applied system innovation

Article

Adaptive Navigation in Collaborative Robots: A Reinforcement Learning and Sensor Fusion Approach

Rohit Tiwari [1], A. Srinivaas [2,*] and Ratna Kishore Velamati [2,*]

[1] Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore 641112, India; cb.en.p2ael23017@cb.students.amrita.edu
[2] Department of Mechanical Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore 641112, India
* Correspondence: a_srinivaas@cb.amrita.edu (A.S.); v_ratnakishore@cb.amrita.edu (R.K.V.)

Abstract: This paper presents a new approach for enhancing autonomous vehicle navigation and obstacle avoidance based on the integration of reinforcement learning with multiple sensors for navigation. The proposed system is designed to enable a reinforcement learning decision algorithm capable of making real-time decisions in aiding the adaptive capability of a vehicle. This method was tested on a prototype vehicle with navigation based on a Ublox Neo 6M GPS and a three-axis magnetometer, while for obstacle detection, this system uses three ultrasonic sensors. The use of a model-free reinforcement learning algorithm and use of an effective sensor for obstacle avoidance (instead of LiDAR and a camera) provides the proposed system advantage in terms of computational requirements, adaptability, and overall cost. Our experiments show that the proposed method improves navigation accuracy substantially and significantly advances the ability to avoid obstacles. The prototype vehicle adapts very well to the conditions of the testing track. Further, the data logs from the vehicle were analyzed to check the performance. It is this cost-effective and adaptable nature of the system that holds some promise toward a solution in situations where human intervention is not feasible, or even possible, due to either danger or remoteness. In general, this research showed how the application of reinforcement learning combined with sensor fusion enhances autonomous navigation and makes vehicles perform more reliably and intelligently in dynamic environments.

Keywords: obstacle avoidance; reinforcement learning; dynamic environments; autonomous vehicle navigation

check for updates

Academic Editors: Patrícia Ramos and Jose Manuel Oliveira

Received: 15 October 2024
Revised: 19 December 2024
Accepted: 24 December 2024
Published: 6 January 2025

Citation: Tiwari, R.; Srinivaas, A.; Velamati, R.K. Adaptive Navigation in Collaborative Robots: A Reinforcement Learning and Sensor Fusion Approach. *Appl. Syst. Innov.* **2025**, *8*, 9. https://doi.org/10.3390/asi8010009

1. Introduction

A collaborative robot (Cobot), is an advanced robotic system designed to interact safely and efficiently with human workers in shared workspaces, often featuring sensors and AI for enhanced precision and adaptability [1,2]. Cobots are very useful and are developed to improve efficiency, safety, and adaptability, which is a great help in cases when the working environment includes very little human presence or has no presence at all [3]. Associating with or even characterized by other unexpected obstacles, the operation of these machines in dynamic and unpredictable environments would be more suitable for future navigation and adaptive obstacle avoidance systems in Cobots [4].

These obstacle detection and avoidance methods in the case of the robots are more based on predefined rules or static models, which will work well for restricted environments, but these cannot adapt to real-world applications where situations keep on changing way too frequently. Path planning, a fundamental aspect of autonomous navigation for robots and vehicles, focuses on finding the shortest and most efficient path from an origin to a destination while avoiding collisions. The optimal algorithm choice hinges on whether the environment is static, with unchanging obstacle locations, or dynamic, where obstacles can move or appear suddenly. Dijkstra's algorithm (D *), efficient for static environments, guarantees the shortest path [5]. In contrast, the A star algorithm (A *), while also suited for static environments, employs heuristics to expedite pathfinding, making it computationally less demanding [6]. For dynamic environments, the D * algorithm shines due to its ability to swiftly adapt to changes in the environment, making it ideal for applications like planetary rovers [7]. Rapidly exploring Random Trees (RRTs) are particularly effective in dynamic settings and systems with non-holonomic constraints, such as those found in wheeled robots [8–11]. These algorithms explore possible paths by constructing a tree structure. Beyond these, nature-inspired algorithms like Genetic Algorithms, Ant Colony Optimization, and Firefly Algorithms offer unique advantages in handling complex environments [7]. Genetic Algorithms, mimicking biological evolution, may not always locate the absolute shortest path but deliver effective solutions [12]. Ant Colony Optimization, inspired by ant foraging behavior, utilizes virtual pheromone trails to pinpoint efficient paths and excels in dynamic and multi-robot systems [13]. Finally, Firefly Algorithms, simulating firefly flashing, utilize virtual fireflies' brightness to steer the search for an optimal path, proving suitable for both static and dynamic situations, particularly in underwater vehicles. Therefore, selecting the appropriate path-planning algorithm requires a thorough understanding of the environment's nature and the specific requirements of the task. Of course, these conventional methods are of no use if unexpected obstacles arise, which could result in a collision or inefficient navigation paths. Conventional path-planning algorithms, like Dijkstra's [5] and A * rely [6] on static models and predefined rules, which make them ill-suited for dynamic environments; hence, there is a requirement for more flexible, smart systems that can adapt quickly based on the environment condition for efficient Cobot operation and implementation.

The technique presented in this paper has its foundations in the principles of artificial intelligence (AI), specifically in the field of reinforcement learning (RL), a branch of machine learning within AI. The goal of artificial intelligence (AI) is to build systems that can learn, adapt, and make decisions on their own in challenging situations [14,15]. This is demonstrated by the model-free RL algorithm, which allows the Cobot to adapt its decision for obstacle avoidance when interacting with the surroundings. Unlike conventional ways, the approach does not rely on the model of the environment. Instead, the Cobot can learn from experiences and update its behavior online by fine-tuning its strategies in obstacle avoidance due to new obstacles. Also, by analyzing the decisions that were made by Cobot past decision data and its sensor data, a rough plot can be generated that maps the path that was followed by the Cobot and the positions of various obstacles faced by it. A summary of all the previously discussed features is shown in Figure 1.

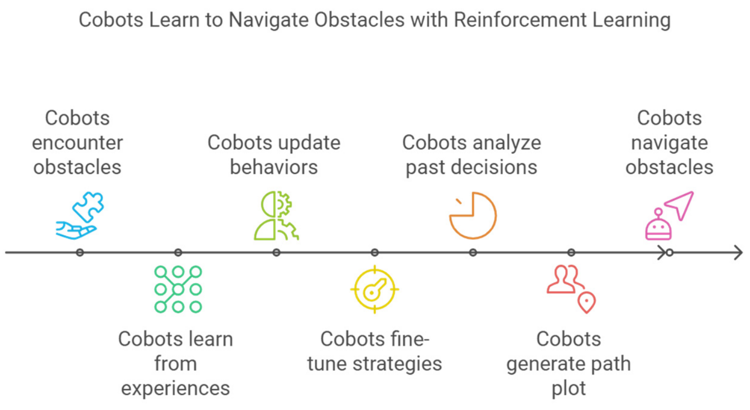
Cobots Learn to Navigate Obstacles with Reinforcement Learning
Cobots encounter obstacles
Cobots learn from experiences
Cobots update behaviors
Cobots fine-tune strategies
Cobots analyze past decisions
Cobots generate path plot
Cobots navigate obstacles

Figure 1. Cobot features.

2. Literature Review

One of the recent technologies being integrated into the autonomous vehicle system is the application of Unmanned Ground Vehicles. It can be seen [16] how GPS fundamentally guides UGVs from a source location to a target location while making sure that obstacles are evaded through ultrasonic sensors. The system is already preset with the GPS coordinates so that the vehicle will adhere to a certain route and adjust whenever it detects an obstacle all throughout. It has such sensors to present a relatively low-cost alternative to more expensive and complex sensors, like LiDAR [16]. The practical application of the ultrasonic sensors involves implementing an additional autonomous navigation system that improves the reliability and efficiency of UGVs in providing safety and guaranteeing correctness.

Another approach can be to [17,18] consider integrating GPS with magnetometers for better autonomous navigation. Position data can be obtained from the GPS while the orientation data can be captured by using the magnetometer. The LabVIEW HMI [17] can use such systems providing an easy-to-use platform that allows for the monitoring and control of the navigation process. The GPS and magnetometer coupling is best suited to be applied in urban or foliage-dense environments when a consistent, precise heading must be maintained or when GPS signals are weak and obstructed.

Further, research has [19] examined how an autonomous vehicle should cope with the possibility of losing its GNSS signals either within the environment or within specific zones. The motivation was to complement navigation with multiple ultrasonic sensors so that the vehicle would not rely solely on GPS. This provides a solution to estimate the navigation states of land vehicles in GNSS-denied environments. An Extended Kalman Filter (EKF) is applied to fuse low-cost ultrasonic sensors with an Inertial Measurement Unit (IMU). The rear wheels are mounted with ultrasonic sensors that measure the range difference in determining the angular velocity and forward velocity of the vehicle. The two experimental tests showed that the information required to navigate considerably improved the position accuracy during GNSS outages.

Where another implementation [20] explains that it is not possible for typical navigation systems that are dependent on GPS to work indoors or in closed environments; therefore, some other methods are to be adopted to use the UAV safely. Using magnetic field mapping as a means of navigation that uses the recording and visualization of changes in the local magnetic field with high resolution to develop magnetic maps. These maps act as a reference framework for UAVs to navigate autonomously within complex indoor spaces. The importance of doing proper calibration of the magnetometer lies in the fact that nonuniform results were obtained from the magnetometers of different batches of production. It is so important, particularly in the generation of correct magnetic field maps. It also shows the importance of real-time magnetic data visualization with tools such as MATLAB and Voxel for instant environment estimation and detection of magnetic anomalies. More emphasis is put into research related to the introduction of permanent magnets into the magnetic field, which in this case, will be taken as intentional perturbations; the anomalies act as a beacon that would give the UAVs their position and map in the environment. The observations conclude that with more exploration, magnetic field mapping will be a reliable way for indoor navigation in the systems of UAVs and provide precision control even if GPS is not available. Though our implementation makes use of GPS data, reliance on GPS could be reduced by using alternate methods [19,20].

On the other hand, various implementations [21] provide an extensive review of various techniques for mobile robot navigation, organized into classical and reactive strategies. Classical techniques [21], such as cell decomposition, roadmap methods, and artificial potential fields, combined with the use of reactive strategies that apply artificial intelligence methods, such as Genetic Algorithms and fuzzy logic, for real-time adaptability. There are a lot of works available that have compared methods, showing strengths and weaknesses in different scenarios and accentuating the trend toward hybrid techniques in which techniques are combined for improved performance. Also, based on the detected research gaps, these works have identified future directions for further enhancement of path-planning strategies to address the challenges generated by dynamic and complex environments.

A new system [22] that fuses 2D LiDAR data with RGB-D cameras can be used, leading to vastly superior mobile robot navigation in dynamic environments. Already, Bayesian estimation for multi-sensor data fusion is applied in this approach to further enhance the two-dimensional grid maps' accuracy and enable the discovery of obstacles below the scanning height of the LiDAR. The dynamic window method integrates a global path of planning and a local path of planning with an improved algorithm for effective global path planning and efficient local path planning, respectively. The experimental results demonstrated that this system could navigate static and dynamic environments, showing one application of a real robot system.

Moreover, the use of deep learning architectures in path planning for mobile robots has been explored, showing promising results in enhancing navigation and obstacle avoidance capabilities [23]. Among various models, the hybrid CNN-LSTM model exhibited superior performance, indicating the potential of deep learning in improving the efficiency and adaptability of path planning compared to traditional algorithms.

A hybrid sampling-based RRT * algorithm was introduced to address the limitations of conventional path-planning methods [24]. By integrating uniform and non-uniform sampling strategies, this algorithm demonstrated improved navigation efficiency in complex environments. The findings suggest that hybrid approaches can significantly contribute to autonomous navigation performance.

Another complex way to develop an autonomous navigation system [25] for mobile robots is by combining DRL with sensor fusion techniques, particularly using LiDAR and depth camera data. This leverages the heuristic global navigation function to select optimal points of navigation while employing the Soft Actor–Critic algorithm for local navigation, which in turn ensures that the robot can pave its way into the unknown environment. It underlines the system's capacity to move effectively around obstacles and respond to dynamic situations with superior performance in experimental trials compared to those under traditional methods, especially in the aspects of travel time and distance. This consequently implies that a quantifiable performance increase in mobile robot navigation within complex environments has been achieved using the approach taken here.

Neuromorphic computing and spiking neural networks (SNNs) have the potential to greatly enhance robotic navigation and obstacle avoidance algorithms by mimicking the biological information processing in the human brain. Bartolozzi discusses [26] how event-driven neuromorphic sensors such as Dynamic Vision Sensors (DVSs) capture relevant changes in a scene, ceasing to be clock-driven and thus reducing irrelevant data to be processed. It minimizes duplicate information, increases the temporal resolution of the observation, and allows for neural–semantic–motor integration, which is essential to traverse unstructured spaces. Martínez [27] in turn highlights the importance of SNNs in the context of autonomous driving, where their capacity for temporal processing allows for real-time decision-making with a high degree of energy efficiency, especially when implemented on neuromorphic hardware. But they admit difficulties in achieving the performance of traditional neural networks in essential tasks. Aitsam [28] highlights the utility of SNNs in the context of robotics, as SNNs are capable of reducing latency and energy consumption through the use of spiking mechanisms to process sensory data efficiently as well as adapt to environmental changes. Collectively, these studies establish firm groundwork for coupling ultrasonic sensors, GPS, and magnetometers to neuromorphic systems. Real-time map generation, representation, and semantic learning can all be achieved through model-free reinforcement learning; coupled with data-driven approaches, they pave the way to developing adaptive robots capable of continuous learning from their environments and improving navigation while avoiding unexpected obstacles. This integration of event-based sensing with brain-inspired computation and learning systems is a promising approach toward developing autonomous robotic systems capable of efficient operation in complicated, dynamic environments.

The paper [29] discusses integrating ethical reasoning into reinforcement learning (RL) agents using a modular normative supervisor. This supervisor employs defeasible deontic logic to ensure that the agent complies with ethical norms while maintaining optimal learned behavior. The approach dynamically checks for compliance, handles conflicting norms, and selects 'lesser evil' actions when no fully compliant option exists. Applications of this approach in Cobots could include avoiding high-risk zones based on GPS or sensor data, limiting speed, or halting movement if predetermined objects, such as pedestrians, are detected.

Lastly, the development of a hybrid algorithm integrating classical and heuristic methods for mobile robot path planning has shown enhanced capabilities in dynamic environments [30]. This approach addresses the challenges of real-time navigation in the presence of both static and dynamic obstacles, resulting in superior adaptability and efficiency in navigating complex scenarios.

Cobot navigation system implementations reviewed above show methods that help the robot vehicle to navigate using cost-effective sensors like GPS, ultrasonic sensors, and magnetometers, while others use advanced sensors, like LiDAR, etc., along with intelligent algorithms. Our focus is to use cost-effective sensors along with an intelligent algorithm to get an effective performance out of the system to perform desired navigation and obstacle avoidance.

3. Methodology

The proposed methodology system combines a model-free RL algorithm with real-time sensor data to give the Cobot flexibility in navigation and the ability to avoid obstacles effectively. The system allows the Cobot to make on-the-fly decisions about its path by utilizing the Ublox Neo 6M GPS, which is combined with a tri-axis magnetometer for navigation and ultrasonic sensors to fine-tune its path as needed. Thus, this approach helps to navigate effectively, ensuring that the Cobot can safely move to its destination in a complex or unknown environment.

This study presents the development and implementation of a cost-effective prototype robotic system, providing a practical solution for various autonomous navigation tasks. Apart from the use of Cobot, this method can also have applications in deploying vehicle robots into remote or potentially dangerous environments where human presence is not practical. An overview of the RL Cobot system is shown in Figure 2.

Cobot is fed with the destination GPS coordinates alone to the test vehicle, which then uses the readings from the GPS and tri-axis magnetometer sensors to adjust its heading angle and move forward. Additionally, three ultrasonic sensors are used to detect obstacles and ensure the vehicle follows a drivable path. When an obstacle is detected, the vehicle system employs a model-free reinforcement learning algorithm to navigate around it.

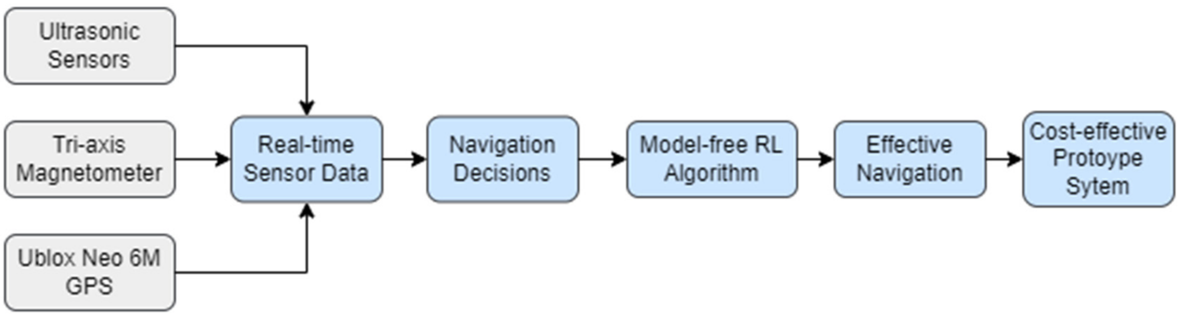

Figure 2. Overview of the RL Cobot system.

3.1. The Haversine Formula

The Haversine formula is used to get the distance between two positions on a sphere using their coordinates. It is especially valuable in navigation and mapping applications for determining the shortest path between two locations on Earth [31,32]. Using this formula, a vehicle can determine whether it has reached its destination. Mathematical equations are shown below:

$$x = sin^2\left(\frac{\Delta\phi}{2}\right) + cos(\phi_1)cos(\phi_2)sin^2\left(\frac{\Delta\lambda}{2}\right) \tag{1}$$

$$y = 2{\cdot}a{\cdot}tan2\left(\sqrt{x}, \sqrt{1-x}\right) \tag{2}$$

$$d = R{\cdot}c \tag{3}$$

where ϕ_1, ϕ_2 are the latitudes of current location and destination in radians. λ_1, λ_2 are the longitudes of current location and destination in radians. $\Delta\phi = \phi_1 - \phi_2$ is the latitude difference. $\Delta\lambda = \lambda_1 - \lambda_2$ is the longitude difference. R is the radius of Earth. c is the angular distance between the points. d is the distance between the two points.

3.2. The Initial Bearing Formula

The Initial Bearing Formula, also known as the Forward Azimuth Formula, calculates the direction or bearing from one geographical point to another on the Earth's surface. It uses the latitudes and longitudes of the two points to determine the angle relative to true north [33]. This bearing is crucial in navigation and mapping for determining the shortest path or direction to follow over the Earth's curved surface. The result is typically expressed in degrees, ranging from 0 to 360.

Mathematically: (ϕ_1, λ_1): latitude and longitude of the starting point. (ϕ_2, λ_2): latitude and longitude of the destination point.

$$\theta = atan2(sin(\Delta\lambda) \times cos(\phi_2) \times cos(\phi_1) \times sin(\phi_2) - sin(\phi_1) \times cos(\phi_2) \times cos(\Delta\lambda)) \tag{4}$$

where θ is the initial bearing angle.

3.3. The Bearing Angle Formula for Magnetometer

To calculate the bearing angle from LIS2MDL data, use the formula below:

$$\theta = atan2(Y, X) \tag{5}$$

where X and Y are the magnetic field strength values obtained after calibration of sensor in the X and Y directions, respectively. This formula provides the angle relative to magnetic north. Convert the result from radians to degrees and adjust for magnetic declination if needed. Finally, normalize the angle to the 0–360-degree range. This bearing angle helps determine direction relative to the Earth's magnetic field for navigation and orientation.

3.4. Navigation Algorithm

Figure 3 shows the algorithm for navigation that was used in this study in our test vehicle. The algorithm begins by obtaining the vehicle's current GPS coordinates and the desired destination coordinates. The GPS module supplies the vehicle's current location, and the microcontroller processes this information to calculate both the required bearing angle, which indicates the direction to the destination and the distance between the current position and the destination.

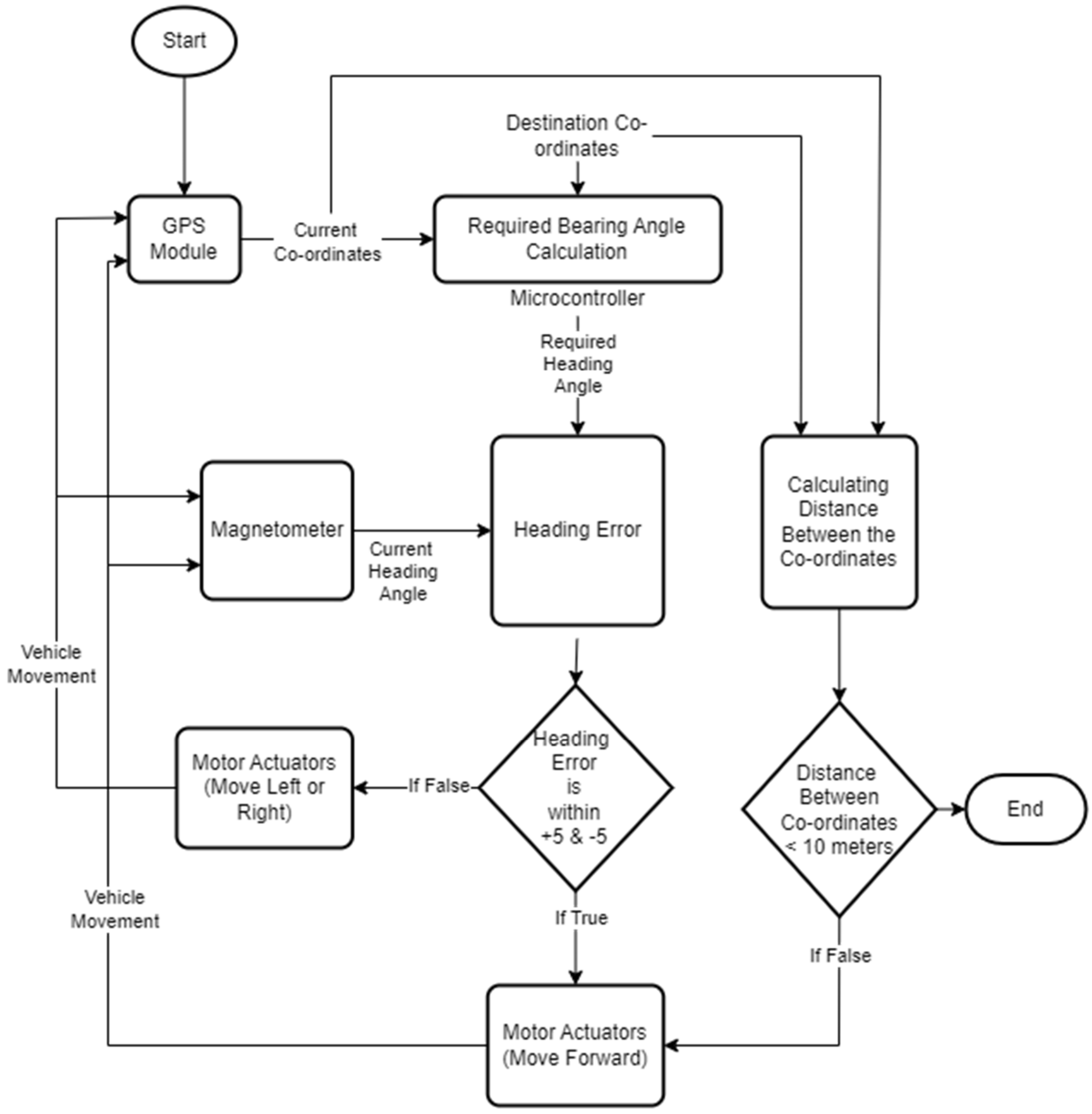
Start
Destination Co-ordinates
GPS Module
Current Co-ordinates
Required Bearing Angle Calculation
Microcontroller
Required Heading Angle
Magnetometer
Current Heading Angle
Heading Error
Calculating Distance Between the Co-ordinates
Vehicle Movement
Motor Actuators (Move Left or Right)
If False
Heading Error is within +5 & -5
Distance Between Co-ordinates < 10 meters
End
Vehicle Movement
If True
If False
Motor Actuators (Move Forward)

Figure 3. Navigation algorithm flow diagram.

Next, the vehicle's current heading angle is measured by the magnetometer. The microcontroller then determines the heading error by comparing this current heading with the calculated bearing angle.

The vehicle's movement is guided by the heading error and the distance to the destination. If the heading error falls within a specified tolerance of ±5 degrees and the distance to the destination is less than 10 m, the vehicle will stop. If the heading error is outside the tolerance, the vehicle's motor actuators adjust its direction by turning left or right to correct the heading. If the heading is correct but the distance to the destination exceeds 10 m, the vehicle moves forward.

3.5. Obstacle Avoidance Algorithm

For obstacle avoidance, the suggested approach makes use of a model-free reinforcement learning (RL) algorithm. This approach is recommended over other machine-learning (ML) algorithms because it may learn optimal behavior directly from interactions with the environment, making it ideal for dynamic, real-time decision-making tasks. Unlike other algorithms, RL does not require large, labeled datasets and can adapt to changing conditions over time [34]. This flexibility makes RL particularly suited for applications like robotics, gaming, and autonomous vehicles, where traditional machine-learning approaches may struggle. Model-free RL continuously adjusts its approach based on real-time feedback. Also, for microcontrollers with less programming memory, this model-free RL algorithm can be used due to low memory requirements as compared to other supervised ML algorithms [35,36].

In this reinforcement learning-based algorithm, the test vehicle navigates its environment by integrating sensor inputs and learning from interactions. The process begins with the front ultrasonic sensor measuring the distance to the nearest obstacle. If an obstacle is detected within a specified range (40 cm or 60 cm), the test vehicle must avoid it. The test vehicle then chooses a direction to turn—left or right—based on a 50/50 probability. Post-turn, the test vehicle continues forward. The algorithm tracks the success rate of each direction: If turning in a given direction successfully avoids the obstacle, the success rate for that direction increases; if it results in another obstacle, the success rate decreases. To balance exploration and exploitation, a random element is introduced. A random number between 1 and 100 determines the test vehicle's behavior: If the number is between 1 and 10, the test vehicle performs a random turn to explore new paths; if the number is between 11 and 100, it exploits the direction with the higher success rate. Meanwhile, the left and right ultrasonic sensors help maintain the vehicle on its path by avoiding side crashes through vehicle tilting rather than detecting hard obstacles. One potential source of the problem that can arise from this algorithm (also known as reward hacking issue in reinforcement learning) is when difference between the success rate values becomes too large, causing the turn decision with the higher value to dominate. To avoid this, a maximum value can be predefined for success rate [37]. Once a success rate reaches this maximum value, it can be reset to the initial value.

This approach exemplifies how reinforcement learning can be effectively applied to small robot vehicles such as the test vehicle used in this study, balancing the need for exploration of new strategies and exploitation of successful ones to optimize performance in real time. The flow diagram is shown in Figure 4.

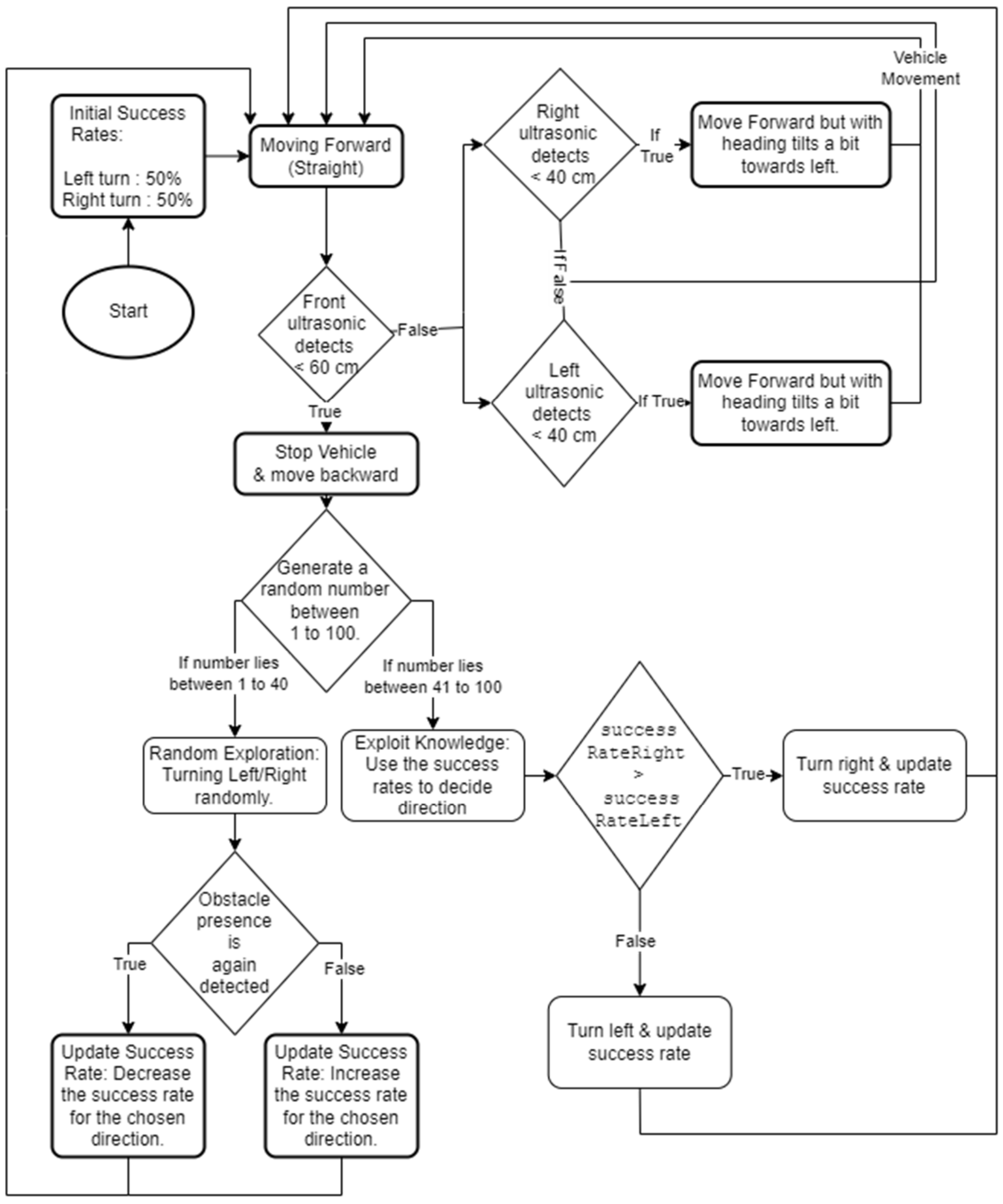
Initial Success Rates:
Left turn : 50%
Right turn : 50%
Start
Moving Forward (Straight)
Front ultrasonic detects < 60 cm
False
True
Right ultrasonic detects < 40 cm
If True
Move Forward but with heading tilts a bit towards left.
If False
Left ultrasonic detects < 40 cm
If True
Move Forward but with heading tilts a bit towards left.
Vehicle Movement
Stop Vehicle & move backward
Generate a random number between 1 to 100.
If number lies between 1 to 40
If number lies between 41 to 100
Random Exploration: Turning Left/Right randomly.
Exploit Knowledge: Use the success rates to decide direction
successRateRight > successRateLeft
True
Turn right & update success rate
False
Turn left & update success rate
Obstacle presence is again detected
True
False
Update Success Rate: Decrease the success rate for the chosen direction.
Update Success Rate: Increase the success rate for the chosen direction.

Figure 4. Obstacle avoidance algorithm flow diagram.

4. System Architecture

The system architecture is shown in the following Figure 5. It contains several key components, including sensors, a microcontroller, a power supply, communication modules, and movement control units.

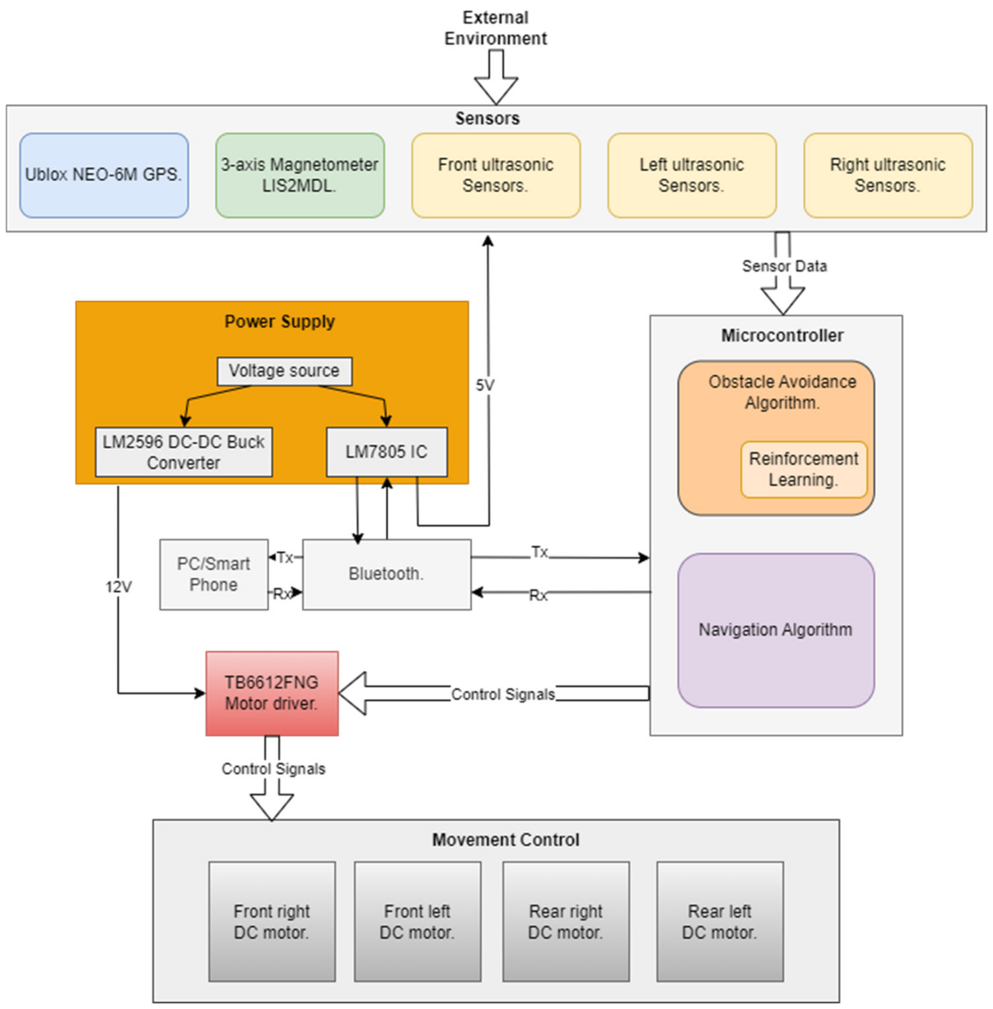

Figure 5. System architecture diagram.

4.1. Sensors

4.1.1. Ublox NEO-6M GPS Module

This module is a high-performance, compact GPS receiver designed to provide accurate position data. It operates by receiving signals from multiple GPS satellites orbiting the Earth. The module uses its antenna to capture these signals, which include each satellite's location and time information. It then processes the signals to calculate the distance between the satellites and the receiver. Using this distance information along with the known satellite positions, the NEO-6M triangulates the receiver's position on the Earth's surface. The module outputs the calculated position and time data via serial communication (UART), which can be utilized by connected devices, such as a robot car, for navigation and tracking purposes. Tables 1 and 2 shows the default configuration of this module and the default configuration of the NEO-6M GPS module respectively.

Furthermore, a computer application, u-center, is available, allowing connection to the GPS module via a TTL to USB connector. This application enables the configuration of various parameters of the module to meet specific requirements. Figure 6 shows the interface of the u-center application connected to the GPS module.

Table 1. Technical specifications of the Ublox NEO-6M GPS module.

Feature	Specification
Antenna	25 × 25 × 4 mm (ceramic, active)
Power Supply	3–5 V
Baud Rate	9600 bps (default)
Position Update Rate	5 Hz
Cold Start Time	38 s
Hot Start Time	1 s
Baud Rate Range	4800 to 115,200 bps
Tracking Sensitivity	−161 dB
Satellite Channels	22 satellites, 50 channels
Current Consumption	45 mA
SBAS Support	WAAS, EGNOS, MSAS, GAGAN

Table 2. Used configuration of the Ublox NEO-6M GPS module.

Parameter	Default	Description
Baud Rate	9600 bps	Speed of communication.
Protocol	NMEA	GPS data output protocol.
Update Rate	1 Hz	GPS position update frequency.
Accuracy	2.5 m (approx.)	Typical positional accuracy in open sky conditions.
Coordinate System	WGS-84	Default coordinate system used for position data.

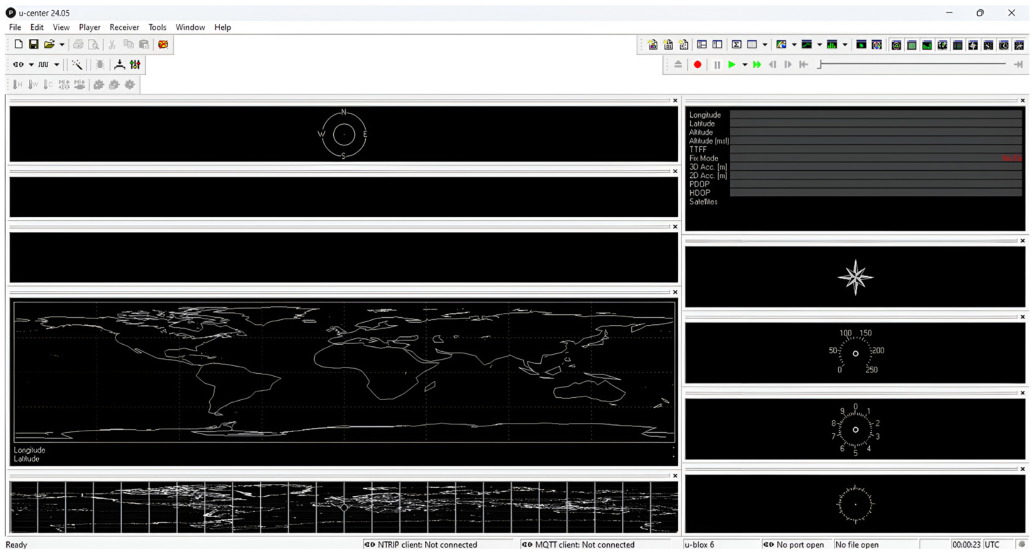

Figure 6. U-center interface.

4.1.2. LIS2MDL 3-Axis Magnetometer

The LIS2MDL is a high-performance three-axis magnetometer designed for precise magnetic field measurements, making it ideal for applications such as navigation systems and electronic compasses. It operates by detecting the Earth's magnetic field along three orthogonal axes (X, Y, and Z) using magneto resistive elements. The detailed specifications are shown in Table 3. These elements measure the magnetic field's strength and convert the analog signals into digital data. Calibration of this sensor is a must to obtain correct readings. The steps below are performed to calibrate the sensor used in this study:

1. Move the sensor to capture the full range of magnetic field values.
2. Identify the minimum and maximum magnetic field readings for each axis.
3. Calculate the offsets and scale factors based on the min/max values.
4. Adjust raw data by subtracting offsets and dividing by scale factors.

Table 3. Technical specifications of the LIS2MDL 3-axis magnetometer.

Feature	Specification
Magnetic Field Range	±50 gauss
Data Output	16-bit
Interfaces	SPI, I2C
Supply Voltage	1.71 V to 3.6 V
Temperature Range	−40 °C to +85 °C
Power Modes	Selectable
Measurement Mode	Single
Interrupt Generator	Programmable
Self-test	Embedded
Temperature Sensor	Embedded

As this sensor is used to calculate the current heading angle (range: 0 to 360 degrees) of the vehicle, only X and Y values will be used.

4.1.3. Ultrasonic Sensor

HC-SR04 ultrasonic sensor emits a pulse, which travels through the air and bounces back after hitting an object. By measuring the time it takes for the echo to return, the sensor calculates the distance to the object using the speed of sound, having a maximum range of 4 to 5 m. However, more advanced versions have a higher range than this. A total of three sensors are used and mounted on the left, right, and front of the test vehicle [16]. The detailed specifications are shown in Table 4.

Table 4. Technical specifications of the HC-SR04 ultrasonic sensor.

Feature	Specification
Operating Voltage	5 V DC
Working Current	15 mA
Frequency	40 Hz
Measurement Range	2 cm to 400 cm
Accuracy	±3 mm
Measuring Angle	<15°
Trigger Input	10 µs TTL pulse
Echo Output	TTL level signal
Dimensions	45 × 20 × 15 mm

4.2. Power Supply

The system’s power supply is designed to provide the necessary voltage to different components. It uses an LM2596 DC–DC Buck Converter to step down the input voltage and provides an output for the motor driver and a stable 5 V regulated output for sensors. The detailed specifications are shown in Table 5.

Table 5. Technical specifications of the LM2596 DC–DC Buck Converter.

Feature	Specification
Input Voltage Range	4.0 V to 40 V
Output Voltage	1.2 V to 37 V (adjustable)
Output Current	3 A (max)
Switching Frequency	150 kHz
Conversion Efficiency	92% (highest)
Load Regulation	±0.5%
Voltage Regulation	±0.5%
Output Ripple	30 mA (max)
Dynamic Response Speed	5% in 200 µs
Thermal Shutdown Protection	Yes
Current Limit Protection	Yes
Standby Mode Current	80 µA (typical)

4.3. Microcontroller: Arduino Mega 2560

The Arduino Mega 2560 is the processing unit for this system and its configuration is shown in Table 6. It manages the input and output signals from various sensors and controls the vehicle's actuators based on sensor data. The Arduino Mega is selected for its ample number of input/output pins and large code memory size.

Table 6. Technical specifications of the Arduino Mega 2560.

Feature	Specification
Microcontroller	ATmega2560
Operating Voltage	5 V
Input Voltage (recommended)	7–12 V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
Flash Memory	256 KB (8 KB used by bootloader)
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Communication Interfaces	UART, SPI, I2C, USB

4.4. Bluetooth Module: HC-05

The system includes a communication interface i.e., Bluetooth, allowing for remote control and monitoring through a PC or smartphone and its technical specification is provided in the Table 7. The Tx (transmit) and Rx (receive) lines facilitate data exchange between the microcontroller, the Bluetooth module, and the control system. Sensor data can also be exported as logs for further analysis.

Table 7. Technical specifications of the HC-05 Bluetooth module.

Feature	Specification
Bluetooth Protocol	Bluetooth V2.0 + EDR
Frequency	2.4 GHz ISM Band
Operating Voltage	3.3 V (regulated)
Input Voltage	3.6 V to 6 V
Communication Range	Up to 10 m
Data Rate	Asynchronous: 2.1 Mbps (max)/160 kbps
Communication Method	UART (Default Baud Rate: 9600 bps)
Master/Slave	Configurable
Default PIN	1234 or 0000
Power Consumption	30 mA (average)

4.5. Motor Driver and Movement Control

The vehicle's movement is managed by a TB6612FNG motor driver, which receives control signals from the microcontroller. The motor driver controls the four DC motors (front right, front left, rear right, and rear left) responsible for the vehicle's movement. This setup enables control over the vehicle's direction, speed, and stopping mechanisms, ensuring smooth navigation and effective obstacle avoidance. Technical specifications are shown in Table 8.

Table 8. Technical specification of the TB6612FNG motor driver.

Feature	Description
Description	The TB6612FNG is a dual DC motor driver IC by Toshiba
Control Capability	Independently controls two DC motors or one bipolar stepper motor
Control Modes	CW (clockwise), CCW (counterclockwise), short-brake, and stop
Voltage Range	4.5 V to 13.5 V
Current Output	1 A continuous, 3 A peak per channel

4.6. Cost Comparison of Components Used

The below Tables 9 and 10 shows a cost of the components, and the cost of components used in alternate solution for comparison of the components used for navigation and obstacle avoidance problems.

Table 9. Cost of components used in this study.

Feature	Description	Cost (INR)
GPS module	Ublox Neo 6M	198
Magnetometer	LIS2MDL 3-axis	257
Obstacle detection	HC-SR04	55
Microcontroller	ATmega2560	2000–4000

Table 10. Cost of components used in alternate solutions.

Feature	Description	Cost (INR)
GPS module	High-precision GPS module	2335–20,000
Magnetometer	High-grade digital compass	1000 and above
Obstacle detection	LiDAR sensor	5000–10,499
Microcontroller	Many options are available	8000 and above

5. Experimental Results

To test the navigation and obstacle avoidance capabilities of the test vehicle, a hallway was selected with obstacles arranged in such a way that most are on the right side, while some are positioned at the center and left. Figure 7 below shows the map view of the experiment test path.

The purpose of this arrangement is to evaluate the performance of the vehicle's navigation and obstacle avoidance systems, specifically how well the vehicle adapts to varying path conditions.

A logging system was installed in the vehicle to record data related to decision-making during navigation, including the vehicle's heading angle and how it adapts to its path. These data were transmitted to a computer via Bluetooth for analysis, and visualization and was used to assess the vehicle's performance.

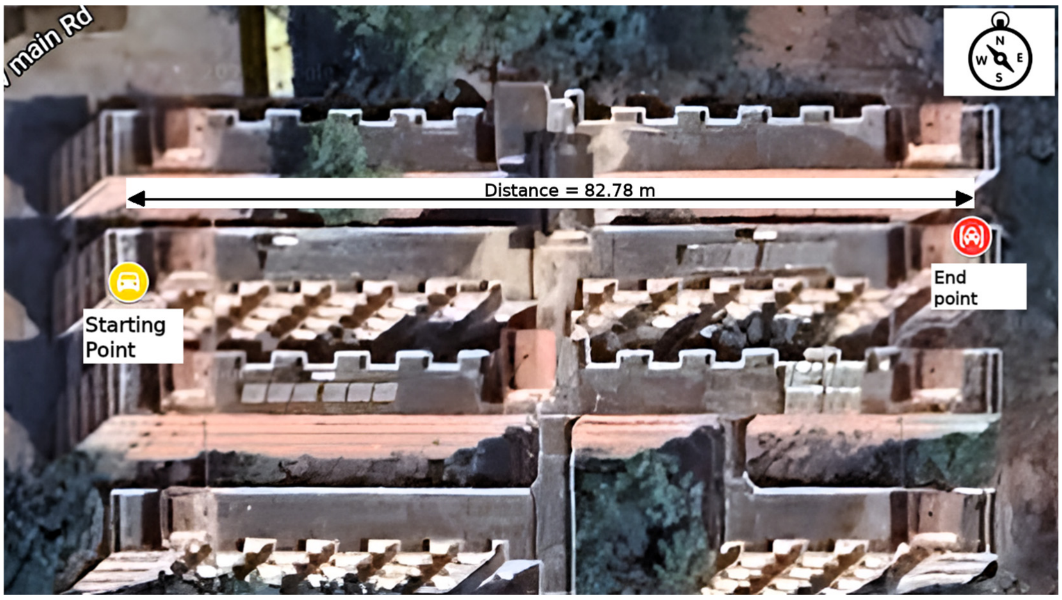

Figure 7. Test track satellite view.

Using the sequence of decisions made by the vehicle while driving autonomously from the start point to the endpoint, a plot has been created that visually represents the path followed and the decisions made along the way. The x and y axes do not have any specific units; the plot is intended to give a general sense of the vehicle's trajectory and the sequence of decisions taken. This plot facilitates basic path mapping. Figure 8 below illustrates this plot.

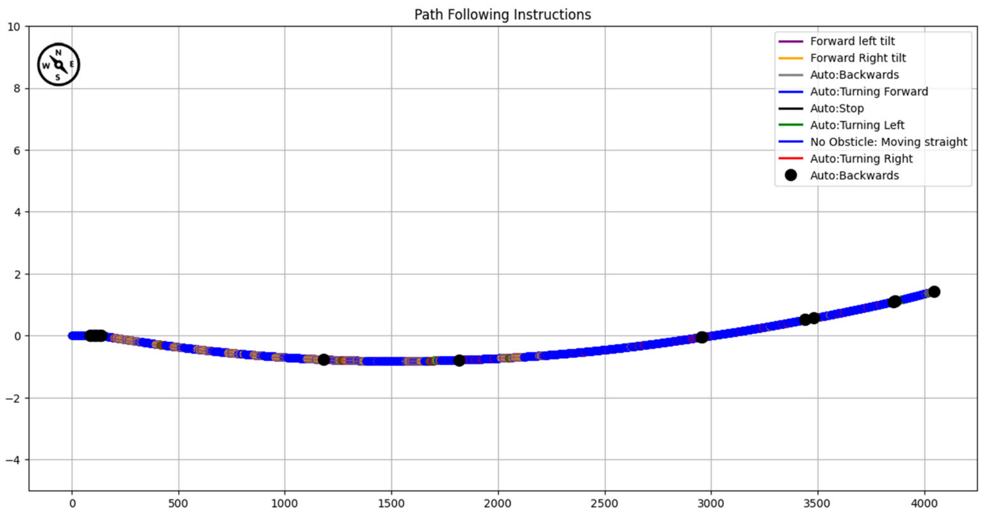

Figure 8. Path mapping from test vehicle's decisions.

According to the obstacle avoidance algorithm, the vehicle moves backward when the front sensor detects an obstacle. The black points in Figure 8 indicate backward movement and the presence of obstacles. There are a total of seven clusters of such points on the graph. Nearby black points suggest the vehicle is dealing with the same obstacle, representing a single obstacle. Comparing this path with the test track shown in Figure 7, it closely aligns with the decisions made during the test.

Using the data logged from the magnetometer, visualization of the current heading angle of the vehicle can be performed. Figure 9 represents the magnetometer data in polar plot form.

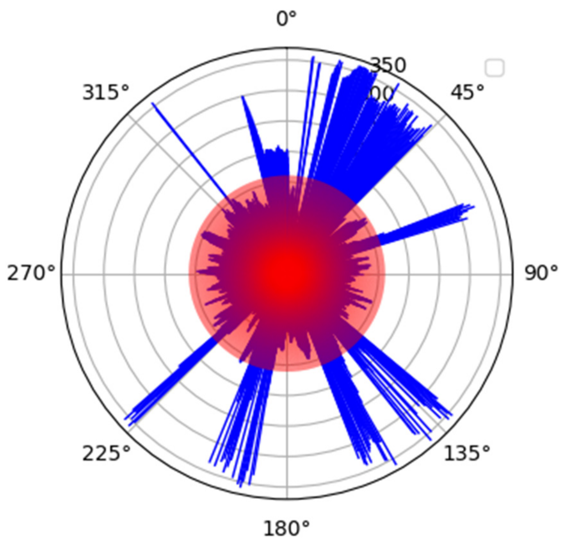

Figure 9. Polar plot of heading angle data(red: high density; blue: data points).

A bar graph representation of the same data is shown below in the same sequence as per vehicle's movement.

In both plots, the red color indicates the desired heading to reach the endpoint, while the blue color represents the current heading of the vehicle. Figure 10 shows spikes in the current heading values, which represent either the presence of obstacles and the vehicle's attempts to navigate around them or potential external magnetic disturbances affecting the sensor. A total of seven spike clusters can be observed, which also correspond to the path mapped in Figure 8 using navigation instruction data.

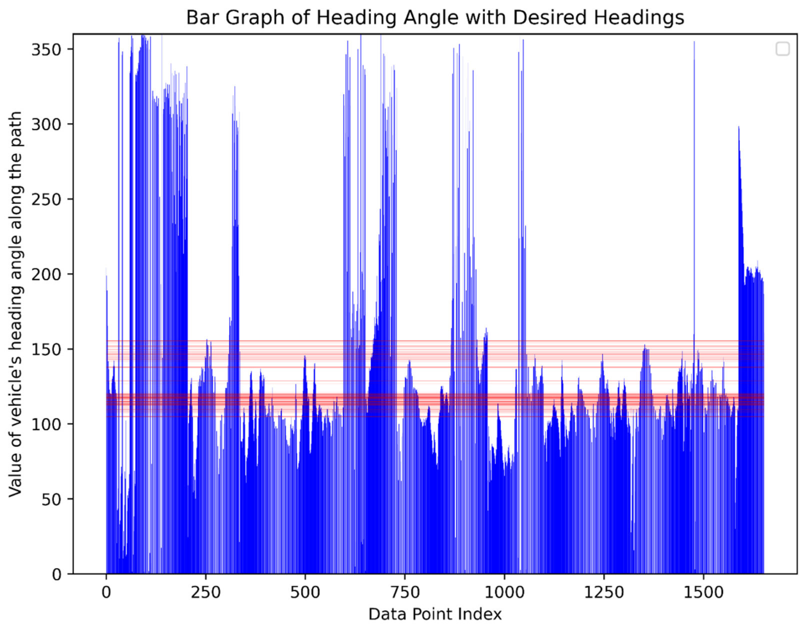

Figure 10. Bar graph of heading angle data (red: desired values; blue: data points).

Furthermore, Figure 10 demonstrates that the vehicle closely follows the desired heading angle, indicated by the dark red horizontal line, even after overcoming obstacles.

Figure 11 shows the test vehicle. Test vehicle employs reinforcement learning to make left- or right-turning decisions for obstacle avoidance. The decision-making weight, reflecting the success rate of obstacle avoidance, is continually updated. Figure 12 shows the plot of these weights and the success rate as the vehicle progresses from the start point to the endpoint.

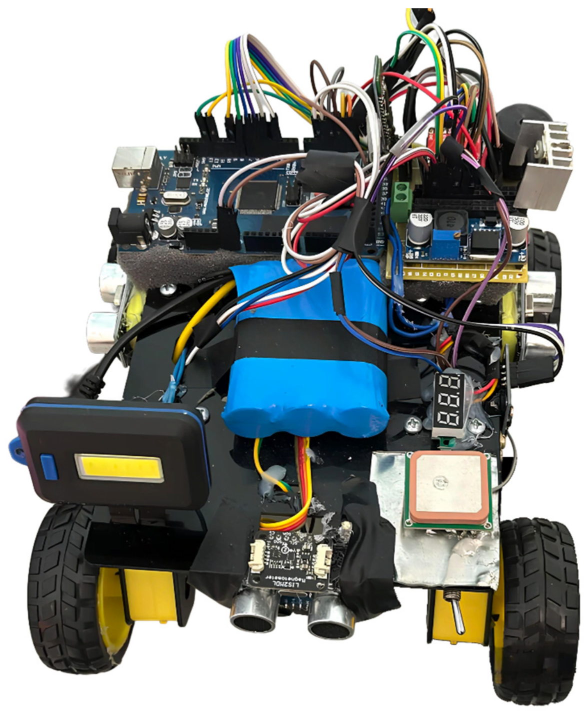
Figure 11. Test vehicle.

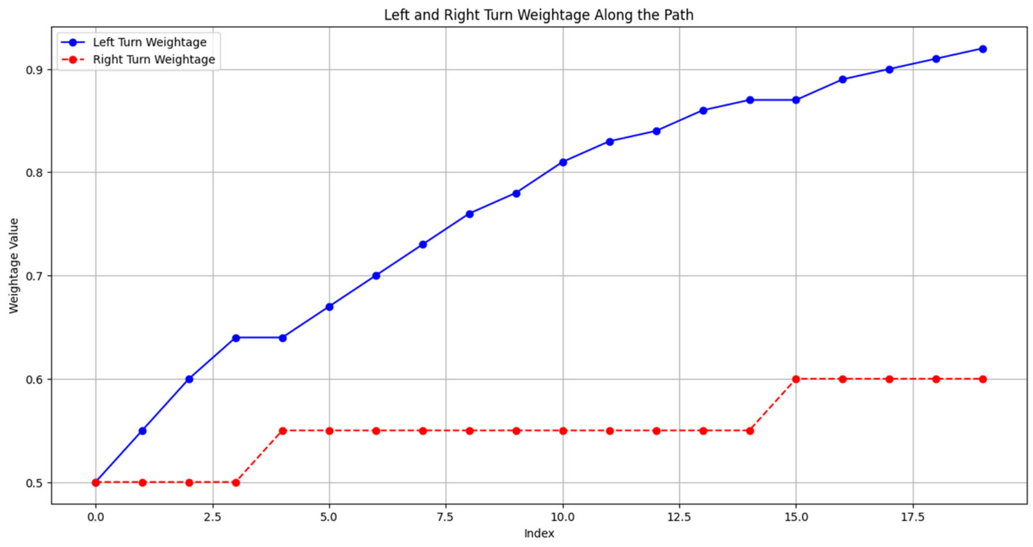

Figure 12. Change in decision weights.

The weightage for left-turn decisions increases as the vehicle progresses toward the destination, while the right-turn weightage also rises but lags the left. This pattern is expected because the test track has most of its obstacles on the right side. Thus, the algorithm is adapting well to the test track conditions. The vehicle balances exploration and exploitation, using knowledge 60% of the time and exploring new options 40% of the time, maintaining a balance to avoid biasing any single turning decision.

6. Future Scope

The future potential of this system lies in expanding its application to more complex environments, integrating advanced sensors, and adapting it for use in larger fleets of autonomous vehicles or drones [15,38]. Key areas of focus will include improving energy efficiency, reducing computational demands, and enhancing real-time decision-making capabilities. By incorporating reinforcement learning alongside sensor fusion, the system can achieve better navigation accuracy and adaptability in dynamic, real-world conditions.

However, adapting this technology for drones presents unique challenges. Drones have limited payload and battery life, which can limit the types of sensors they can carry. GPS reliability for aerial navigation is also often compromised, especially in areas with poor signal reception. The need for 3D obstacle avoidance adds another layer of complexity to the system's design. Ultrasonic sensors, for instance, may not offer the necessary range for fast-moving drones, and the high demands of real-time processing could strain onboard systems. Nevertheless, with thoughtful required adjustments and modifications in the system algorithm, this method can be successfully adapted for drone applications.

Looking ahead, the system can benefit from more sophisticated sensor fusion techniques and the integration of advanced sensors such as LiDAR and cameras to further enhance the capabilities of test bots. A failsafe mechanism could also be implemented, ensuring that in the event of a sensor failure, data from alternative sensors such as a magnetometer [18] or ultrasonic sensors [17] could provide reliable navigation.

7. Conclusions

This research explores the integration of reinforcement learning (RL) and sensor fusion to enhance autonomous vehicle navigation and obstacle avoidance. By combining multiple sensors, the system leverages their unique capabilities to improve navigation precision and reliability. The approach integrates GPS for positioning, a magnetometer for directional guidance, and ultrasonic sensors for obstacle detection.

The experimental setup demonstrated the vehicle's ability to dynamically adapt to its environment through a lightweight flexible learning algorithm. By processing data from GPS, a three-axis magnetometer, and ultrasonic sensors, the test vehicle successfully navigated to its destination while effectively avoiding obstacles in its path. Furthermore, the vehicle's decision-making and sensor data can be processed further to obtain a normalized map of the Cobot's path.

This solution is particularly promising for environments where human operations are challenging or dangerous. The collaborative robot (Cobot) used in this study offers significant advantages, including adaptability, cost-effectiveness, and minimal computational requirements. The research highlights the potential of RL and sensor fusion in developing intelligent, autonomous navigation systems.

Author Contributions: Conceptualization, Formal Analysis—R.T.; Investigation, Writing—Original Draft Preparation—A.S.; Supervision, Project Administration—R.K.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: All data relevant to this study are included within the article itself. Any additional data and materials can be obtained from the corresponding author upon reasonable request.

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

References

1. Muratore, L.; Laurenzi, A.; De Luca, A.; Bertoni, L.; Torielli, D.; Baccelliere, L.; Del Bianco, E.; Tsagarakis, N.G. A Unified Multimodal Interface for the RELAX High-Payload Collaborative Robot. *Sensors* **2023**, *23*, 7735. [CrossRef]
2. Torielli, D.; Muratore, L.; De Luca, A.; Tsagarakis, N. A Shared Telemanipulation Interface to Facilitate Bimanual Grasping and Transportation of Objects of Unknown Mass. In Proceedings of the 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids), Ginowan, Japan, 28–30 November 2022; pp. 738–745. [CrossRef]

3. Cao, H.-L.; Esteban, P.G.; Albert, D.B.; Simut, R.; Van de Perre, G.; Lefeber, D.; Vanderborght, B. A Collaborative Homeostatic-Based Behavior Controller for Social Robots in Human–Robot Interaction Experiments. *Int. J. Soc. Robot.* **2017**, *9*, 675–690. [CrossRef]
4. Huang, T.; Chen, Z.; Gao, W.; Xue, Z.; Liu, Y. A USV-UAV Cooperative Trajectory Planning Algorithm with Hull Dynamic Constraints. *Sensors* **2023**, *23*, 1845. [CrossRef]
5. Dijkstra, E. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
6. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
7. Karur, K.; Sharma, N.; Dharmatti, C.; Siegel, J.E. A Survey of Path Planning Algorithms for Mobile Robots. *Vehicles* **2021**, *3*, 448–468. [CrossRef]
8. Urmson, C.; Simmons, R. Approaches for Heuristically Biasing RRT Growth. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), Las Vegas, NV, USA, 27–31 October 2003; IEEE: Piscataway, NJ, USA, 2003; pp. 1178–1183.
9. Chen, X.; Zhao, Y.; Fan, J.; Liu, H. Three-Dimensional UAV Track Planning Based on the GB-PQ-RRT* Algorithm. In Proceedings of the 2023 42nd Chinese Control Conference (CCC), Tianjin, China, 24–26 July 2023; pp. 4639–4644. [CrossRef]
10. Liu, Y.; Tao, W.; Li, S.; Li, Y.; Wang, Q. A Path Planning Method with a Bidirectional Potential Field Probabilistic Step Size RRT for a Dual Manipulator. *Sensors* **2023**, *23*, 5172. [CrossRef]
11. Dai, J.; Zhang, Y.; Deng, H. Novel Potential Guided Bidirectional RRT* with Direct Connection Strategy for Path Planning of Redundant Robot Manipulators in Joint Space. *IEEE Trans. Ind. Electron.* **2024**, *71*, 2737–2747. [CrossRef]
12. McCall, J. Genetic Algorithms for Modelling and Optimisation. *J. Comput. Appl. Math.* **2005**, *184*, 205–222. [CrossRef]
13. Akka, K.; Khaber, F. Mobile robot path planning using an improved ant colony optimization. *Int. J. Adv. Robot. Syst.* **2018**, *15*, 1729881418774673. [CrossRef]
14. Fang, C.; Cheng, C.; Tang, Z.; Li, C. Research on Routing Algorithm Based on Reinforcement Learning in SDN. *J. Phys. Conf. Ser.* **2019**, *1284*, 012053. [CrossRef]
15. Xu, Y. Research on Reinforcement Learning Algorithm for Path Planning of Multiple Mobile Robots. *J. Phys. Conf. Ser.* **2021**, *1915*, 042022. [CrossRef]
16. Raja, M.; Guven, U. Design of Obstacle Avoidance and Waypoint Navigation using Global Position Sensor/Ultrasonic Sensor. *INCAS Bull.* **2021**, *13*, 149–158. [CrossRef]
17. Shahid, M.B.; Shahzad, M.U.; Bukhari, S.M.R.; Abasi, M.A. Autonomous vehicle using GPS and magnetometer with HMI on LabVIEW. In Proceedings of the 2016 Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Tokyo, Japan, 20–22 July 2016; pp. 163–167. [CrossRef]
18. Parakkal, P.G.; Variyar, V.V.S. GPS-Based Navigation System for Autonomous Car. In Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, 13–16 September 2017; pp. 1888–1893. [CrossRef]
19. Moussa, M.; Moussa, A.; El-Sheimy, N. Multiple ultrasonic aiding system for car navigation in GNSS denied environment. In Proceedings of the 2018 IEEE/ION Position, Location and Navigation Symposium (PLANS), Monterey, CA, USA, 23–26 April 2018; pp. 133–140. [CrossRef]
20. Brzozowski, B.; Kazmierczak, K. Magnetic field mapping as a support for UAV indoor navigation system. In Proceedings of the 2017 IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace), Padua, Italy, 21–23 June 2017; pp. 583–588. [CrossRef]
21. Patle, B.K.; Pandey, A.; Parhi, D.; Jagadeesh, A. A review: On path planning strategies for navigation of mobile robot. *Def. Technol.* **2019**, *15*, 582–606. [CrossRef]
22. Zhang, B.; Zhang, J. Robot mapping and navigation system based on multi-sensor fusion. In Proceedings of the 2021 4th International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 28–31 May 2021; pp. 632–636. [CrossRef]
23. Siddarth, K.S.; Barathraj, M.; Dhipika, A.; Vignesh, K.; Supriya, P. Path Planning for Mobile Robots Using Deep Learning Architectures. In Proceedings of the 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India,, 8–9 October 2021; pp. 1–6. [CrossRef]
24. Ganesan, S.; Ramalingam, B.; Mohan, R.E. A Hybrid Sampling-Based RRT* Path Planning Algorithm for Autonomous Mobile Robot Navigation. *Expert Syst. Appl.* **2024**, *258*, 125206. [CrossRef]
25. Ou, Y.; Cai, Y.; Sun, Y.; Qin, T. Autonomous navigation by mobile robot with sensor fusion based on deep reinforcement learning. *Sensors* **2024**, *24*, 3895. [CrossRef]
26. Bartolozzi, C.; Glover, A.; Donati, E. Neuromorphic Sensing, Perception and Control for Robotics. In *Handbook of Neuroengineering*; Thakor, N.V., Ed.; Springer: Singapore, 2022. [CrossRef]

27. Martínez, F.S.; Casas-Roma, J.; Subirats, L.; Parada, R. Spiking Neural Networks for Autonomous Driving: A Review. *Eng. Appl. Artif. Intell.* **2024**, *138*, 109415. [CrossRef]
28. Aitsam, M.; Davies, S.; Di Nuovo, A. Neuromorphic Computing for Interactive Robotics: A Systematic Review. *IEEE Access* **2022**, *10*, 122261–122279. [CrossRef]
29. Neufeld, E.A.; Bartocci, E.; Ciabattoni, A.; Governatori, G. Enforcing Ethical Goals over Reinforcement-Learning Policies. *Ethics Inf. Technol.* **2022**, *24*, 43. [CrossRef]
30. Thangamuthu, M.; Dinesh, T.; Guruchandhramavli, B.; Sanjai, S.; Sheshadhri, B. Mobile Robot Path Planning and Obstacle Avoidance Using Hybrid Algorithm. *Int. J. Inf. Technol.* **2023**, *15*, 4481–4490. [CrossRef]
31. Susrama, G.; Diyasa, M.; Anggraini, T.; Sari, P.; Prasetya, D.A.; Anuar, T.; Kassim, M.; Idhom, M. Implementation of Haversine Algorithm and Geolocation for Travel Recommendations on Smart Applications for Backpackers in Bali. In Proceedings of the 2022 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 16–17 November 2022; pp. 504–508.
32. Anisya, G.; Swara, G.Y. Implementation of Haversine Formula and Best First Search Method in Searching of Tsunami Evacuation Route. *IOP Conf. Ser. Earth Environ. Sci.* **2017**, *97*, 012004. [CrossRef]
33. Movable Type Scripts. Available online: https://www.movable-type.co.uk/scripts/latlong.html (accessed on 28 June 2024).
34. Saputri, T.R.D.; Lee, S.-W. The Application of Machine Learning in Self-Adaptive Systems: A Systematic Literature Review. *IEEE Access* **2020**, *8*, 205948–205967. [CrossRef]
35. Swazinna, P.; Udluft, S.; Hein, D.; Runkler, T. Comparing Model-free and Model-based Algorithms for Offline Reinforcement Learning. *IFAC-PapersOnLine* **2022**, *55*, 19–26. [CrossRef]
36. Yang, Y.; Zhang, S.; Dong, J.; Yin, Y. Data-Driven Nonzero-Sum Game for Discrete-Time Systems Using Off-Policy Reinforcement Learning. *IEEE Access* **2020**, *8*, 14074–14088. [CrossRef]
37. Leike, J.; Krueger, D.; Everitt, T.; Martic, M.; Maini, V.; Legg, S. Scalable Agent Alignment via Reward Modeling: A Research Direction. *arXiv* **2018**, arXiv:1811.07871. [CrossRef]
38. An, C.; Jia, S.; Zhou, J.; Wang, C. Fast Model-Free Learning for Controlling a Quadrotor UAV With Designed Error Trajectory. *IEEE Access* **2022**, *10*, 79669–79680. [CrossRef]