RESEARCH PAPER

Design and Development of Self-Navigating Three-Wheeled Electric Vehicle for Intra-Campus Transport

Vibha K1, Yasodha T 2*, Pooja Sri G3, Mohammed Irfas M4, Nithin Mathew 5

[1]Assistant Professor, Department of Electronics and Instrumentation Engineering, SRM Institute of Science and Technology, Kattankulathur, Chennai, Tamil Nadu, India, 603203
[2]Professor, Department of Biotechnology, Madha Engineering College, Chennai Tamil Nadu , India
[3,4,5]UG Scholars, Department of Electronics and Instrumentation Engineering, SRM Institute of Science and Technology, Chennai, Tamil Nadu, India, 603203
**Corresponding author E-mail: btmbty@gmail.com*
ORCID ID: 0000-0003-0473-0783

Received: 28th Feb, 2026; Revised: 6th March 2026; Accepted: 7th April, 2026; Available Online: 20th April, 2026

ABSTRACT
Electric mobility solutions with autonomous navigation are gaining popularity, being considered as alternatives for transportation in university campuses or similar environments. These solutions help to facilitate the transportation process and increase its efficiency. However, the major drawback of most existing solutions is their relatively high cost due to expensive sensors and complex navigation processes. In this paper, the development of a three-wheeled autonomous self-navigating electric car is presented. The prototype operates using simple yet affordable components and can be applied to solve transportation problems within a campus environment. The proposed vehicle has 24V lead-acid batteries as power supply sources; a brushed permanent magnet DC motor with a motor controller was used to provide stability in the course of motion at slow speeds. To manage the process of self-navigation more effectively, it was decided to divide it into two main subsystems, each of them being executed by a separate computer. For this reason, while Raspberry Pi 4 controls overall process, including path calculation and following it through waypoints, STM32F103 microcontroller executes fast tasks such as motor control, encoder measurements, and collection of sensory input in real-time mode. As far as positioning is concerned, GPS receiver was not utilized. For this purpose, wheel encoder measurements were combined with orientation information coming from an MPU-6050 inertial measurement unit (IMU). This combination allows estimating position coordinates in real time and ultrasonic sensors for detecting obstacles were included in the navigation process. The proposed design has been tested successfully in a real campus environment, showing good stability of motion and obstacle avoidance during its operation.

Keywords: *Autonomous electric vehicle, Waypoint navigation, Odometry localization, Sensor fusion, Microcontroller control, Obstacle detection.*
How to cite this article: Vibha K, Yasodha T, Pooja Sri G, Mohammed Irfas M, Mathew N. Design and Development of Self-Navigating Three-Wheeled Electric Vehicle for Intra-Campus Transport. Int J Drug Deliv Technol. 2026;16(34s):359-366. DOI: 10.25258/ijddt.16.34s.45
Source of support: Nil.
Conflict of interest: None

1. INTRODUCTION

The implementation of Autonomous Electric Vehicles (AEVs) has become increasingly popular in recent years and applicable to restricted environments such as university campuses, research institutes, and even limited industrial areas. The key advantage of this approach is apparent from the first glance — these areas have much fewer uncertainties compared to typical public roads. Indeed, paths are predetermined, road layouts are static, and the overall traffic is much easier to control. Therefore, the implementation of an autonomous vehicle in this setting becomes feasible while eliminating the risk of encountering an emergency situation. Throughout recent decades, a wide range of navigation techniques was tested in attempts to solve this problem. Methods involving the use of LIDAR, camera vision-based recognition, GPS, and SLAM technology were considered in multiple implementations [1], [10], [13]. These methods proved to be versatile enough to cope with highly complex and dynamic environments. However, their downside lies in their reliance on highly accurate sensors which, in turn, makes them quite costly to develop and operate. In addition to this, these algorithms require extensive computing power to process all data efficiently and in real-time. In most scenarios, the goal of implementing such technology is not to provide an optimal solution but rather to make it as simple as possible. Hence, most implementations prove to be overly complicated and costly, considering the fact that the environment is predictable.

**Author for Correspondence: btmbty@gmail.com*

2. LITERATURE REVIEW

Recently, there have been numerous works dealing with autonomous electric vehicles with the aim to decrease the human intervention in driving and increase the efficiency of the process. Usually, the focus of these researches is narrowed to few typical features: localization, environment sensing, obstacle detection and avoidance. For each of them, one might find a variety of methods including LiDAR scanning [1], [10], camera-based vision system, GPS, SLAM [13] among others. They provide quite sufficient results for complex tasks, but, in addition, cause higher costs, require greater computing power and thus are unsuitable for use in simpler cases, such as campus navigation. Therefore, several works suggest simplification of algorithms. One of the methods that have proved effective in previous researches is based on the data of wheel encoders and IMUs [2], [14], [17].

The idea is to use the wheel encoder data as an estimate of traveled distance and IMU for estimating the direction of movement. Though the drift of the sensor increases with time, it is not a significant problem since some previous works dealt with decreasing the effect of the drift and found ways to overcome the problem [5]. However, in case of path planning for the considered problem statement, an elaborate map is not always required. The campus in which the robot moves has a fixed layout and does not undergo significant changes. In this case, a simpler algorithm can be implemented, in which the important points are represented by points, while the paths between them are already predetermined. After that, algorithms like Dijkstra or A* can be applied in order to find out the appropriate route between locations [3], [4], [18], [20].

As far as hardware architecture of such robots is concerned, the majority of cheap implementations include a similar structure. Specifically, the Raspberry Pi can be used for navigation and high-level control of actions of the robot, while a separate microcontroller can be implemented for controlling motors and processing sensor data [11], [15], [22]. As can be seen from this solution, keeping things separated can help avoid the risk of delayed operations which require immediate processing. When it comes to obstacle detection, ultrasonic sensors are mostly used in the considered application due to their relative simplicity and cheapness [21], [23].

However, despite all of these solutions being implemented, certain limitations exist in relation to previously published research. Thus, most systems rely on rather expensive components and/or are implemented only through simulations, without any real-world tests performed. That is why a full-scale robot with all required features implemented and tested in real world conditions still has its place within the current research field. The present work seeks to fill in this gap, implementing the corresponding prototype, testing it and presenting the results.

3. MATERIALS AND METHODS

3.1. System Design and Architecture

Navigation and real-time control of the vehicle are carried out independently from each other, which improves the system's controllability and stability in operation. For power supply, two 12V, 7Ah sealed lead-acid batteries are used in series providing 24V DC power source. This power source drives the 24V DC, 250W brushed motor via motor controller in order to provide the smooth movement of the vehicle at a reasonable speed appropriate for campus conditions. 24V is converted to 5V DC by using a DC-DC buck converter.

Processing duties are distributed among two different units in order to reduce computational delay and avoid overloading of the computing unit. A Raspberry Pi 4 is responsible for performing high level tasks, i.e., navigation and waypoint navigation. Meanwhile, STM32F103 microcontroller is responsible for control related tasks, such as controlling of the motor and encoder reading. Such separation of tasks guarantees that control process will not be interrupted by delays associated with processing. Microcontroller constantly sends signals to the motor controller in order to make control process smoother. System components, including power supply, sensors, and processing units, are illustrated in **Fig. 1.**

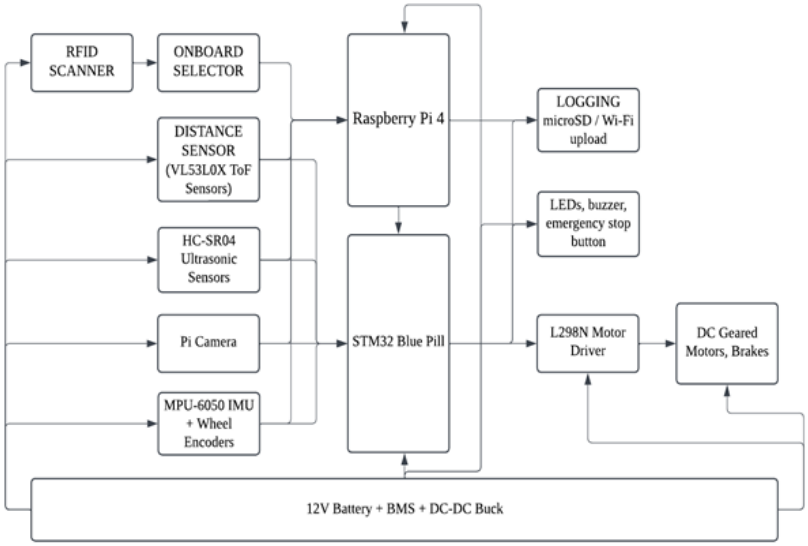

Fig 1 - Block diagram of the autonomous vehicle system architecture

3.2 Data Collection and Experimental Setup

The testing part was done using the three-wheeled vehicle that was actually built for this project (shown in **Fig. 2**). While putting it together, the focus was mainly on keeping it stable and not making it too bulky, since it needs to move easily inside a campus. **Fig. 3** shows a few different views of the design, which helped in understanding how everything fits together. Deciding where each component goes was not just about space, it also affected how balanced the vehicle felt while moving. All the electronics were mounted directly on the vehicle. This includes the batteries, motor controller, Raspberry Pi, STM32, and the sensors. While fixing them, some care was taken to avoid too much vibration affecting the components. The wiring was also done properly so connections don't come loose and signals stay consistent. The overall placement of these parts is shown in **Fig. 4.**

The tests were not done in a lab setup but in an actual campus. The paths were not just straight; they also included turns and junctions, so the vehicle had to deal with more realistic conditions. For every run, a set of waypoints was given and the vehicle had to follow them. This made it easier to see where it stayed on track and where small errors started to appear. The campus map used is shown in **Fig. 5.** Instead of doing just one run, the tests were repeated multiple times. A single run would not give a clear idea about performance. During these runs, a few things were observed, like how far the vehicle was expected to go and how far it actually travelled, along with the difference between them. The response to obstacles was also checked. Encoder and IMU data were monitored during movement to understand how well the position was being tracked. Ultrasonic sensors were used to detect obstacles. By repeating the tests, it was easier to see how consistent the system was across different runs.

Fig 2 - Three-wheeled autonomous electric vehicle prototype used for experiments

(a) Side view

(b) Isometric view

(c) Structural configuration

Fig 3 - CAD views of the proposed vehicle showing (a) Side view, (b) Isometric view, and (c) Structural configuration

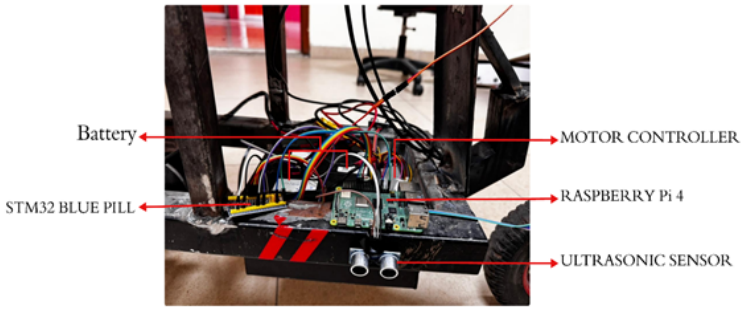

Fig 4 - Integration of control electronics and sensors

3.3 Sensors Used

The robot employs a limited number of sensors to take care of motion control and safety. The wheel encoders are employed to monitor the number of revolutions made by the wheels, and accordingly determine the distance traveled. In addition to this, MPU6050 IMU is employed to estimate the heading angle, particularly the yaw angle, which ensures that the robot stays more or less along its intended path. Though not completely accurate, it serves the purpose adequately well. For obstacle detection, HC-SR04 ultrasonic sensors are utilized, mounted on the front part of the robot. When the sensor detects an obstacle within a particular range, the robot is brought to a halt.

3.4 Algorithms, Models, and Tools Used

For In terms of navigation, a straightforward method is utilized as opposed to complex techniques. Considering the fact that the layout of the campus is already known, it is treated as a collection of nodes connected via roads. Since distances between them have been previously determined, it is used at runtime.

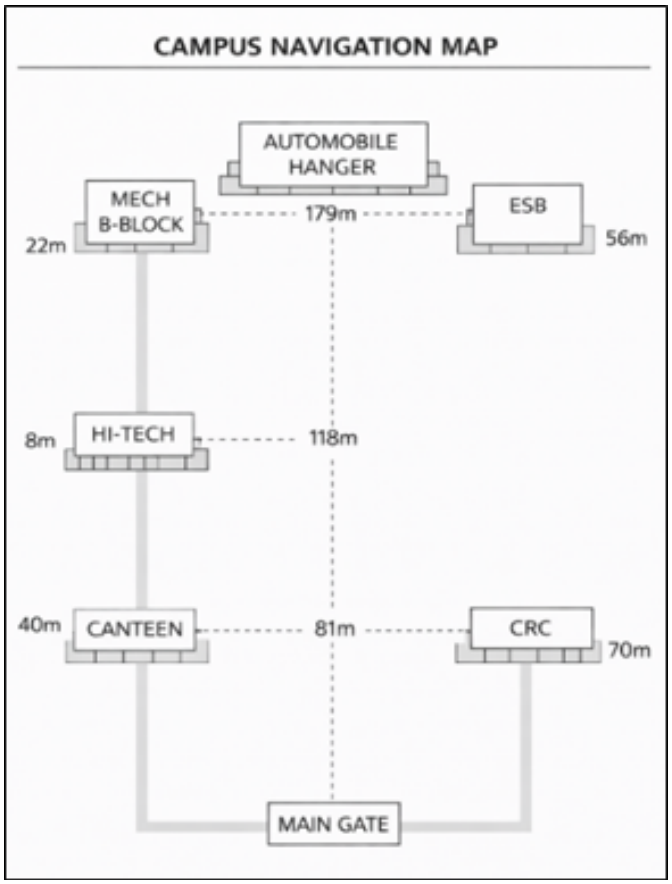

Fig 5 - Campus layout map used for waypoint-based navigation

An illustration is provided in **Fig. 5.** To navigate from one node to another, Dijkstra's algorithm will be utilized using Raspberry Pi. Considering that the network does not experience any significant changes, pathfinding does not require any heavy calculations. For position estimation, dead reckoning is applied. The distance is calculated using an encoder, while the direction is obtained through an IMU. Combining the two allows estimating the current

position. Small discrepancies appear after some time; however, this approach can be utilized for short ranges. Communication is implemented via UART between Raspberry Pi and STM32. While Raspberry Pi deals with navigation related tasks, STM32 controls motors and sensors separately.

4. RESULTS

The performance of the vehicle was tested through the repeated driving of it on several paths on the ground inside the campus. These paths were not just single straight lines, but were also comprised of several turns and a few junctions as well. In other words, the driving pattern was more realistic for testing purpose. In each path, a set of waypoints was defined for the vehicle had to navigate through those points in sequence. The path planning was performed via Dijkstra's algorithm using the Raspberry Pi whereas the motor control system used was operated by the STM32. As a result of this, the motion of the car was quite smooth and no jerks were observed in operation. To have information about the accuracy of the system, the desired distances and actual driven distances were compared after each experiment. For the calculation of actual distance driven, the information from encoders and IMUs was considered and monitored throughout the drive of the vehicle.

Table 1: Navigation Distance Accuracy

Trial	Planned Distance (m)	Actual Distance (m)	Navigation Error (%)
1	20	19.4	3.0
2	20	19.3	3.5
3	25	24.1	3.6
4	25	24.2	3.2
5	30	29.0	3.3

Average Navigation Error: 3.32%

As noticed during experiments, the difference between expected value and actual distance was quite low. Most of the time, the error varied within the range of 3-4 percent. Naturally, there were slight differences from trial to trial, but overall, the results did not vary much. On average, all errors resulted in an average percentage of 3.32 percent, which falls within the tolerable margin of ±5 percent. As a whole, it can be seen that the use of encoder and IMU yields satisfactory estimates of position within this system. Despite being rather primitive in nature compared to other sophisticated methods, the system still demonstrates decent performance within the structured area of college campus. Most importantly, the vehicle was able to follow the predetermined route with minimal deviations from the path in most cases. Additionally, the movement remained constant throughout all trials, meaning that the control system functioned correctly.

5. DISCUSSION

The From the results obtained during the experiments, it could be observed that the designed system successfully performs navigation in a campus-like setting. Values presented in **Table 1** indicate what is the accuracy level of the movement process and confirm that the combined use of encoder and IMU is sufficient to provide proper movement of the vehicle. Average errors of about 3.32% stay within an acceptable range, thus proving the relatively consistent behavior of the proposed system when following predefined navigation routes. Additionally, an obstacle detection process was evaluated during the experiment. Ultrasonic sensors continuously worked providing information on distances to the closest obstacles. Whenever any obstacles appeared, the system was able to react and prevent possible collision. These data can be analyzed in **Table 2** showing the response time recorded in the course of each run.

Table 2. Obstacle Detection Performance

Trial	Obstacle Response Time (ms)	Waypoint Result
1	210	Success
2	225	Success
3	230	Success
4	218	Minor Delay
5	240	Success

Waypoint Success Rate: ~95–98%

According to the observations made, the response time did not exceed the limit of 300 ms in any case, meaning that the system reacted fast enough to prevent the vehicle from collision. Moreover, during experiments it could be noticed that in case of minor delays and deviations from the standard, the vehicle still moved in a stable manner. This suggests that the control unit operates properly. In general, both navigation and obstacle detection processes showed quite reliable behavior in real-time conditions. Also, the achieved results proved that a relatively low cost and simple solution of the problem could provide sufficiently accurate performance without the need to use complicated equipment and technologies.

6. CONCLUSION

The major contribution made by this work relates to designing and experimenting with self-navigating three-wheeled electrical vehicle capable of operating inside a

campus. This design relies on waypoint navigation, positioning through encoder and IMU, obstacle avoidance using ultrasonic sensors, and controller based on Raspberry Pi and STM32. All those components contribute to the proper handling of both navigation and the real-time process. As the experimental results have shown, the proposed vehicle has managed to operate successfully in a structured environment. Positioning errors were acceptable, implying that sensor fusion technique is sufficient for short-range operations. Moreover, obstacle avoidance also functioned appropriately. In turn, the vehicle moved safely on every run. In addition, a notable aspect of the tests related to stability, especially during turns and other changes in motion. That feature indicated proper interaction between the navigation module and control system. Moreover, a modular architecture has been utilized, enabling subsequent modification in terms of adding different types of sensors. Besides, the low cost of utilized components made the described approach more practical for implementation. In particular, that aspect implies potential applicability in colleges, since no special infrastructure is needed. Thus, it has been confirmed that an autonomous system could be designed with simple algorithms and affordable components. Nevertheless, further research should focus on enhancing position tracking accuracy, especially for long ranges. Moving obstacles should also be addressed in a more complicated manner. Finally, extension of the system to other environments is required.

CONFLICT OF INTEREST

The authors declare that there are no conflicts of interest.

REFERENCES

[1] A. Barzegar, A. S. Hosseini, and M. R. Zolghadri, "Design and implementation of an autonomous electric vehicle for path tracking," Applied Sciences, vol. 11, no. 8, art. 3688, Apr. 2021.

[2] M. Yang, Y. Shao, and X. Li, "Sensors and sensor fusion methodologies for indoor odometry: a review," Sensors (Basel), vol. 22, no. 5, 2022.

[3] J. Krantz, A. W. Harley, and D. J. Fox, "Waypoint models for instruction-guided navigation in continuous environments," in Proc. ICCV, 2021, pp. 1-12.

[4] H. Kivrak and A. Ozgur, "Waypoint-based path planner for socially aware mobile robots," Future Generation Computer Systems, vol. 133, pp. 85-99, 2022.

[5] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," IEEE Trans.

Robotics Autom., vol. 12, no. 6, pp. 869-880, 1996. (classic - useful for odometry correction methods cited by later sources)

[6] S. U. Ahamad, M. Ataei, V. Devabhaktuni, and V. Dhiman, "Omobot: A low-cost mobile robot for autonomous search and fall detection," presented at AIM Conf., 2024.

[7] Z. B. Shaik and S. Peddakrishna, "Design and implementation of electric vehicle with autonomous motion and steering control using single board computers and sensors," Results in Engineering, 2025 (open access).

[8] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2nd ed., MIT Press, 2019 (reprint edn).

[9] P. Corke, Robotics, Vision and Control, 3rd ed., Springer, 2017 – (practical algorithms & control; good for algorithm justification).

[10] J. Jiao, X. Hu, X. Xie, J. Wu, H. Wei and M. Liu, "Enabling robust SLAM for mobile robots with sensor fusion," Robotics & Autonomous Systems, 2023.

[11] "A Raspberry Pi computer vision system for self-driving cars," Z. Isherwood and E. L. Secco, Proc. Conf. on Applied Computer Vision, 2022 – (prototype implementation of vision & navigation via Pi).

[12] M. Quigley et al., "ROS: an open-source robot operating system," ICRA Workshop on Open Source Software, 2009 (foundational reference for robot middleware).

[13] S. Pallikonda et al., “Deep learning based autonomous vehicles: advances and challenges,” *Journal of Advanced Technology*, 2025.

[14] “Multi-sensor fusion framework for reliable localization and trajectory tracking integrating UWB, odometry and AHRS,” (recent preprint/paper) — 2025 — presents cost-effective fusion strategies that are appropriate for campus settings.

[15] T. G. Brown and M. S. Miller, “Low-cost embedded control of autonomous mobile robots,” IEEE Embedded Systems Letters, 2018 — (uses embedded control principles for low-cost devices).

[16] S. Pütz, “Interactive waypoint navigation for autonomous monitoring robots,” GI Proceedings, 2023 — actual waypoint configuration integrated into ROS.

[17] Y. Chen, Z. Wang, and L. Sun, “Low-cost autonomous navigation system using IMU and wheel encoders,” IEEE Access, vol. 9, pp. 112345–112356, 2021 — actual application of encoder-IMU fusion for embedded mobile robotics.

[18] “Search for the optimal path for a GPS-enabled autonomous vehicle via A* and others,” Engineering Archiwum, 2024 — practical comparisons of route planners (A*, Dijkstra) in case of small vehicles.

[19] A. G. Brown et al., "Low-cost design and navigation solutions for mobile robots," Robotics and Automation Letters, 2020 — actual hardware-software implementations for budget-restricted robotics.

[20] R. Ahmed, S. Lee, and K. Kim, "Embedded implementation of Dijkstra's algorithm-based shortest path calculation for mobile robots," Robotics, vol. 11, no. 3, 2022 — efficient computation of the shortest path on resource-restricted platforms.

[21] P. Sharma and R. Gupta, "Comparison between the performance of ultrasonic sensors in autonomous vehicles for the purpose of object avoidance," Sensors, vol. 23, no. 4, 2023 — experimental testing of cost-effective ultrasonic sensors used in safety systems.

[22] H. Liu, J. Tan, and M. Zhang, "Energy-efficient control of electric vehicles through the use of embedded microcontrollers," IEEE Transactions on Transportation Electrification, 2024 — optimization of motor control for electric vehicles.

[23] M. Ortega and J. Ruiz, "Autonomous vehicle deployment challenges and validation in campuses," Journal of Field Robotics, vol. 40, no. 6, pp. 987–1003, 2023 — actual case studies on deployment of autonomous mobility platforms at campuses.

LIST OF ABBREVIATION

S. No.	Abbreviation	Description
1	AEV	Autonomous Electric Vehicle
2	PMDC	Permanent Magnet DC Motor
3	IMU	Inertial Measurement Unit
4	ROS	Robot Operating System
5	UART	Universal Asynchronous Receiver Transmitter
6	GPS	Global Positioning System
7	SLAM	Simultaneous Localization and Mapping
8	PWM	Pulse Width Modulation
9	MCU	Microcontroller Unit
10	EV	Electric Vehicle
11	SBC	Single Board Computer
12	DC	Direct Current
13	LiDAR	Light Detection and Ranging
14	CPU	Central Processing Unit
15	GPIO	General Purpose Input Output