kashyap and Dayal R. Parhi. Implementation of intelligent navigational techniques for inter-collision avoidance of multiple humanoid robots in complex environment. *Applied Soft Computing*, 124:109001, July 2022.

[10] Chaochao Chen and Paul Richardson. Mobile robot obstacle avoidance using short memory: a dynamic recurrent neuro-fuzzy approach. *Transactions of the Institute of Measurement and Control*, 34(2–3):148–164, July 2010.

[11] José Ricardo Sánchez-Ibáñez, Carlos J. Pérez-del Pulgar, and Alfonso García-Cerezo. Path planning for autonomous mobile robots: A review. *Sensors*, 21(23):7898, November 2021.

[12] Zafer Duraklı and Vasif Nabiyev. A new approach based on bezier curves to solve path planning problems for mobile robots. *Journal of Computational Science*, 58:101540, February 2022.

[13] Gongfeng Xin, Lei Shi, Guanxu Long, Weigang Pan, Yiming Li, and Jicun Xu. Mobile robot path planning with reformative bat algorithm. *PLOS ONE*, 17(11):e0276577, November 2022.

[14] Yaonan Dai, Jiuyang Yu, Cong Zhang, Bowen Zhan, and Xiaotao Zheng. A novel whale optimization algorithm of path planning strategy for mobile robots. *Applied Intelligence*, 53(9):10843–10857, August 2022.

[15] Abhishek Kumar Kashyap and Dayal R. Parhi. Obstacle avoidance and path planning of humanoid robot using fuzzy logic controller aided owl search algorithm in complicated workspaces. *Industrial Robot: the international journal of robotics research and application*, 49(2):280–288, September 2021.

[16] P.B. Fernandes, R.C.L. Oliveira, and J.V. Fonseca Neto. Trajectory planning of autonomous mobile robots applying a particle swarm optimization algorithm with peaks of diversity. *Applied Soft Computing*, 116:108108, February 2022.

[17] Zhang Qing, LIU Xu, PENG Li, and Zhu Fengzeng. Path planning for mobile robots based on jps and improved a* algorithm. *Journal of Frontiers of Computer Science & Technology*, 15(11):2233, 2021.

[18] Zhihai Liu, Hanbin Liu, Zhenguo Lu, and Qingliang Zeng. A dynamic fusion pathfinding algorithm using delaunay triangulation and improved a-star for mobile robots. *IEEE Access*, 9:20602–20621, 2021.

[19] Hongwei Tang, Wei Sun, Anping Lin, Min Xue, and Xing Zhang. A gwo-based multi-robot cooperation method for target searching in unknown environments. *Expert Systems with Applications*, 186:115795, December 2021.

[20] Jian Chen, Chengshuai Wu, Guoqing Yu, Deepak Narang, and Yuexuan Wang. Path following of wheeled mobile robots using online-optimization-based guidance vector field. *IEEE/ASME Transactions on Mechatronics*, 26(4):1737–1744, August 2021.

[21] Divyendu Kumar Mishra, Aby Thomas, Jinsa Kuruvilla, P. Kalyanasundaram, K. Ramalingeswara Prasad, and Anandakumar Haldorai. Design of mobile robot navigation controller using neuro-fuzzy logic system. *Computers and Electrical Engineering*, 101:108044, July 2022.

[22] Yao Rong, Chao Han, Christian Hellert, Antje Loyal, and Enkelejda Kasneci. Artificial intelligence methods in in-cabin use cases: A survey. *IEEE Intelligent Transportation Systems Magazine*, 14(3):132–145, May 2022.

[23] Jing Xu, Karen Kendrick, and Alex R. Bowers. Clinical report: Experiences of a driver with vision impairment when using a tesla car. *Optometry and Vision Science*, 99(4):417–421, February 2022.

[24] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2), April 2004.

[25] Seyed Matin Malakouti. Estimating the output power and wind speed with ml methods: A case study in texas. *Case Studies in Chemical and Environmental Engineering*, 7:100324, June 2023.