Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 46 No. 04 (2025)

Real-Time Traffic Signal Detection and Navigation in Raspberry Pi Autonomous Cars- A Low-Cost Approach to Self-Driving Technology

[1]Niranjan Murthy, [2]B P Harichandra, [3] Balasubramanya H S, [4]Prajwal J S, [5]Mayur P, [6]Syed Zaid Raza

[1 & 2]Associate Professor, Department of Mechanical engineering, Ramaiah Institute of Technology, Bangalore, Karnataka, India 560097

[3]Assistant Professor, Department of Mechanical engineering, Ramaiah Institute of Technology, Bangalore, Karnataka, India 560097

[4, 5, 6 & 7]Department of Mechanical engineering, Ramaiah Institute of Technology, Bangalore, Karnataka, India 560097

Abstract

This paper presents the design and implementation of a simple self-driving car system inspired by recent progress in autonomous vehicle technology. The vehicle is capable of detecting road signals and performing accurate turning actions. The system connects the car's frame to a computing system via Wi-Fi, allowing real-time video frame processing. For real-world use, the analysis unit can be embedded on board the vehicle. The system effectively delivers precise and dependable decision-making for navigation.

Keywords: Raspberry Pi, MATLAB, GPIO pins, Python programming, Arduino, GPU, Dual antennas

1. Introduction

A self-driving car is proficient of perceiving its environment and forming decisions independently, deprived of human intervention. These vehicles utilize a combination of sensors to detect roadways and traffic signals within their surroundings [1]. Autonomous cars offer several advantages over traditional vehicles, including less fuel consumption, improved safety, better mobility, and greater customer satisfaction. One of the most significant benefits is the potential for a substantial decrease in traffic accidents, as over 90% of accidents result from human errors such as distraction, impaired driving, or poor judgment [2]. By enabling vehicles to communicate and make decisions autonomously, the number of accidents is expected to decline considerably.

Driverless cars are also known as autonomous vehicles (AVs) or robo-cars, self-driving cars are designed to sense their environment and operate securely with minimal or no human input [3]. These vehicles rely on sophisticated control systems that analyse sensory data to determine safe navigation routes while detecting obstacles and interpreting relevant traffic signs. Applications for this technology include privately owned autonomous cars, shared rob taxi services, connected vehicle convoys, and long-haul trucking. Multiple efforts to improve fully autonomous commercial vehicles are underway. Figure 1 shows a brief flow charat of development of autonomous transport system.

Notably, Waymo launched the first rob taxi service open to the public in Phoenix, Arizona, in 2020. Tesla announced plans to offer a subscription-based "full self-driving" feature to private owners in 2021. Nuro received approval for autonomous commercial delivery operations in California the same year [5]. Research on automated

driving systems (ADS) dates back to at least the 1920s, with practical trials beginning in the 1950s. The first semi-automated vehicle was created in 1977 by Japan's Tsukuba Mechanical Engineering Laboratory. This vehicle relied on specially marked roads, which were detected by two on board cameras and processed with an analog computer. It was capable of speeds up to 19 mph and operated with the assistance of an elevated rail. Significant progress occurred in the 1980s with projects like Carnegie Mellon University's Navlab and ALV, sponsored by the U.S. Defense Advanced Research Projects Agency (DARPA) starting in 1984, and the EUREKA Prometheus Project initiated by Mercedes-Benz and Bundeswehr University Munich in 1987. By 1985, the ALV demonstrated autonomous driving at speeds of 31 km/h (19 mph) on two-lane roads, added obstacle avoidance capabilities in 1986, and achieved off-road operation during both day and night by 1987 [4]. A milestone was reached in 1995 when Carnegie Mellon's NavLab 5 completed the first coast-to-coast autonomous drive across the United States, covering 2,849 miles (4,585 km) from Pittsburgh to San Diego, with 98.2% of the journey (2,797 miles, 4,501 km) driven autonomously, maintaining an average speed of 63.8 mph (102.7 km/h). From the 1960s until the second DARPA Grand Challenge in 2005, U.S. automated vehicle research was primarily supported by DARPA, the Army, and the Navy, focusing on gradual improvements in speed, navigation in complex environments, control systems, and sensors. In 1991, the U.S. government invested $650 million in the National Automated Highway System, which combined vehicle automation with infrastructure and vehicle networking. The initiative culminated in a successful demonstration by 1997 but lacked further funding and a clear path for widespread implementation [6]. CMU's Navlab's 1995 cross-country autonomous drive remained unmatched for nearly twenty years until 2015, when Delphi surpassed the record with an Audi equipped with Delphi technology, completing 5,472 kilometres (3,400 miles) across 15 states, with self-driving mode active 99% of the time. In 2015, Nevada, Florida, California, Virginia, Michigan, and Washington, DC legalized the testing of autonomous vehicles on public roads [7].

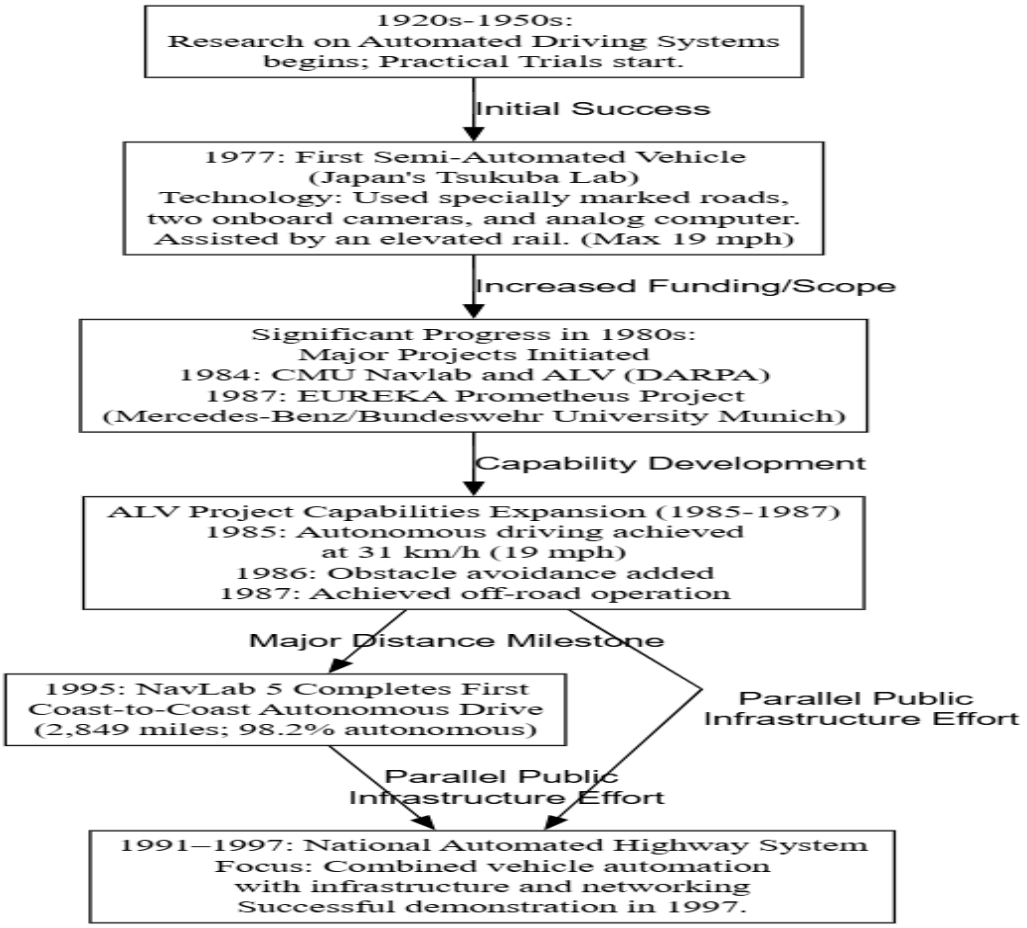

Figure 1 Representation of technological development of autonomous cars

This paper proposes and implements a low-cost prototype of a self-driving car. Equipped with an on board camera, the system utilizes video feed analysis by a processing computer to detect traffic signals such as right turn, left turn, and stop, enabling the vehicle to make accurate driving decisions [8].

The main objectives here, is to develop a low-cost, reproducible, real-time self-driving car prototype; However, the scope is limited to Educational lab-scale demo; Limitations: Only three signal types detected, no advanced AI or sensor array, limited obstacle handling, no field test or large-scale validation [9].

2. Raspberry Pi architecture

The Raspberry Pi is a compact, small single-board computer as shown in Figure 2. Currently, 5 models are existing: Model B+, Model A+, Model B, Model A, and the Compute Module (which is accessible only within the Compute Module development kit). While all these models share the same System on Chip (SoC), the BCM2835, which integrates both the CPU and GPU, they differ in other hardware specifications [10].

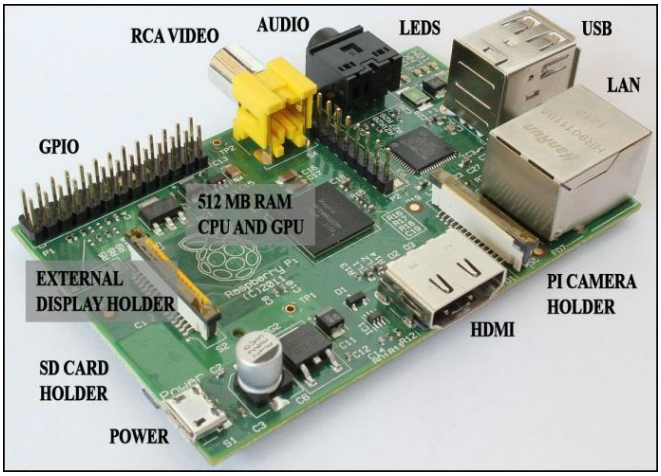

Figure 2: Raspberry Pi Board

Among the OS available for Raspberry Pi, including Arch, Risc OS, Plan 9, and Raspbian stands out as the most user-friendly and visually appealing. It offers the widest range of pre-installed software and is optimized specifically for Raspberry Pi hardware. Raspbian is a free, Debian-based Linux operating system and can be downloaded at no cost from the official Raspberry Pi website. The current project uses a minimal Raspberry Pi setup, Wi-Fi-based camera feed, simple algorithms (correlation, Hough Transform), and is focused on cost/pedagogy, unlike many prior projects with more expensive sensors or complex hardware. Cost reduction comes from using low-cost camera and Raspberry Pi, open-source OS/software, and excluding LIDAR [11]. Precise prices aren't given but a typical setup is below INR 8,000-10,000; academic models often exceed INR 15,000-40,000. Further, for Wi-Fi streaming, low-cost DIY build, and MATLAB-based processing with Circle Hough Transform—no LIDAR or heavy processing—distinguish this as a scalable education/research solution.

2.1 Motor Driver L298

The L298 Motor Driver Module is a robust, high-power driver considered for controlling both stepper and DC motors. It incorporates the L298 motor driver IC alongside a 78M05 5V voltage regulator. This module can accomplish up to 4 DC motors or 2 DC motors with individual speed and directional control, making it suitable for various robotics and automation applications. It operates with a driver voltage range from 5 volts to 35volts and can deliver up to 2A per channel, featuring dual H-bridge circuits for efficient motor control. The module supports PWM signals for speed regulation and includes on board features such as heatsinking and power-on indicators for enhanced reliability [12]. A DC motor is a electrical motor that converts direct current electrical energy into mechanical motion. It operates primarily through the magnetic forces generated by electric currents. Most DC motors include internal components, either electronic or electromechanical, which periodically reverse the current direction within the motor to sustain continuous rotation. This project utilizes two 5V DC motors for

its operation. It also houses on-board camera was used to feed live video of the pathway to the analyser pc via Wi-Fi.

2.2 Hardware Components

The chassis of the vehicle has four wheels, each powered by a separate motor. The motor driver IC, L293D, can control two motors simultaneously as illustrated in Figure 3. The wheels on the left side (front and rear) rotate in synchronization, as do the wheels on the right side. Consequently, each pair of motors on a side receives the same digital input from the L293D at any given time. This configuration enables the car to move forward or backward when the wheels on both sides rotate in the similar direction at identical speeds. The vehicle turns when the wheels on the left side rotate in the opposite direction to those on the right side. The chassis features two shelves above the wheels, separated by roughly 2 inches. The L293D IC is mounted on the lower shelf using two 0.5-inch screws, with motor wires always connected to the IC. Jumper wires from the L293D link it to the Raspberry Pi for control. The remaining space on the lower shelf houses eight AA batteries that supply power to the motors.

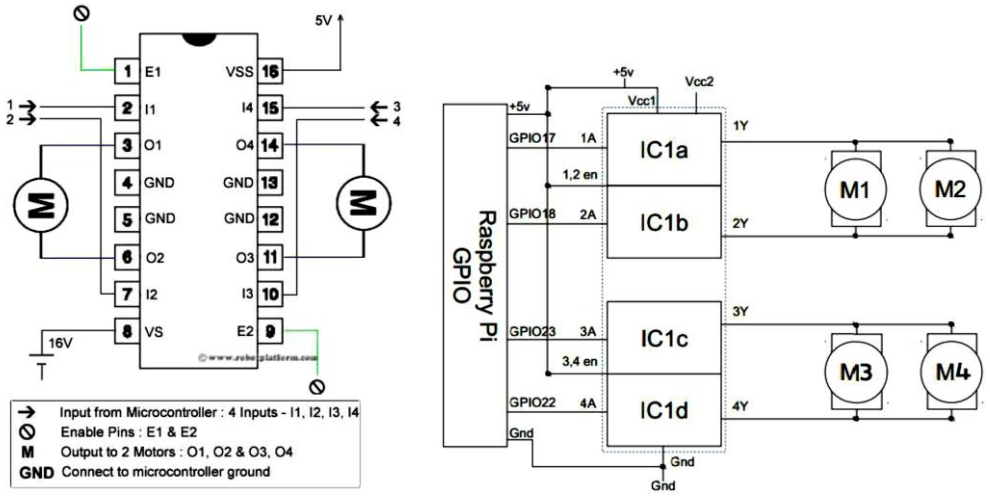

Figure 3: Raspberry Pi GPIO

The Raspberry Pi case is secured on the top shelf, along with an L-shaped aluminium bracket. The Raspberry Pi is installed inside the case, while the aluminium bracket provides support for the camera mounted on a servo motor and the ultrasonic sensor. A Wi-Fi dongle is connected to the Raspberry Pi's USB port to enable wireless connectivity. The full wiring diagram showing the connection between the Raspberry Pi and the L293D motor controller IC is illustrated in Figure 3. Since the Raspberry Pi requires a unique IP address, it must connect to a Wi-Fi router or hotspot. To ensure the Raspberry Pi recognizes the router automatically upon start-up, specific configuration changes are made to the network settings.

3. System Architecture and Process Implementation

System Architecture is in **Error! Reference source not found.**. The process implementation is depicted in Fig 4, and the For video feed acquisition, a mobile phone camera paired with the DroidCam app is utilized [13]. This app streams the camera feed data to a specified TCP server. An ESP8266 Wi-Fi module (NodeMCU v0.9) is employed to establish a TCP server, serving as the communication interface between the image processing system and the prototype car. To detect traffic signs from real-time video, the continuous feed is segmented into frames at 30-millisecond intervals. Given that only three traffic signs need recognition (stop, turn left, and turn right), a straightforward correlation-based method is applied. The process is broken down into several steps: Training Image Acquisition: Approximately 10 images of each traffic sign are captured. The sign region is cropped from these images and saved as MATLAB data files. Each image is standardized to a resolution of 600x600 pixels for consistent processing. It is broken down into following steps.

i. Training Image Acquisition: Approximately 10 images of each traffic sign are captured. The sign region is cropped from these images and saved as MATLAB data files. Each image is standardized to a resolution of 600x600 pixels for consistent processing.
ii. Real-Time Video Segmentation: Frames are extracted from the live video feed at intervals of 30 milliseconds. Each extracted frame is then individually processed to identify traffic signs.
iii. Circular Object Detection in Images: The Hough Transform technique is applied to detect circular shapes within the extracted frames, aiding in the recognition of specific traffic signs.
iv. Correlation: Then we correlate the extracted circle with each sign. Thresholds were selected to notice each type of sign through trial and error.

The Circle Hough Transform (CHT) is a fundamental feature extraction method used in digital image processing for identifying circular shapes in images that may be imperfect or noisy. It is a specialized form within the broader Hough Transform family. The algorithm works by voting in the Hough parameter space, where potential circle candidates accumulate votes in a matrix known as the accumulator. Local maxima in this accumulator correspond to detected circles. This approach is particularly effective at recognizing circular objects despite irregularities in the image. The CHT technique is implemented using MATLAB functions in this project to detect circles within video frames.

In many self-driving car prototypes built using Raspberry Pi and Arduino, the Raspberry Pi serves as the primary processing unit. It processes data from sensors such as cameras and ultrasonic sensors to interpret the surrounding environment and make navigation decisions. These decisions are then sent as control instructions to the Arduino, which acts as the motor controller. The Arduino receives commands from the Raspberry Pi via GPIO pins or serial communication interfaces like UART. Based on these received instructions, the Arduino controls the motor driver (such as L293D or L298N) to regulate the speed and direction of the car's DC motors. This setup allows synchronized movement of the wheels for forward, backward, turning left, or turning right actions. The communication flow typically involves: Raspberry Pi capturing video and sensor data, Processing the inputs using image processing algorithms (OpenCV) and sensor data fusion, Sending high-level commands (e.g., move forward, turn right) to Arduino, Arduino translating commands into motor control signals to drive the wheels. For wireless operation, the Raspberry Pi might connect to a mobile device or local network through Wi-Fi, enabling remote control and monitoring. Applications such as VNC Viewer or SSH clients on a mobile device provide an interface to control or monitor the prototype car remotely. Sources include implementations with Python programming on Raspberry Pi, interfacing Arduino with motor drivers, and wireless communication modules (Wi-Fi dongle or ESP8266) for remote connectivity and command transmission.

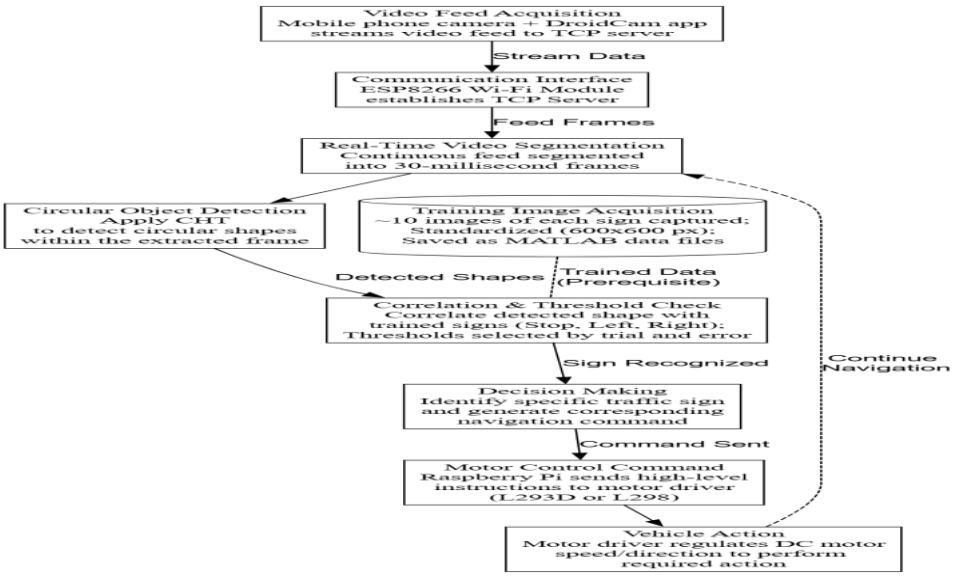

Figure 4: Process implementation

The assembled prototype shown in Fig 5 represents a compact autonomous vehicle platform designed for experimentation and development in self-driving car technology. The chassis integrates several critical components, including a robust sensor array, control boards, and communication modules. Key hardware elements featured include an on-board processing unit equipped with a high-performance embedded GPU, enabling real-time data processing and advanced machine learning tasks essential for autonomous decision-making. A high-resolution camera and ultrasonic sensors are mounted on servo motors to capture environmental data and detect obstacles. The prototype uses wireless communication, facilitated by dual antennas, which allows for seamless data exchange between the vehicle and external controllers or monitoring systems. The vehicle's drivetrain consists of four independently controlled wheels powered by dedicated motors, providing precise movement control such as synchronized forward/backward motion and differential turning capability. The open design permits easy access for customization, sensor upgrades, and integration of additional components. Overall, this prototype serves as a versatile testbed for evaluating algorithms related to image processing, object detection, sensor fusion, and autonomous navigation, demonstrating practical application of embedded systems and robotics in the field of autonomous vehicles. Overall the system is simple and yet robust right-turn signal recognition, quick prototyping, and reproducibility for tutorials and workshops. Reliable real-time decision process for the tested scenarios and the car consistently detects turning signs using segmented video frames (30ms), with robust sign region correlation—an efficient path for low-cost sign detection. Accurate turning detection, real-time streaming, proof-of-concept for traffic sign-based navigation is achieved.

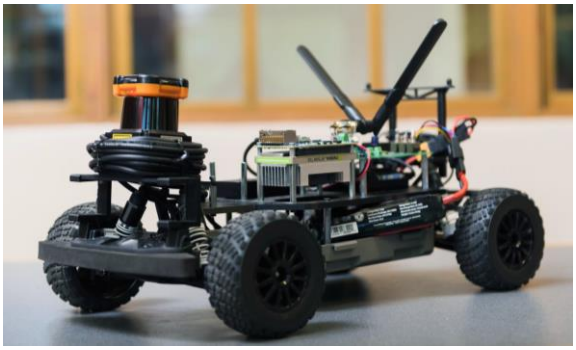

Figure 5: Assembled prototype

4. Conclusion

The primary contribution of this paper is the successful design and implementation of a low-cost prototype of a self-driving car, utilizing the Raspberry Pi as the central processing unit. This system integrates real-time video processing, sensor input fusion, and motor control to enable reliable navigation, traffic sign recognition, and obstacle avoidance, thereby having the potential of accessible and scalable technologies towards better autonomous driving research. Having an on board camera, the vehicle is capable of precise and dependable decision-making, specifically detecting road signals and executing accurate turning actions. For traffic signal recognition, the system analyses a continuous video feed, segmented into 30-millisecond frames, to identify three key signs: stop, turn left, and turn right. This detection uses a straightforward correlation-based method combined with the Circle Hough Transform, implemented via MATLAB functions, to reliably detect circular shapes within the frames despite image irregularities. Architecturally, the system connects the car's chassis to an analysis computing system via Wi-Fi, utilizing an ESP8266 module to establish a TCP server, with video feed acquisition managed through a mobile phone camera paired with the DroidCam app Overall, by combining computer vision techniques with embedded hardware, this prototype demonstrates the potential of accessible and scalable technologies in advancing autonomous driving research. More work can be, in future focus on optimizing algorithms and enhancing sensor precision to manage more complex driving environments

References

1. G. S. Pannu, M. D. Ansari, and P. Gupta, "Design and Implementation of Autonomous Car using Raspberry Pi," International Journal of Computer Applications, vol. 113, no. 9, pp. 22–29, Mar. 2015, doi: 10.5120/19854-1789.
2. R. P. Kharapkar, A. S. Khandare, and Z. W. Siddiqui, "IoT based Self Driving Car," International Research Journal of Engineering and Technology, vol. 7, no. 3, pp. 5177–5181, Mar. 2020.
3. T.-H. S. Li, M.-H. Lee, C.-W. Lin, G.-H. Liou, and W.-C. Chen, "Design of Autonomous and Manual Driving System for 4WIS4WID Vehicle," IEEE Access, vol. 4, pp. 2256–2271, Mar. 2016.
4. R. Shirolkar, A. Dhongade, R. Datar, and G. Behere, "Self-Driving Autonomous Car using Raspberry Pi," International Journal of Engineering Research & Technology, vol. 8, no. 5, pp. 100–110, May 2019.
5. M. Alfian Ma'arif, A. A. Nuryono, and I. Iswanto, "Vision-Based Line Following Mobile Robot using Raspberry Pi," International Journal of Intelligent Engineering and Systems, vol. 10, no. 1, pp. 339–348, Feb. 2017.
6. S. M. Ali, "Real Time Object Detection and Tracking for Autonomous Car Using Raspberry Pi and OpenCV," International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, vol. 4, no. 1, pp. 46–53, Jan. 2016.
7. P. S. Rao, B. V. Prasad, and K. Dhanunjay, "Developing a Real-Time Autonomous Car with Raspberry Pi," International Journal of Engineering Trends and Technology, vol. 68, no. 6, pp. 39–44, Apr. 2020.
8. Y. Wang, M. Yan, and L. Wang, "Autonomous Vehicle with Raspberry Pi and Deep Learning," IEEE International Conference on Intelligent Transportation Systems, pp. 1919–1924, Oct. 2019.
9. A. K. Singh and R. Kumar, "Design and Implementation of Self-Driving Car using Raspberry Pi," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol. 7, no. 9, pp. 3470–3477, Sep. 2018.
10. S. M. K. Bharti, S. Mandloi, and P. Jain, "Lane Detection and Obstacle Avoidance for Low Cost Autonomous Vehicle Prototype," International Journal of Computer Applications, vol. 180, no. 44, pp. 5–10, Mar. 2018.
11. K. A. Hassan and M. S. Hossain, "Vision-Based Autonomous Vehicle Using Raspberry Pi," International Journal of Computer Science and Information Security, vol. 14, no. 5, pp. 89–96, May 2016.
12. M. J. Jadhav and S. V. Patil, "Object Detection for Autonomous Car Using Raspberry Pi," International Journal of Engineering and Advanced Technology, vol. 8, no. 6, pp. 2523–2527, Aug. 2019.
13. P. Singh and R. Kumar, "Implementation of Autonomous Car on Raspberry Pi using Image Processing," International Journal of Innovative Technology and Exploring Engineering, vol. 8, no. 6, pp. 1235–1240, Apr. 2019.
14. H. C. Nguyen, T. N. Tran, and M. S. Islam, "Autonomous Vehicle Control Using Raspberry Pi and Machine Learning," IEEE Access, vol. 7, pp. 173658–173669, Dec. 2019.
15. N. Patel and J. Patel, "Self-Driving Car Prototype using Raspberry Pi and Computer Vision," International Journal of Computer Applications, vol. 176, no. 44, pp. 1–6, Oct. 2017.
16. S. H. Lee and J. H. Kim, "Sensor Fusion-Based Autonomous Driving System on Raspberry Pi Platform," IEEE Sensors Journal, vol. 20, no. 18, pp. 10924–10932, Sep. 2020.