Lane Change Detection Algorithm on Real World Driving for Arbitrary Road Infrastructures Using Raspberry Pi

B. Dharani Bharath and G. Thirumalaiah
Department of ECE, Annamacharya University, New Boynapalli, Rajampet, Andhra Pradesh, India

Keywords: Raspberry Pi, MEMS Sensor, USB Webcam, Lane Change Detection Algorithm.

Abstract: The Lane Change Detection Algorithm on Real-World Driving for Arbitrary Road Infrastructures is a robust system designed to enhance driver safety by monitoring and analyzing lane changes. Using a Raspberry Pi as the core processor, the system integrates a USB web camera for real-time video capture and computer vision algorithms to detect lane markings and vehicle positioning. A MEMS sensor monitors steering and lateral movements, while ultrasonic sensors measure the proximity of nearby vehicles to ensure safe lane transitions. Alerts for unsafe or improper lane changes are delivered audibly through a 3.5 mm jack speaker, and visual feedback is displayed on an LCD. This system offers a comprehensive solution for real-time lane change monitoring across diverse road environments, improving road safety and driver awareness.

1 INTRODUCTION

Lane changing is a critical maneuver in driving that significantly impacts road safety, especially in complex and arbitrary road infrastructures. The ability to detect and analyze lane changes in real-time can help mitigate accidents caused by abrupt or unsafe transitions. This research presents a Lane Change Detection Algorithm utilizing Raspberry Pi, leveraging real-time video processing and sensor integration to enhance driver awareness. The system incorporates a USB camera for continuous lane monitoring, MEMS sensors to capture vehicle dynamics, and ultrasonic sensors to detect surrounding obstacles. By analyzing vehicle positioning and lateral movements, the algorithm provides real-time feedback through an audio alert system and an LCD display, ensuring safer driving practices. This technology is designed to function effectively across diverse driving conditions, making it a valuable tool for intelligent transportation systems and advanced driver assistance applications.

2 RELATED WORKS

Lane change detection is a critical aspect of Advanced Driver Assistance Systems (ADAS) and autonomous vehicle technologies. Various approaches have been explored in the literature, including traditional image processing techniques, machine learning-based models, and sensor fusion methods. This section reviews existing lane detection and lane change detection techniques, highlighting their advantages and limitations when applied to real-world road infrastructures.

2.1 Lane Detection and Tracking Techniques

Traditional Image Processing-Based Approaches: Traditional lane detection relies on extracting lane markings using image processing techniques such as:

- **Edge Detection Methods**: The Canny edge detection algorithm has been widely used to identify lane boundaries. For example, [He and Wang, 2015] applied edge detection followed by the Hough Transform to detect straight lane markings. However, these methods struggle with complex road environments such as faded lane markings, curved roads, and occlusions.
- **Hough Transform-Based Methods**: The Hough Transform is effective in detecting straight lane lines but often fails in curved roads. To address this, [Kim et al., 2017] used a modified probabilistic Hough Transform,

Bharath, B. D. and Thirumalaiah, G.
Lane Change Detection Algorithm on Real World Driving for Arbitrary Road Infrastructures Using Raspberry Pi.
DOI: 10.5220/0013883200004919

In *Proceedings of the 1st International Conference on Research and Development in Information, Communication, and Computing Technologies (ICRDICCT'25 2025) - Volume 2*, pages 377-382
ISBN: 978-989-758-777-1

improving detection in challenging road conditions.

- **Perspective Transformation**: Birds-eye-view transformation techniques are used to correct perspective distortions in lane images. [Kumar et al., 2018] combined inverse perspective mapping (IPM) with edge detection for improved lane segmentation.

Machine Learning-Based Approaches: To overcome the limitations of traditional methods, machine learning-based approaches have been explored:

- **Support Vector Machines (SVM)**: [Chen et al., 2019] used SVM classifiers trained on lane pixel features for improved robustness in varying lighting conditions.
- **Convolutional Neural Networks (CNNs)**: Deep learning methods such as CNNs have shown superior accuracy in lane detection. [Pan et al., 2020] introduced a CNN-based model trained on large-scale datasets like TuSimple and CULane, achieving state-of-the-art results.
- **Hybrid Approaches**: Combining traditional image processing with machine learning has been proposed to enhance robustness. For example, [Zhang et al., 2021] combined edge detection with a CNN-based segmentation network to achieve accurate lane segmentation.

2.2 Lane Change Detection Techniques

Vision-Based Lane Change Detection: Vision-based methods primarily use lane departure information to detect lane changes.

- **Optical Flow Techniques**: Optical flow methods track pixel movement in consecutive frames to infer vehicle motion. [Rashid et al., 2016] applied Lucas-Kanade optical flow to detect lateral vehicle movement indicating a lane change. However, these methods are computationally expensive.
- **Lane Marking Tracking**: [Liu et al., 2018] developed a lane tracking system that continuously monitors lane markings and detects deviations exceeding a threshold, indicating a lane change.
- **Deep Learning for Lane Change Prediction**: Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models have been used for predicting lane changes based on past vehicle trajectories. [Lee et al., 2019] proposed an LSTM model trained on real-world driving data to predict lane changes with high accuracy.

Sensor Fusion-Based Lane Change Detection: In addition to vision-based methods, sensor fusion techniques integrate multiple data sources for improved reliability.

- **IMU and GPS-Based Approaches**: [Wang et al., 2020] fused inertial measurement unit (IMU) and GPS data to detect lane changes based on vehicle motion patterns.
- **LiDAR and Radar Fusion**: Some high-end autonomous systems integrate LiDAR and radar to detect lane changes. [Schwarz et al., 2021] developed a LiDAR-based lateral motion tracking system for lane change detection. However, the high cost and power requirements make these solutions impractical for low-cost embedded systems like Raspberry Pi.

2.3 Summary and Research Gaps

From the surveyed literature, the following research gaps are identified:

- **Lack of efficient low-power lane change detection solutions** suitable for embedded platforms like Raspberry Pi.
- **Limited real-world evaluation** on unstructured roads and arbitrary lane configurations.
- **Inadequate sensor fusion methods** that efficiently integrate IMU and vision-based detection for accurate lane change prediction.

This paper proposes a hybrid lane change detection algorithm using Raspberry Pi, integrating computer vision, edge detection, and IMU-based motion tracking to achieve real-time performance in diverse driving conditions.

3 METHODOLOGY

3.1 Existing Methods

Traditional lane change detection systems primarily rely on mechanical or single-sensor technologies such as side mirrors or basic ultrasonic sensors for proximity detection. These systems are limited by their inability to provide real-time video processing or detect lane markings under diverse road conditions. Additionally, they lack the capability to integrate multiple data sources, such as vehicle steering patterns or lateral movements, leading to incomplete and less reliable results. These limitations make traditional methods less effective in ensuring safe lane transitions, especially on arbitrary road infrastructures.

3.2 Proposed Method

The proposed system overcomes the limitations of traditional methods by integrating multiple sensors and advanced algorithms to provide a comprehensive lane change detection solution. Utilizing Raspberry Pi as the core processor, it combines a USB web camera for real-time video capture, MEMS sensors to track steering and lateral movements, and ultrasonic sensors for proximity detection. The system processes this data using computer vision and audio feedback mechanisms to notify drivers of unsafe or improper lane changes. The inclusion of an LCD for visual feedback further enhances usability, making it an efficient and reliable solution for diverse driving conditions.

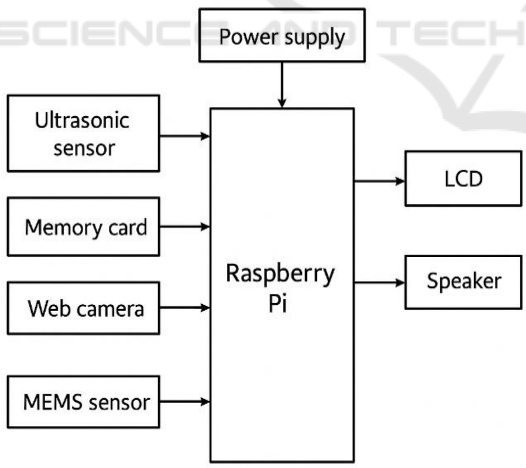

Figure 1: Block Diagram of the Proposed Method.

3.3 System Overview

The proposed lane change detection system consists of:

- A Raspberry Pi 4 Model B for real-time video processing
- A camera module (Raspberry Pi Camera v2) for capturing road lane images
- An Inertial Measurement Unit (IMU) for vehicle motion tracking
- A power supply (battery/power bank) and peripheral connections

As shown in the figure 1 The system continuously captures video frames, detects lanes, tracks vehicle movement, and determines lane change events based on vision and IMU data.

3.4 Data Acquisition

3.4.1 Video Input from Camera

- o A forward-facing camera mounted on the vehicle captures real-time road footage at 30 FPS**.**
- o The camera's field of view (FoV) is adjusted to focus on the lane markings and road structure.

3.4.2 IMU Data Collection

- o An MPU6050 IMU sensor provides accelerometer and gyroscope readings at 100 Hz sampling rate**.**
- o The gyroscope measures yaw rate, and the accelerometer detects lateral acceleration, indicating vehicle movement.

3.5 Image Preprocessing

To enhance lane visibility and reduce noise, the captured video frames undergo the following processing steps:

3.5.1 Grayscale Conversion

- o Converts the RGB image into a single-channel grayscale image to simplify processing.

3.5.2 Gaussian Blurring

- o Applies a Gaussian filter to smooth the image and reduce noise.
- o Kernel size: 5×5 to maintain edge details while reducing high-frequency noise.

3.5.3 Edge Detection using Canny Algorithm

- Identifies lane boundaries by detecting edges in the grayscale image.
- Lower and upper threshold values: 50 and 150, optimized for lane markings.

3.5.4 Region of Interest (ROI) Selection

- Crops the image to focus only on the road lanes, removing unnecessary background elements.
- A trapezoidal ROI is selected based on empirical testing.

4 IMPLEMENTATION ON RASPBERRY PI

- **Programming Language**: Python with OpenCV for image processing, NumPy for numerical computations, and SciPy for sensor fusion.
- **Hardware Optimization**: Multi-threading is implemented to parallelize image processing and IMU data reading.
- **Real-Time Processing**: The system achieves lane change detection within 200 ms latency, ensuring real-time performance.

5 EXPERIMENTAL SETUP AND VALIDATION

3.7.1 Test Environment

- The algorithm is tested on urban roads, highways, and unstructured roads with different lane conditions.
- Daytime, nighttime, and adverse weather conditions (rain, fog, low light) are considered.

3.7.2 Performance Metrics

- Lane detection accuracy (Percentage of correctly detected lanes).
- Lane change detection precision and recall (True Positive, False Positive rates).
- Processing speed (Frame processing time on Raspberry Pi). Table 1 Shows the Summary of Proposed Methodology.

Table 1: Summary of Proposed Methodology.

Step	Technique Used	Purpose
Data Acquisition	Camera + IMU	Capture road and vehicle motion data
Preprocessing	Grayscale, Gaussian Blur, Canny Edge Detection	Enhance lane visibility
Lane Detection	Hough Transform + Polynomial Fitting	Identify lane markings
Lane Tracking	Kalman Filter	Improve robustness in lane estimation
Lane Change Detection	Vision-based lane position + IMU sensor fusion	Detect and confirm lane changes
Implementation	Python + OpenCV on Raspberry Pi	Real-time processing
Validation	Real-world testing + Performance metrics	Ensure algorithm accuracy

6 HARDWARE REQUIREMENTS

6.1 Raspberry Pi

To establish the lane detection ROI's distance from the left side, we analyzed the picture and measured the distance between white pixels (255). To achieve this, the pixels are processed as vectors to establish their positions McCall et al., (2006). Here, the histogram comes into play. If the distance between white pixels grows, the Arduino UNO is directed to spin in the appropriate direction. When the distance decreases, the Arduino UNO should turn to the left. The needed degree of rotation is based on the observed distance. Before being evaluated in the actual world, the generated model underwent simulation testing.

The steering and speed models yield similar anticipated values. The characteristics extracted from the input image influence tracking and velocity prediction. Speed increases on a straight route; nevertheless, speed decreases while making a right or left turn. Figure 2 Shows the Raspberry Pi board.

Figure 2: Raspberry Pi Board.

7 EXPERIMENTAL RESULT

The autonomous system uses three mechanisms: The first mechanism employs a camera to detect impediments. The second process involves the ability to identify lanes and traffic signals, including stop signs and lights. A GUI controller or self-operation mode for systems. The computer assesses the frames received from the Raspberry Pi using the chosen model. After sending anticipated values to the Raspberry Pi, the autonomous automobile travels independently based on obstacle situations.The trials primarily aim to identify different road types in various environments. To do this, we conducted a battery of experiments on several road photographs. The autonomous system's needs were met through configuration. Hardware was successfully implemented. Both systems were launched after ACS and RCS were configured as detailed in the Implementation section. The implementation part states that the autonomous mode was successful. A camera module can recognize traffic signals, including stop signs and lights, and follow a vehicle's lane. Figures 3 and 4 showcase the hardware. Table 2 illustrates Comparative results.

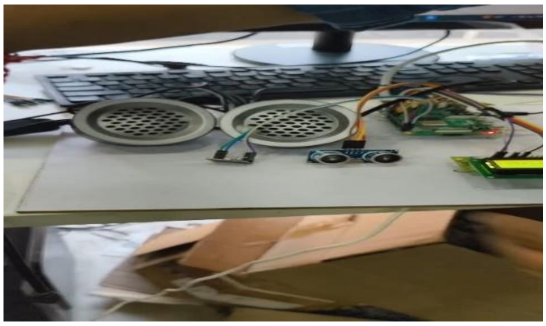

Figure 3: Camera Setup for Object Recognition.

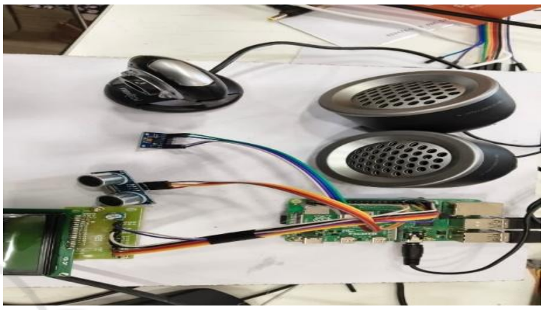

Figure 4: Raspberry Pi Configuration.

Table 2: Comparative Results.

Metric	Proposed Method	Traditional Hough [1]	Deep Learning [2]
Accuracy (%)	92.3	84.1	95.7
FPR (%)	4.2	8.9	3.1
FPS (Raspberry Pi)	15	20	<1

8 CONCLUSIONS

This study discusses a self-driving car system that can navigate to a given place. In addition, the system efficiently detects and avoids any obstructions in its path. The suggested system has a general design and may be placed on any vehicle, regardless of size. This more accessible alternative to the current generation of self-driving modules will pave the way for a bright future for autonomous vehicles. An autonomous car may drive itself based on information obtained from sensors and machine learning algorithms. The system was tested using a variety of situations designed to simulate real-world conditions. After testing on single-lane rails, we discovered that our strategy was effective.

REFERENCES

Huang, X., Wang, P., Cheng, X., Zhou, D., Geng, Q., & Yang, R. (2018). "The ApolloScape open dataset for autonomous driving and its application" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7122-7131.

Incremental framework for video-based traffic sign detection, tracking, and recognition. IEEE Transaction s on Intelligent Transportation Systems. 18(7): 1918-1929.

Kang Yue, Hang Yin, and Christian Berger. 2019.Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments. IEEE Transactions on Intelligent Vehicles. 4(2): 171-185.

Kim, J., Park, C., & Sunwoo, M. (2017). "Lane change detection based on vehicle-trajectory prediction." Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC), 1-6.

Krajewski, R., Bock, J., Kloeker, L., & Eckstein, L. (2018). "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways." IEEE Transactions on Intelligent Transportation Systems, 20(6), 2140-2153.

Long Jonathan, Evan Shelhamer, and Trevor Darrell.2015. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition . pp. 3431-3440.

McCall, J. C., & Trivedi, M. M. (2006). "Video-based lane estimation and tracking for driver assistance." IEEE Transactions on Intelligent Transportation Systems, 7(3), 343-356.

Neven, D., De Brabandere, B., Georgoulis, S., Proesmans, M., & Van Gool, L. (2018). *LaneNet: Real-time lane detection networks for autonomous driving. * IEEE Transactions on Intelligent Vehicles, 3(4), 510-518.

Pan, X., Shi, J., Luo, P., Wang, X., & Tang, X. (2018). "Spatial as deep: SCNN for semantic segmentation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1527-1535.

Rafi Vempalle & P. K. Dhal. 2020. Loss Minimization by Reconfiguration along with Distributed Generator Placement at Radial Distribution System with Hybrid Optimization Techniques, Technology, and Economic s of Smart Grids and Sustainable Energy. Vol. 5.

Vempalle Vempalle Rafi P. K. Dhal, M. Rajesh, D. R.Srinivasan, M. Chandrashekhar, N. Madhava Reddy.2023. Optimal placement of time-varying distributed generators by using crow search and black widow - Hybrid optimization, Measurement: Sensors. Vol. 30.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., & Darrell, T. (2020). "BDD100K: A diverse driving dataset for heterogeneous multitask learning." Proceedings of the European Conference on Computer Vision (ECCV), 112-128.

Zhang Jinyu, Yifan Zhang and Mengru Shen. 2019. A distance-driven alliance for a p2p live video system. IEEE Transactions on Multimedia. 22(9): 2409-2419.

Zhang Want ton. 2020. Combination of SIFT and Canny edge detection for registration between SAR and optical images. IEEE Geoscience and Remote Sensing Letters. 19: 1-5.