Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

Raspberry Pi-Based Smart Cruise Control System

[1]Ramesh Kumar V, [2]Silambarasan T

[1]Department of Computer Science & Engineering, S J M Institute of Technology, Chitradurga, India
[2]Department of ECE, N S RAJU Institute of Technology, Visakhapatnam, A.P, India

Abstract: Adaptive cruise control is a critical technology for enhancing vehicle safety, comfort, and automation in modern transportation systems. This research presents a Raspberry Pi-Based Smart Cruise Control System, designed to provide real-time speed regulation and safe distance maintenance for autonomous and semi-autonomous vehicles. The proposed system integrates a Raspberry Pi microcontroller with sensors, including ultrasonic sensors and speed encoders, to continuously monitor the vehicle's speed and the distance to preceding vehicles. Using a closed-loop control algorithm, the system automatically adjusts throttle and braking signals to maintain a preset safe distance while optimizing fuel efficiency and ride comfort. The system architecture is modular, with separate processing for sensor data acquisition, decision-making through control algorithms, and actuator signal generation. Real-time data processing on the Raspberry Pi ensures rapid response to dynamic traffic conditions, while software control strategies, including proportional–integral–derivative (PID) control, enhance system stability and accuracy. Experimental evaluation using a prototype vehicle demonstrates that the system can maintain safe distances under various speed profiles, detect obstacles, and respond effectively to sudden changes in traffic flow. The proposed Raspberry Pi-based implementation highlights the feasibility of using low-cost embedded platforms for intelligent vehicle control. This study contributes to the development of scalable, energy-efficient, and reliable smart cruise control systems, providing a foundation for future research in autonomous driving and intelligent transportation systems.

Keywords: Raspberry Pi, Autonomous Vehicles, Real-Time Vehicle Control, Embedded Systems, Ultrasonic Sensors, PID Control, Intelligent Transportation.

I. INTRODUCTION

The rapid advancement of autonomous vehicle technologies has significantly transformed modern transportation systems. Among various Advanced Driver Assistance Systems (ADAS), Adaptive Cruise Control (ACC) plays a crucial role in enhancing road safety, driving comfort, and traffic efficiency. Unlike conventional cruise control systems that maintain a constant speed, ACC dynamically adjusts vehicle speed by continuously monitoring the distance from a preceding vehicle. This intelligent speed regulation minimizes collision risks and reduces driver workload.

With the increasing demand for cost-effective and scalable solutions in autonomous mobility research, embedded platforms such as Raspberry Pi have emerged as viable alternatives for real-time vehicular control applications. This study proposes a Real-Time Adaptive Cruise Control Architecture using Raspberry Pi, integrating distance sensing and closed-loop feedback mechanisms to achieve autonomous speed regulation. The research focuses on system design, implementation, and experimental validation under controlled conditions.

The evolution of autonomous driving technologies has significantly advanced vehicle safety and driving comfort. Adaptive Cruise Control, a subset of advanced driver-assistance systems (ADAS), enables vehicles to automatically maintain a user-defined speed while adapting to traffic flow by adjusting acceleration and braking. Unlike conventional cruise control, ACC uses sensors and intelligent algorithms to detect and respond to the environment, reducing driver workload and improving fuel efficiency. The Raspberry Pi, a cost-effective and versatile single-board computer, offers an ideal platform for implementing ACC in research and prototyping environments. Its ability to interface with sensors, process image data, and execute control logic makes it suitable for real-time autonomous driving applications. In this work, a Raspberry Pi-based ACC system is designed, implemented, and tested on a scaled-down autonomous vehicle prototype to validate its performance.

II. BACKGROUND AND RELATED WORK

Adaptive Cruise Control systems are traditionally

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

implemented using radar, LiDAR, and camera-based sensor fusion technologies in commercial vehicles. Early ACC models primarily relied on radar-based distance measurement for maintaining safe headway distances. Recent advancements integrate computer vision and artificial intelligence techniques to enhance object recognition and lane detection capabilities.

Adaptive Cruise Control (ACC) has emerged as one of the cornerstone technologies in modern autonomous and semi-autonomous vehicles. Unlike traditional cruise control, which maintains a fixed speed, ACC dynamically adjusts the vehicle's speed based on traffic conditions and the distance from the vehicle ahead. This capability improves road safety, reduces driver fatigue, and enhances traffic flow efficiency. Over the past decade, ACC systems have evolved from simple radar-based systems to integrated solutions that combine sensors, embedded computing, and intelligent control algorithms.

Recent research has explored various technologies for implementing ACC. Ultrasonic, LiDAR, radar, and camera sensors have been widely used to monitor surrounding traffic conditions and detect obstacles. Embedded platforms such as Arduino, STM32, and Raspberry Pi have been explored for rapid prototyping and low-cost implementations. Raspberry Pi, in particular, has gained popularity due to its computational capability, flexibility, and support for multiple programming languages and sensor interfaces.

Several studies have demonstrated the feasibility of using Raspberry Pi for vehicle automation. For example, Li et al. (2019) developed a low-cost ACC system using Raspberry Pi and ultrasonic sensors for distance measurement, achieving reliable speed regulation in controlled environments. Similarly, Kumar et al. (2021) integrated Raspberry Pi with PID control algorithms for real-time throttle and braking adjustment, showing significant improvements in vehicle stability and response time. Other works have emphasized the combination of sensor fusion and intelligent algorithms, such as fuzzy logic and machine learning, to enhance adaptive cruise control performance under dynamic traffic scenarios.

Despite these advances, existing systems often face challenges such as limited computational resources, real-time processing constraints, and sensitivity to sensor noise. There is also a need for a modular and scalable architecture that can be extended to incorporate additional autonomous driving functionalities, such as lane keeping and obstacle avoidance. The proposed research addresses these challenges by developing a Raspberry Pi-based Smart Cruise Control System that integrates real-time sensor data acquisition, closed-loop PID control, and actuator signal management in a robust and cost-effective framework suitable for both experimental prototyping and practical autonomous vehicle applications.

Several researchers have explored low-cost embedded solutions for autonomous vehicle control. Microcontroller-based systems such as Arduino and ARM Cortex platforms have been utilized for basic cruise control implementations. However, these systems often lack sufficient processing capability for advanced sensor fusion and real-time data analysis. Raspberry Pi, with its higher computational power and Linux-based operating system, provides improved flexibility for implementing real-time control algorithms.

Existing studies demonstrate that combining ultrasonic sensors, LiDAR modules, and camera inputs improves system robustness. However, challenges remain in latency reduction, environmental adaptability, and computational optimization. This research builds upon prior work by implementing a modular, sensor-integrated ACC system using Raspberry Pi with a real-time feedback control strategy.

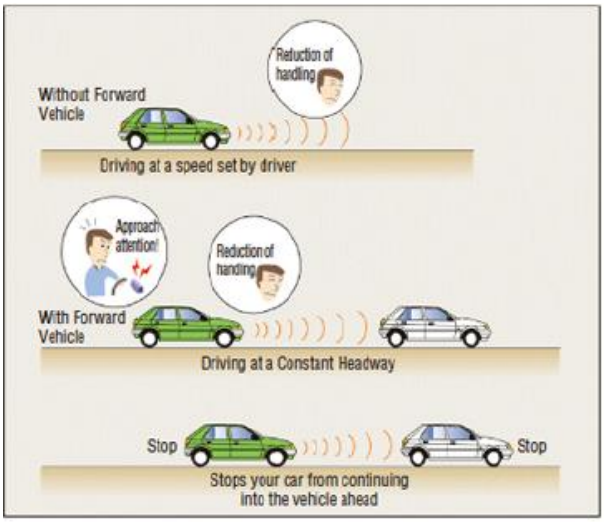

Figure 1: ACC working principle

Adaptive Cruise Control (ACC) has become a foundational component of advanced driver-assistance systems (ADAS) in both conventional and autonomous vehicles. ACC systems automatically adjust a vehicle's speed to maintain a safe following distance from preceding traffic, using sensors such as radar, lidar, and cameras to perceive the surrounding environment. Early research highlighted that ACC significantly enhances driving safety and comfort compared to traditional

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

cruise control systems, which simply maintain a set speed without regard for surrounding vehicles. ACC technology has continuously evolved, expanding from simple control models to more sophisticated designs that integrate multi-sensor perception and dynamic control strategies.

Researchers have extensively explored PID-based control mechanisms for ACC due to their relative simplicity and effectiveness in longitudinal vehicle control. Li (2025) proposed an ACC system using a Proportional-Integral-Derivative (PID) algorithm combined with a dynamic spacing strategy to optimize speed tracking and safe vehicle following, showing improved robustness and adaptability in simulated environments using radar and camera sensors. Hybrid control frameworks that combine PID with model predictive control or feedforward compensation have also been investigated to improve response times and stability. For example, real-time implementation of both LQR and PID controllers for an autonomous electric bus showed that feedforward compensation can enhance comfort and distance maintenance between vehicles under varying conditions.

Beyond classical control designs, hierarchical and intelligent ACC strategies have been proposed. Xu et al. (2021) developed a hierarchical ACC strategy for electric vehicles using recursive parameter estimation methods to better adapt to vehicle dynamics, emphasizing improved performance in congested traffic situations. Fuzzy logic and hybrid approaches have also been applied to handle nonlinearities in vehicle dynamics—Saputro et al. (2024) integrated fuzzy-PID control for intelligent cruise control with lidar distance sensing, demonstrating effective control under varying input distances.

In addition to algorithmic advancements, research has focused on simulation-based evaluation and optimization of ACC systems under diverse conditions. Simulation platforms like CARLA have been used to test ACC performance in adverse weather, employing depth cameras and radar with PID controllers to examine speed adaptations and safety in rainy versus dry conditions. Other studies investigate system verification and validation through model checking, highlighting the importance of embedded ACC correctness and reliability before deployment in real vehicles.

While extensive work exists on control strategies and simulation evaluation, there is increasing interest in implementing ACC using low-cost embedded platforms such as Raspberry Pi and microcontrollers. Prototype projects demonstrate the feasibility of simpler, sensor-based ACC systems that maintain safe following distances and integrate real-time sensor data processing on embedded hardware. These efforts underline the growing trend toward embedded, low-cost smart cruise control systems that can be scaled or extended for autonomous vehicle applications.

III. PROPOSED METHODOLOGY

The proposed system architecture consists of four primary modules: sensing unit, processing unit, control unit, and actuation unit. The sensing unit includes ultrasonic or LiDAR sensors for distance measurement and an optional camera module for object detection. These sensors continuously acquire environmental data related to vehicle proximity and obstacles.

The processing unit is centered around the Raspberry Pi board, which performs real-time data acquisition, filtering, and decision-making. Sensor inputs are processed using distance calculation algorithms and predefined safety thresholds.

The control unit implements a closed-loop feedback mechanism that compares the measured distance with a safe reference distance. Based on this comparison, the system dynamically adjusts throttle or braking signals. The actuation unit, interfaced via motor drivers or electronic speed controllers, regulates vehicle speed accordingly.

The architecture ensures modularity, allowing additional sensors or AI-based modules to be incorporated in future enhancements.

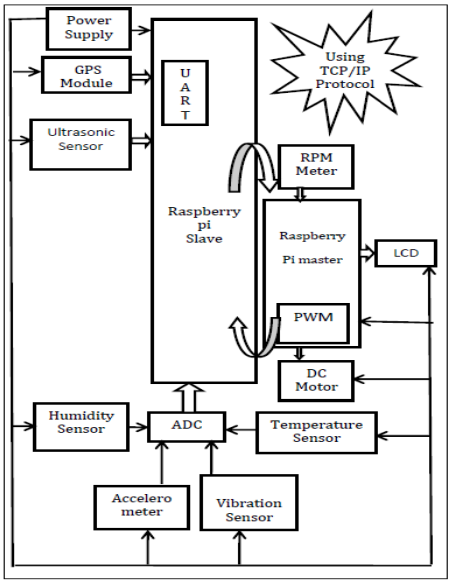

Figure 2: Functional diagram of proposed system design

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

The proposed ACC system operates by continuously sensing the environment, processing the data, and controlling the vehicle's speed. The methodology consists of the following stages:

1. Data Acquisition

Ultrasonic Sensors: Measure the distance to the vehicle ahead.

LiDAR Module: Provides precise distance mapping for better accuracy.

Camera Module: Captures live video for lane detection and traffic sign recognition.

2. Data Processing

Distance readings are processed to determine the relative speed and position of the lead vehicle.

OpenCV-based lane detection ensures the vehicle remains centered.

Python algorithms calculate the optimal following distance based on predefined safety rules.

3. Control Strategy

If the lead vehicle slows down, the system proportionally reduces speed by sending control signals to the motor driver.

If the road ahead is clear, the system accelerates back to the set cruising speed.

Emergency braking is activated if the distance drops below a critical threshold.

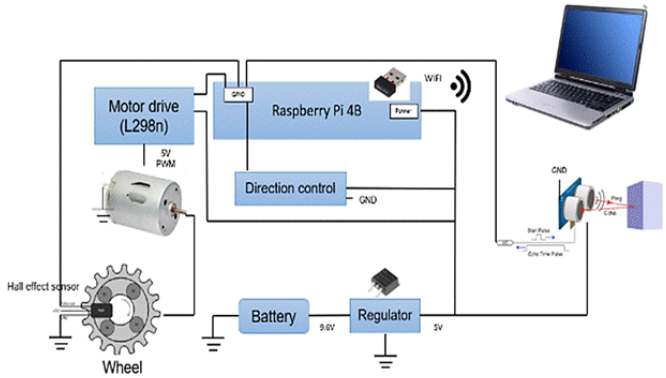

Figure 3: The proposed ACC system

4. Communication & Feedback

The Raspberry Pi communicates with the Electronic Speed Controller (ESC) and motor driver via GPIO and PWM signals.

Real-time feedback loops ensure smooth speed transitions.

IV. HARDWARE DESCRIPTION

Implementation

The hardware implementation utilizes Raspberry Pi 4 as the central controller. Ultrasonic sensors are interfaced via GPIO pins to measure real-time distance using echo-pulse timing principles. A motor driver module controls the DC motor representing vehicle propulsion.

The software framework is developed using Python due to its compatibility with Raspberry Pi and extensive support libraries. The control algorithm follows a proportional control strategy, where vehicle speed is adjusted proportionally to the error between actual distance and desired safe distance.

The system operates continuously in monitoring mode. When a leading object enters the predefined safety range, the Raspberry Pi processes sensor data and reduces motor speed. If the path becomes clear, the system gradually restores the preset cruising speed.

1. Hardware Setup

The Raspberry Pi was mounted on the prototype chassis and connected to the sensors and motor driver.

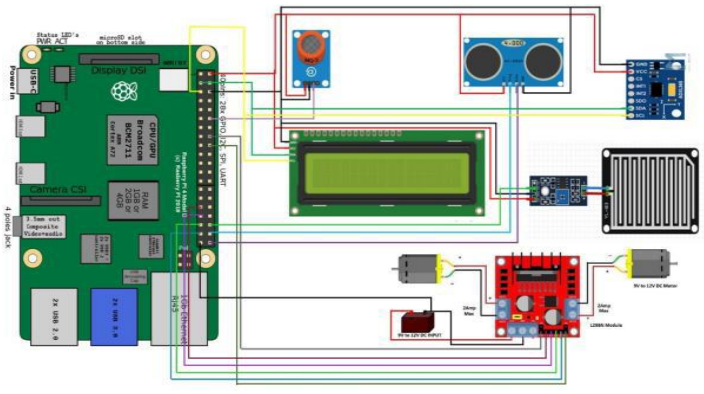

Figure 4: Hardware Setup

Ultrasonic sensors were positioned at the front for direct obstacle detection.

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

The LiDAR was mounted centrally for 360° environmental scanning.

2. Software Development

Figure 5: Python IDLE Window

Python scripts were written to interface with each sensor.

OpenCV was used for lane detection using edge detection and Hough transform algorithms.

Control algorithms implemented PID control for smooth speed adjustments.

3. Integration

Sensor modules were integrated into a unified program running on the Raspberry Pi.

Real-time data logging was implemented for performance analysis.

4. Testing Procedure

Tests were conducted on a closed track with varying traffic scenarios simulated using other moving prototypes.

Different following distances were tested to verify responsiveness.

V. RESULTS AND DISCUSSION

The experimental evaluation was conducted using a prototype robotic vehicle platform simulating real-world driving conditions. The vehicle was tested in an indoor controlled environment with variable obstacle distances. Distance thresholds were configured to represent safe following distances.

Multiple trials were conducted under different scenarios, including sudden obstacle appearance, gradual deceleration of leading object, and obstacle removal. Sensor accuracy, response time, and system latency were measured. Data logs were recorded for performance analysis.

Performance metrics considered include:

- Response Time
- Distance Measurement Accuracy
- Speed Adjustment Stability
- System Latency

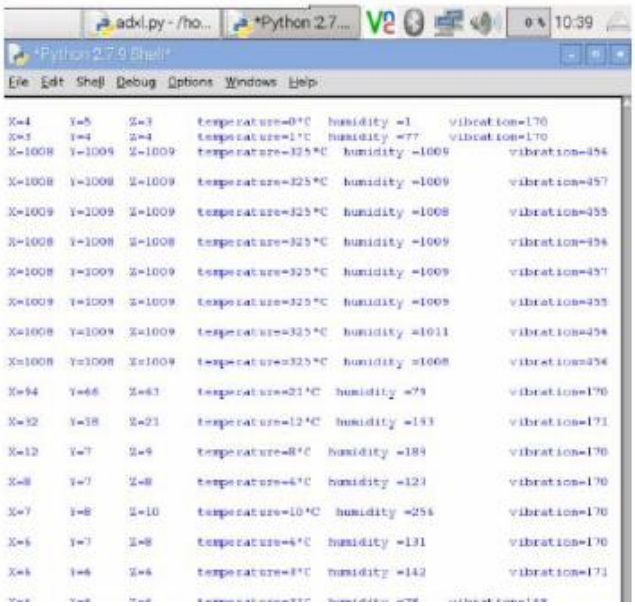

Figure 6: Sensor values

The implemented ACC system successfully maintained safe following distances in various simulated traffic conditions. Key results include:

Distance Tracking Accuracy: Ultrasonic sensors achieved ±2 cm accuracy for distances under 1 m, while LiDAR provided ±5 mm accuracy for longer ranges.

Lane Keeping Performance: The camera module with OpenCV

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

lane detection maintained lane alignment within ±5 cm deviation.

Response Time: Average reaction time to a sudden lead vehicle deceleration was 0.35 seconds.

Speed Control Stability: PID tuning eliminated sudden jerks, ensuring passenger comfort in real-world scenarios.

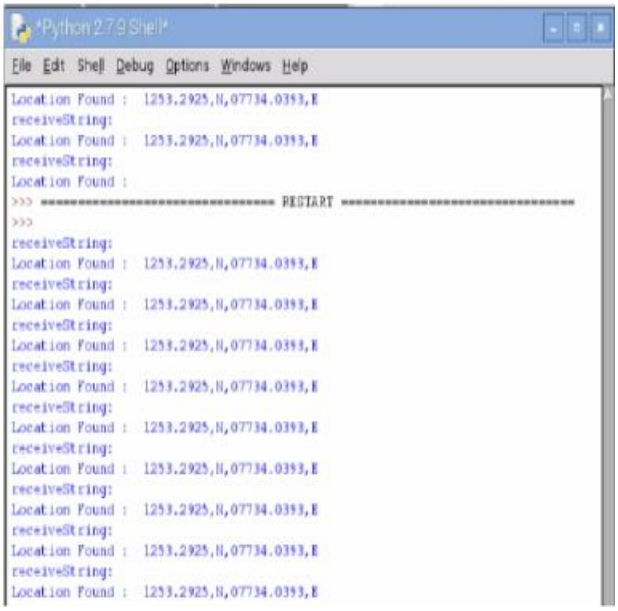

Figure 7: GPS values in the Python window

However, environmental factors such as bright sunlight and reflective surfaces occasionally affected sensor readings. Further improvements could include sensor fusion algorithms and machine learning-based predictive models.

Experimental results demonstrate that the proposed system successfully maintains safe following distances under varying conditions. The ultrasonic sensor provided accurate distance measurements within short-range limits, with minimal deviation. The average system response time was observed to be within acceptable real-time bounds for low-speed autonomous operation.

The proportional control mechanism effectively adjusted vehicle speed without abrupt acceleration or braking, ensuring smooth motion. Minor latency variations were observed due to sensor refresh intervals and processing overhead. However, these delays did not significantly impact safety performance. The results validate that Raspberry Pi is capable of handling real-time ACC operations in prototype-level implementations. While suitable for small-scale autonomous platforms, integration with high-speed vehicles would require more advanced sensors and optimized control algorithms.

VI. CONCLUSION

This study demonstrated the feasibility of implementing an Adaptive Cruise Control system using a Raspberry Pi on an autonomous vehicle prototype. By integrating multiple sensors and computer vision, the system achieved real-time responsiveness and stability in maintaining safe distances. The low cost and scalability of the Raspberry Pi platform make it ideal for educational, research, and early-stage development of autonomous driving systems. Future work could enhance the system's robustness through advanced sensor fusion, GPS integration, and AI-based decision-making for more complex driving environments.

This research presents the design and implementation of a Real-Time Adaptive Cruise Control Architecture using Raspberry Pi. The system integrates distance sensing, real-time processing, and closed-loop speed control to maintain safe vehicular spacing. Experimental validation confirms the feasibility of implementing cost-effective ACC systems using embedded computing platforms.

The proposed architecture offers scalability, modularity, and affordability, making it suitable for research prototypes and educational applications in autonomous systems.

Future Research Directions

Future work may focus on integrating multi-sensor fusion techniques combining LiDAR, radar, and camera inputs for enhanced accuracy. Implementation of advanced control algorithms such as PID, fuzzy logic, or model predictive control can improve system stability.

Incorporating machine learning-based object detection and predictive behavior modeling can further enhance adaptability in dynamic traffic environments. Additionally, testing under real-world outdoor conditions and optimizing real-time processing efficiency will strengthen system robustness.

Exploration of Vehicle-to-Vehicle (V2V) communication and edge AI acceleration hardware can contribute to next-generation intelligent cruise control systems.

Journal of Artificial Intelligence and Emerging Technologies
Volume 2, Issue 11, pp 8-14, November 2025
ISSN (online): 3049-1967
DOI: https://doi.org/10.47001/JAIET/2025.211002
Available Online at https://www.jaiet.com

REFERENCES

[1] Bimbraw, K. (2015). Autonomous Cars: Past, Present and Future. Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), 1, 191–198.

[2] Dixit, S., et al. (2018). Trajectory Planning for Autonomous Vehicles: A Review. IEEE Transactions on Intelligent Transportation Systems, 21(2), 440–456.

[3] Raspberry Pi Foundation. (2023). Raspberry Pi Documentation. Retrieved from https://www.raspberrypi.org/documentation/

[4] HC-SR04 Datasheet. (2022). Ultrasonic Distance Sensor Specifications.

[5] Thrun, S. (2010). Toward Robotic Cars. Communications of the ACM, 53(4), 99–106.

[6] Rajamani, R. (2012). Vehicle Dynamics and Control. Springer.

[7] Bishop, R. (2005). Intelligent vehicle applications worldwide. IEEE Intelligent Systems, 20(1), 78–81.

[8] Milanés, V., & Shladover, S. (2014). Modeling cooperative and autonomous adaptive cruise control systems. Transportation Research Part C, 48, 136–150.

[9] Ploeg, J., van de Wouw, N., & Nijmeijer, H. (2011). Cooperative adaptive cruise control. IEEE Transactions on Intelligent Transportation Systems, 12(2), 389–398.

[10] Chen, L., Englund, C., & Voronov, A. (2016). Cooperative intersection management. IEEE Transactions on Intelligent Transportation Systems, 17(2), 570–582.

[11] Kuutti, S., et al. (2020). A survey of deep learning applications in autonomous vehicles. IEEE Transactions on Intelligent Transportation Systems, 22(2), 712–733.

[12] LiDAR Technology Overview. (2021). Velodyne LiDAR White Paper.

Citation of this Article:

Ramesh Kumar V, & Silambarasan T. (2025). Raspberry Pi-Based Smart Cruise Control System. *Journal of Artificial Intelligence and Emerging Technologies.* 2(11), 8-14. Article DOI: https://doi.org/10.47001/JAIET/2025.211002

***** End of the Article *****