Al-Azhar Engineering 16th International Conference

Vol. 19, No. 72, July 2024, 167 - 182

A COMPREHENSIVE APPROACH TO AUTONOMOUS VEHICLE NAVIGATION

Manal Mustafa Ali[*], Reham Abobeah, Momtaz Saad Elkholy

Computer and Systems Engineering, Faculty of Engineering, Al-Azhar University, Nasr City, 11884, Cairo, Egypt

***Correspondence:** manal.mustafa@ azhar.edu.eg

Citation:

M.M. Ali, R. Abobeah and M.S. Elkholy, "A Comprehensive Approach to Autonomous Vehicle Navigation", Journal of Al-Azhar University Engineering Sector, vol. 19, pp. 167 - 182, 2024.

Received: 07 March 2024

Revised: 14 May 2024

Accepted: 27 May 2024

DoI:10.21608/auej.2024.275435.1634

ABSTRACT

Autonomous vehicles are revolutionizing transportation, and the accuracy of road lane detection is a pivotal aspect of this innovation. This paper presents an in-depth exploration of a sophisticated lane detection system, geometric modeling to estimate the geometric structure of lane boundaries based on images captured by an onboard vehicle camera, and the deployment of object detection techniques. The lane detection system is meticulously designed, employing a series of computer vision techniques to identify and track lanes in various driving conditions. The curve fitting component utilizes a second-order polynomial, providing a mathematical model that accurately represents the curvature and intricate dynamics of the detected lanes. This mathematical representation provides a more nuanced understanding of the road geometry, aiding in the prediction of vehicle trajectory. The object detection facet of the research focuses on the recognition and classification of objects within the driving environment, contributing significantly to the overall situational awareness of autonomous driving systems. The YOLO (You Only Look Once) algorithm is commonly used for this purpose as it can process frames at an impressive speed while maintaining high accuracy, making it suitable for real-time applications. The efficacy of the suggested techniques was confirmed by conducting experiments on two distinct datasets. The proposed method achieved an accuracy of 98.64% on the Tusimple and 96.92% on the KITTI dataset, demonstrating its robustness and reliability under varying conditions.

KEYWORDS: Autonomous driving, Lane detection, Edge detection, Curve fitting, Object detection, YOLO.

تصميم نظام شامل للتنقل بالمركبات الذاتية القيادة

منال مصطفى على*، ريهام أبو بيه، ممتاز سعد الخولى

قسم النظم والحاسبات، كلية الهندسة، جامعة الأزهر، مدينة نصر، 11884، القاهرة، مصر

***البريد الالكتروني للباحث الرئيسي:** manal.mustafa@ azhar.edu.eg

الملخص

تُحدث المركبات ذاتية القيادة ثورة في مجال وانظمة النقل، ودقة كشف المسارات على الطرق تعد جانبًا حاسمًا في هذا الابتكار. يقدم هذا البحث استكشافًا معمقًا لنظام كشف المسارات المتطور، وتطبيق معادلة من الدرجة الثانية لتنبوء وحساب الانحناءات، وتنفيذ تقنيات كشف وتحديد الأجسام. يبدأ البحث بتصميم نظام الكشف عن المسارات بعناية فائقة، حيث يستخدم سلسلة من تقنيات الرؤية الحاسوبية لتحديد وتتبع المسارات في ظروف القيادة المختلفة. يستخدم لحساب انحناء الحارات متعددات الحدود معادلة من الدرجة الثانية، مما يوفر نموذجًا رياضيًا يمثل ديناميكية وانحناء المسارات المكتشفة بدقة عالية. يوفر هذا التمثيل الرياضي فهمًا أكثر تفصيلاً لهندسة الطريق، مما يساعد في التنبؤ بمسار السيارة. يركز جانب الكشف عن الأجسام من البحث على التعرف على الأجسام وتصنيفها داخل بيئة القيادة، مما يساهم بشكل كبير في الوعي الظرفي العام لأنظمة القيادة الذاتية. تُستخدم خوارزمية YOLO بشكل شائع لهذا الغرض حيث يمكنها معالجة الصور بسرعة مذهلة مع الحفاظ على الدقة العالية، مما يجعلها مناسبة لتطبيقات الوقت الفعلي.تم تأكيد فعالية التقنيات المقترحة من خلال إجراء تجارب على مجموعتي بيانات متميزتين . هذا وقد حقق النموذج المقترح دقة بلغت 98.64% على مجموعة البيانات Tusimple و% 96.92 على مجموعة البيانات KITTI، مما يدل على قوتها وموثوقيتها تحت الظروف المتغيرة.

الكلمات الافتتاحية: القيادة الذاتية، اكتشاف المسار، اكتشاف الحواف، تركيب المنحنى، اكتشاف الأشياء، YOLO

1. INTRODUCTION

The complexity of autonomous driving environments is a major obstacle, as they are subject to various changes [1]. These changes encompass a multitude of factors, each presenting a formidable challenge for advanced driver assistance systems (ADAS) such as lane detection, object detection, traffic sign recognition, and collision avoidance. The intricacy of autonomous driving environments is a critical bottleneck in this field [2] [3]. Lane line detection is the fundamental component of ADAS as lots of traffic rules are based on the lane line mark [4-6]. However, missing or obscured lane lines pose significant detection challenges [7]. These issues necessitate advanced algorithms for accurate, real-time, and robust lane detection. Curvature fitting plays an indispensable role in enhancing the performance of lane detection systems in autonomous vehicles [3]. It involves the use of mathematical models to accurately represent the curvature of detected lanes [8]. This is crucial as roads are rarely straight and often have varying degrees of curvature. [7] and [9] introduce a method for detecting both straight and curved lanes under various conditions. Their methods are adaptable and versatile, catering to a wide range of scenarios.

One main requirement for intelligent vehicles is that they need to be able to perceive and understand their surroundings in real time [1] [10]. YOLO's efficiency in real-time object detection, particularly vehicles, complements lane detection's ability to identify drivable areas. This combination provides a holistic view of the road environment, enhancing decision-making capabilities in various driving scenarios. [11-13] put forward a novel and efficient approach for detecting objects in self-driving vehicles. Their methods leverage the power of the YOLO algorithm, known for its speed and precision in real-time object detection. This research makes contributions to the field of autonomous driving systems in three key areas:

1. Robust lane detection: The development and implementation of a multi-stage lane detection algorithm that effectively identifies and tracks lanes under various driving conditions. This is achieved through a series of computer vision techniques, including grayscale conversion, bilateral filtering, OTSU thresholding, and Canny edge detection. The detected lanes are further refined using ROI extraction, bird's eye view transformation, and a sliding window approach.
2. Curve fitting for road geometry: The application of a second-order polynomial for curve fitting provides a mathematical model that accurately represents the curvature of detected lanes. This offers crucial insights into the geometry of the road, enhancing the vehicle's ability to navigate safely and efficiently.
3. Object detection for situational awareness: The deployment of object detection techniques contributes significantly to the overall situational awareness of the autonomous driving system. By identifying and classifying objects within the driving environment, the system can make informed decisions, improving safety and performance.

The robustness and reliability of the proposed methods were validated through experiments on two different datasets, achieving an accuracy of 98.64% on the Tusimple and 96.92% on the KITTI dataset respectively, demonstrating the potential for real-world applications. The paper is organized in the following manner: The second section is dedicated to the presentation of related work. The third section explains the proposed research methodology including lane detection, curvature fitting, and object detection. The underlying datasets are introduced in the fourth section. The fifth section is where we delve into the experimental part, which includes implementation and visualizations then a comparison with the previous work. Lastly, the sixth section offers a conclusive summary and indicates potential paths for upcoming research.

2. RELATED WORK

Numerous past researches have investigated the application of edge detectors, Hough transform, thresholding, and inverse perspective transformation in diverse image processing activities. For example, [8] introduces an advanced curve lane detection method for autonomous vehicles, utilizing inverse perspective transformation, OTSU threshold, and Hough transform. Their algorithm employs models of parabolic and circular equations within the Kalman filter to calculate the parameters of a curve lane. Despite its widespread use in detecting lane markings, the Hough transform method often struggles to accurately identify the correct lane markings [5] [15]. Therefore, Jung and Youn [7] in their paper, introduced a method for lane detection that utilizes spatiotemporal images. This approach proved effective in identifying lanes with significant curvature, changes in lane direction, roads at night, obstructions, and even lens flare, thanks to the consistent measurement of lane width along each scanline. Similarly, [9] presents a real-time lane detection method for both straight and curved lanes, adaptable to various conditions. It uses Hough transform optimization and the Kalman filter for accurate lane identification and tracking with detection of 96.3% accuracy for straight lanes, and 97.74% for curved lanes at 16.7 fps.

In their work, Zhang et al [2] propose a spatiotemporal network that utilizes a pair of Convolutional Gated Recurrent Units (ConvGRUs) for the task of lane detection in difficult scenarios. [16] use a lightweight CNN model as a feature extractor on KITTI and Caltech datasets containing small image patches of dimension 16 × 64 pixels. A non-overlapping sliding window approach is employed to achieve fast inference. After the predictions are made, they are clustered and fitted with a polynomial to model the lane boundaries. [17] preprocess the input image by dividing it into several slices along the vertical axis and extracting a set of features for each slice which are then fed into a deep network that combines a CNN and a fully connected network to classify each pixel as a lane or non-lane. The authors of [5] suggest a hierarchical deep Hough transform (DHT) that amalgamates all lane characteristics in an image into the Hough parameter space. They enhance the method of point selection and include a dynamic convolution module to efficiently distinguish between lanes in the source image. Philion [18] presents a unique fully convolutional model for lane detection that directly decodes lane structures and employs unsupervised style transfer to adapt to new conditions, ensuring robust performance with accuracy reaching 95.2%.

Detecting lanes and vehicles offers distinct yet complementary insights into the road scene. While lane detection pinpoints the drivable area of the road, vehicle detection identifies other road users. The integration of these two aspects provides a more holistic understanding of the road scene. Therefore, [11] addresses the development of lane and obstacle detection for self-driving cars. The proposed algorithm was tested using real-time videos and the Tusimple dataset with an accuracy of 97.91% and 81.90% for lane and obstacle detection respectively. [12] propose a new, efficient design for object detection in autonomous vehicles using the YOLOv5. The authors adopted the Python platform, Roboflow tool, and Google Colab for model design, praising Roboflow's features and Google Colab's accessibility and power. The authors of [19] propose a real-time method for vehicle tracking and lane detection. Vehicle tracking uses a static camera and an adaptive background subtraction technique for detection, with the Kalman filter estimating the car's position. Lane detection, performed with an onboard camera, involves pre-processing, edge detection, and broken line detection. The method achieves a detection accuracy of 92.03% and a processing speed of 48 fps.

3. PROPOSED METHOD

The proposed research encompasses three fundamental aspects: lane detection, curve fitting, and vehicle detection as depicted in **Fig. 1**. Lane detection refers to the process of pinpointing and following the edges of lanes on a roadway, a critical task for keeping the vehicle in its designated lane and ensuring safe navigation. The lane detection pipeline compromises various techniques, including filtering, edge detection, Hough transform, ROI extraction, and then inverse perspective transformation to obtain a bird's eye view. Curve fitting is the concept of constructing a curve that best fits a series of data points. In the context of this work, curve fitting could be used to model the trajectory of the detected lanes, especially in scenarios where the lanes are not straight but curved. Polynomial fitting, particularly quadratic, is commonly used for this purpose. Vehicle detection is another vital aspect of autonomous driving and ADAS, ensuring the vehicle's ability to perceive other vehicles in its vicinity to avoid collisions and maintain safe distances. Techniques for vehicle detection can range from traditional computer vision methods, such as Haar cascades and Histogram of Oriented Gradients (HOG), to more advanced deep learning-based approaches like YOLO. Together, these three aspects form the backbone of the proposed work, each contributing to creating a comprehensive and robust system for autonomous driving. The integration of these components is expected to result in a system capable of accurately detecting lanes, predicting their trajectory, and identifying other vehicles, thereby ensuring safe and efficient navigation.

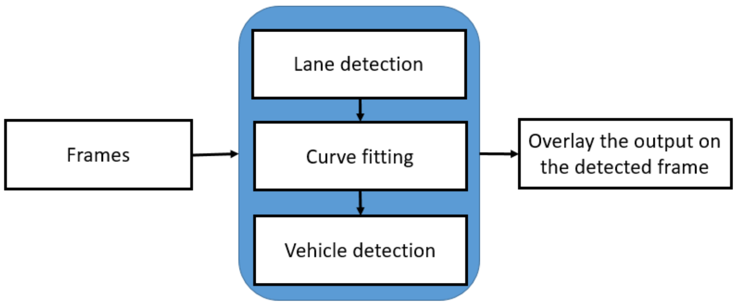

Fig. 1. Autonomous vehicle system

A flowchart that outlines an algorithmic process for analyzing video frames, with a focus on lane detection and vehicle tracking was presented in **Fig 2.** It begins by reading frames from a video source and then proceeds to detect lanes within those frames. Afterward, ROI is extracted, focusing on the relevant area of the road. Perspective transformation adjusts the view to a bird's-eye perspective, enhancing accuracy in lane detection. The sliding window search technique is employed to locate lane lines or other features efficiently. This technique involves dividing the image into smaller windows and analyzing each window to find relevant features. At a decision point, the system checks whether it has previously detected a lane line. If so, it refines the lane line, fits its curvature, and detects vehicles. If no previous line was detected, the sliding window search is reinitiated. The process concludes with the final step.

3.1 Lane Detection

A sequence of steps is often followed to analyze and interpret image data for lane detection. The process begins with the conversion of a colored image into a grayscale one, simplifying the data without losing essential information. This grayscale image then undergoes a smoothing process using a bilateral filter, which is designed to reduce noise while preserving edges, thereby enhancing the quality of the image for further processing. Following this, OTSU's method [4] is applied to segment the smoothed image. This method separates ROI by optimizing intra-class variance, a measure of the variability within classes [8].

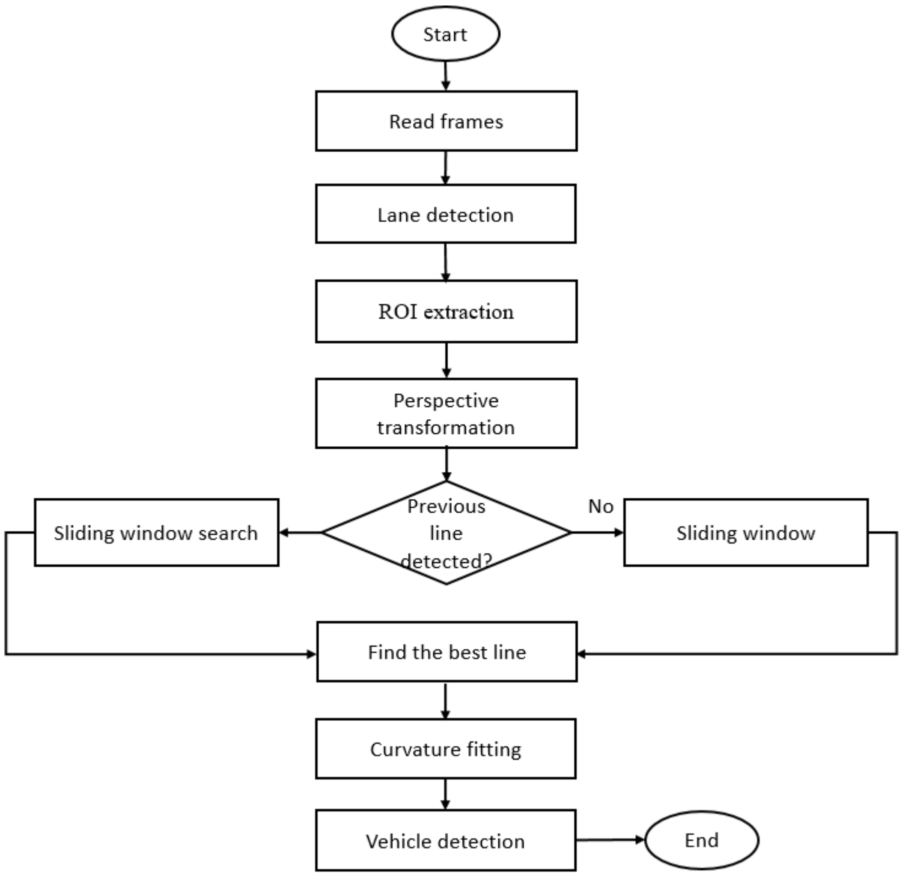

Fig 2. Lane detection and vehicle tracking flowchart

Concurring with Suddamalla [20], our preference leans towards the Canny technique for efficient edge detection. This is because, upon conducting experiments with a range of edge detection filters, it's noticed that the Canny approach produces edges that are not only thin (limited to a single pixel in width) but also better connected, a characteristic due to use of hysteresis-based thresholding. It uses a multi-stage process to detect a wide range of edges in images [21] which involves smoothing the image with a Gaussian filter, computing the gradient of the image, applying non-maximum suppression, and using double thresholding and edge tracking by hysteresis. The Hough transform identifies straight lines based on the principle of accumulating pixel points, resulting in a superior line-fitting outcome [3][19][21]. However, the Hough transform, while effective, can often detect multiple erroneous lines. To mitigate this issue and decrease both computational time and complexity, it's essential to discard these incorrect lanes. For example, post the application of the Hough transform on the binarized frame, the two lengthiest lines are chosen to bypass this problem. A comprehensive pipeline is given in **Fig. 3**, contributing to the overall lane detection method.

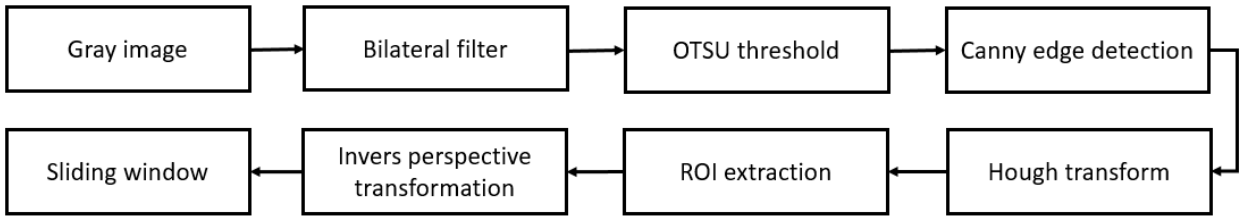

Fig. 3. Lane detection framework

ROI is usually defined as a convex polygon that includes the lane lines and their immediate surroundings while excluding unrelated objects and features. It's noted that the essential details about lane markings are predominantly located in the bottom half of the image, making it unnecessary to process the entire image for lane detection [8] [14]. This observation is especially applicable to videos in datasets, which are often captured with the camera centrally placed behind the windshield. As the vehicle advances, the road consistently remains in front, making the lower part of the image the suggested ROI. Choosing the appropriate ROI can efficiently reduce computational time and minimize environmental noise [4]. This involves selecting a trapezoidal region within the image that encompasses the lane area of interest. The trapezoidal shape accounts for the perspective distortion caused by the camera’s position relative to the road surface. Once the manual ROI is established, it serves as a template for subsequent frames. During real-time lane detection, the automated process dynamically adjusts the ROI based on the initial manual definition. By combining manual setup with dynamic adjustments, the method achieves accuracy, real-time performance, and robustness. In perspective views, parallel lines such as lane markings appear to converge at a distant point. called the vanishing point. Perspective transformations are used to map points from one perspective (e.g., a camera view) to another (e.g., birds eye view) which is beneficial for lane detection. To perform a perspective transformation, we need to select four points in the original image which should be stretched to form a trapezoidal shape that outlines the desired ROI. The transformation then maps these four points to a rectangular shape, effectively flattening the perspective as investigated in **Fig. 4**.

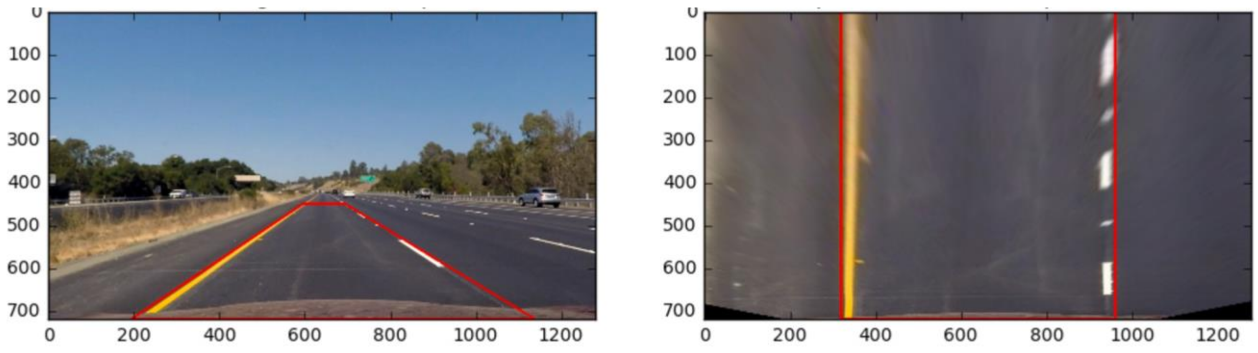

Fig. 4. ROI and perspective transformation

The sliding window method plays a pivotal role in lane detection algorithms, particularly for identifying lane markings in images. Its purpose is to systematically scan an image and identify regions likely to contain lane markings. The process begins by defining a sliding window as a rectangular region with fixed dimensions (width and height). Starting from the bottom of the image (near the vehicle), the first window is placed at a reasonable height within the ROI (e.g., near the bottom of the ROI). The ROI typically encompasses the area where lane markings are expected to appear. Within each window, a histogram of pixel intensities is computed. This histogram represents the distribution of lane-related features (such as gradients or color values). The peaks in the histogram correspond to lane markings. These peaks indicate areas where lane lines are likely to be present. The window position is updated based on the peak location. Specifically, the window is centered around the peak, ensuring that it aligns with the lane markings. The next window is then constructed based on the previous one. This sliding process continues upward, with a predefined step size (often overlapping with the previous window). By iteratively scanning the image using sliding windows, the entire lane region is covered. The sliding window method then complements edge detection, Hough transform, and curve fitting to ensure robust lane tracking even in challenging scenarios.

3.2 Curvature Fitting

The Hough transform method is a commonly used algorithm for detecting lanes in many previous studies. However, this method was originally designed for straight-line recognition and falls short when it comes to identifying curved lanes [8] [15] [22]. The accurate detection and modeling of road lane curvature, especially in high curvature radius scenarios, play a pivotal role in enhancing the performance and safety of autonomous driving systems. In this context, the utilization of second-degree polynomials for curvature fitting has emerged as a promising approach to effectively capture the intricate dynamics of curved lanes. By employing second-degree polynomials, which offer a more flexible and adaptive curve-fitting model compared to linear methods, we can better approximate the complex shape of lanes with sharper curvature radii. This method allows for a more nuanced representation of curved road geometry, enabling autonomous vehicles to navigate challenging bends with increased precision and reliability. Moreover, the use of second-degree polynomials for curvature fitting facilitates the incorporation of curvature constraints and smoothness criteria, essential for ensuring smooth and natural lane tracking in high curvature radius environments. However, it's important to note that this process assumes a relatively flat road surface and may not be accurate on hilly or uneven terrain. Despite this, curvature calculation remains a fundamental component in the lane detection pipeline. A second-degree polynomial, also known as a quadratic function, can be represented by the following equation

$$y = Ax^2 + Bx + C \qquad (1)$$

where A, B and C are constants, and x, y are the variables. In the context of lane detection, the x and y coordinates correspond to the pixel positions in the image. The constants A, B and C are determined based on the detected lane points in the image. Once these constants are determined, equation 1 will be used to predict the y position for any given x position along the lane, effectively fitting a curve to the lane. This method is particularly effective for detecting curved lanes as it can model the curvature of the lane, unlike methods designed to detect straight lines. By fitting a curvature to the lane, the system can accurately predict the path of the lane, even when it curves, enhancing the accuracy and reliability of the lane detection process. To find the curvature of a lane, we can use the formula for the curvature of the lane based on a second-degree polynomial equation. The curvature k at a specific point (x) is given by the following formula:

$$K = \frac{|y''|}{(1 + (y')^2)^{3/2}} \qquad (2)$$

where $y = f(x)$ is the function describing the curve, y' is the first derivative of the function, representing the slope of the tangent line to the curve at a given point, y'' is the second derivative of the function, representing how the slope of the tangent line is changing. We then compute the first and second derivatives as:

$$y' = 2\,Ax + B \qquad (3)$$

and

$$y'' = 2A. \qquad (4)$$

Substituting these into the curvature formula gives:

$$K = \frac{|2A|}{(1+(2Ax+B)^2)^{3/2}} \qquad (5)$$

where, x would be the x-coordinate at the point where the curvature is to be determined (typically the bottom of the image, where the car is located), and A , B are the coefficients of the second-degree polynomial that was fit to the lane lines. According to [14], the radius of curvature R is calculated by taking the inverse of the curvature K as indicated by equation 6.

$$R = \frac{1}{K} \qquad (6)$$

This computation yields the radius of the smallest possible circle that can be drawn tangent to the curve at a specific point. This serves as an indicator of the sharpness of the curve's turn at that point. A smaller radius indicates a sharper turn. It is worth mentioning that, the units of the curvature will be in pixels, so a conversion factor may be needed to convert this to real-world units (like meters) based on the resolution of the image and the known size of objects in the image. By utilizing mathematical models to fit the curvature of lanes, the system can better understand and predict the complex road geometries, such as curves and bends, that vehicles encounter. These equations provide a more accurate representation of lane shapes, enabling the system to anticipate lane deviations more effectively and navigate challenging road scenarios with greater precision. The incorporation of curvature fitting equations not only improves the system's ability to interpret and follow lane markings in complex environments but also enhances its overall performance metrics. By accurately capturing the curvature of the lanes, the system can make more informed decisions, such as adjusting steering angles or lane-keeping maneuvers, leading to smoother and more stable driving behavior. Additionally, the use of curvature fitting equations can help reduce false positives or negatives in lane detection, improving the system's reliability and reducing the risk of errors in autonomous driving applications.

3.3 Object Detection

In the context of self-driving vehicles, detecting objects can offer crucial context about the environment around the vehicle, significantly influencing its decision-making process [23]. The latest progress in deep learning has demonstrated significant effectiveness in this area. Among the numerous available frameworks, we have opted to use the YOLO object detector as the foundational framework for vehicle recognition in autonomous driving. In contrast to conventional techniques that necessitate multiple iterations for object detection, YOLO achieves this in just one iteration, thereby enhancing speed while maintaining precision [1] [13]. YOLO has been trained with an extensive collection of images and can detect a wide variety of objects [24]. This can make the system more robust to different road conditions and scenarios [12]. YOLO's ability to process images at depth layers enables it to extract complex features from input images. Its training on over 80 different classes, allows it to accurately identify and classify vehicles in various driving conditions.

4. DATASET

To assess the effectiveness of various techniques, it's crucial to set benchmarks and evaluate algorithms used for detecting lanes and roads. In our lane detection algorithm, the input typically consists of images or video frames captured by a camera mounted on a vehicle. These input images serve as the basis for our algorithm to analyze and detect lane markings. By specifying the type and resolution of the input images, we can provide a more comprehensive description of the dataset used in the evaluation. Several datasets are publically available for lane detection, such as the Tusimple, and KITTI datasets. The challenging Tusimple lane marking dataset comprises 3,626 training and 2,782 testing clips [15]. These clips are recorded under varying weather conditions and at different times. Every one-second video segment consists of 20 successive frames, with only the lane lines in the 20th frame officially marked as the ground truth [25]. KITTI Road dataset encompasses 6 hours of varied traffic situations, captured at a frequency of 10-100 Hz using an array of sensors such as high-definition cameras, a 3D laser scanner, and an accurate GPS/IMU system [26]. It contains 289 training images and 290 testing images separated into three categories [22]. Some statistics of the underlying datasets are tabulated in **Table 1**.

Table 1. Dataset characteristics

Dataset	Train	Test	Resolution	Scenes
Tusimple	3268	2782	1280 x 720	Highway
KITTI	289	290	1242 x 375	Urban streets

5. IMPLEMENTATION AND RESULTS

The model was developed using the Python platform and Google Colab. Google Colab is a free platform that allows Python code execution in the cloud, has a user-friendly interface, and provides access to high-performance GPUs. Grayscale conversion reduces the image from three color channels to one, simplifying the information and minimizing the computational complexity. A bilateral filter from the OpenCV library was applied to the grayscale image. This filter was chosen because it smooths out textures while preserving edges, which is crucial for enhancing the distinction of lane lines. Additionally, it exhibits superior performance in scenarios, effectively distinguishing between the object and the background. Utilizing this thresholding as an initial step before Canny edge detection can be advantageous as it aids in diminishing the noise level in the image, potentially enhancing the performance of the Canny algorithm. The Hough transform was used to identify lines within the binary edge-detected image. The ROI extraction was carried out manually by defining a trapezoidal region which effectively decreases the occurrence of false positives in lane detection. The output is fed into the perspective transformation module, where the image undergoes transformation based on the source and destination coordinates, and a warped perspective is applied to the image frame. This step offered a clearer visualization and analysis of lanes. Finally, the sliding window method is notably efficient as it not only pinpoints the lane lines but also offers an estimation of the vehicle's location concerning these lines. This was one of the more challenging aspects of the implementation, as it required careful tuning to ensure accurate lane detection. **Fig. 5** illustrates the sequential steps involved in the lane detection algorithm.

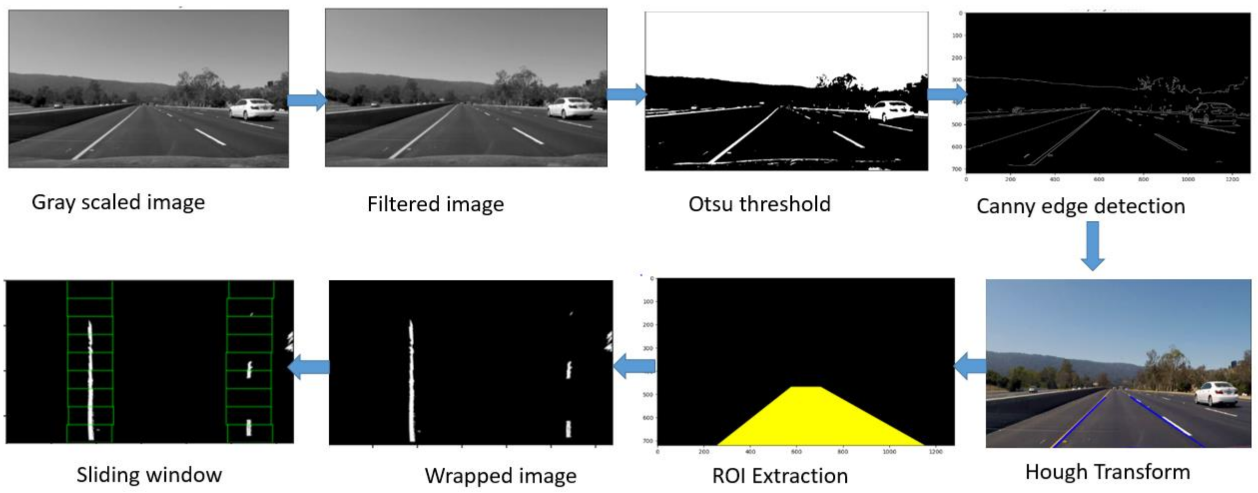

Fig. 5. Lane detection algorithm visualization

To estimate the lane curvature, a second-order polynomial was fitted to the lane line points. This allowed for the calculation of the curvature radii of the lane, providing valuable information for steering control. Lane lines, masks, directions, and curvature radii are then embedded and overlaid on the original images as indicated in **Fig. 6.** The algorithm's performance under varying road geometries, including straight paths along with left and right turns, is demonstrated. In each case, the detected lane is marked, and the estimated curve radius is provided.

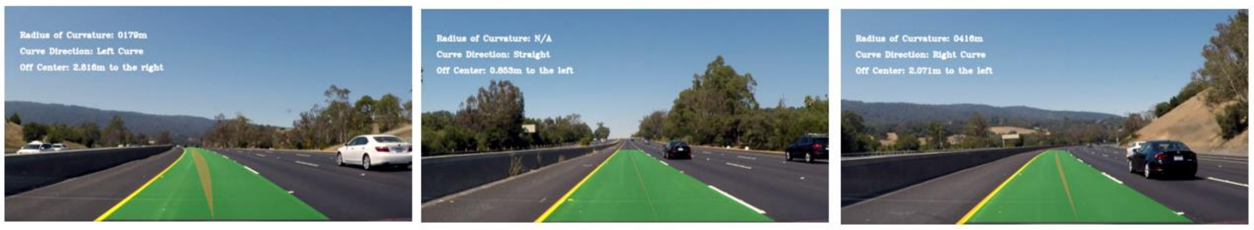
Radius of Curvature: 0179m
Curve Direction: Left Curve
Off Center: 2.816m to the right
Radius of Curvature: N/A
Curve Direction: Straight
Off Center: 0.853m to the left
Radius of Curvature: 0416m
Curve Direction: Right Curve
Off Center: 2.071m to the left

Fig. 6. Curvature estimation and directional analysis

This illustrates the algorithm's ability to handle different road geometries and provide crucial information for vehicle navigation and control. In addition to lane detection, YOLO, a real-time object detection system was used to identify other vehicles, pedestrians, and obstacles on the road. This was implemented using the Darknet framework which requires careful tuning and optimization to achieve satisfactory performance. An example is presented in **Fig. 7** where every object is associated with a confidence value which is used as a minimum value to filter weak and unwanted detections. One of the main challenges encountered during the implementation was the computational complexity of the algorithm, especially when incorporating the YOLO architecture.

To address this, the implementation was optimized for efficiency, and the YOLO model was carefully selected and configured to balance performance and computational cost. YOLO is capable of detecting several cars at once due to its unique ability to partition the input image into a grid and forecast bounding boxes and class probabilities for each grid cell. This facilitates real-time vehicle detection and tracking, which is essential for decision-making processes in autonomous driving systems.

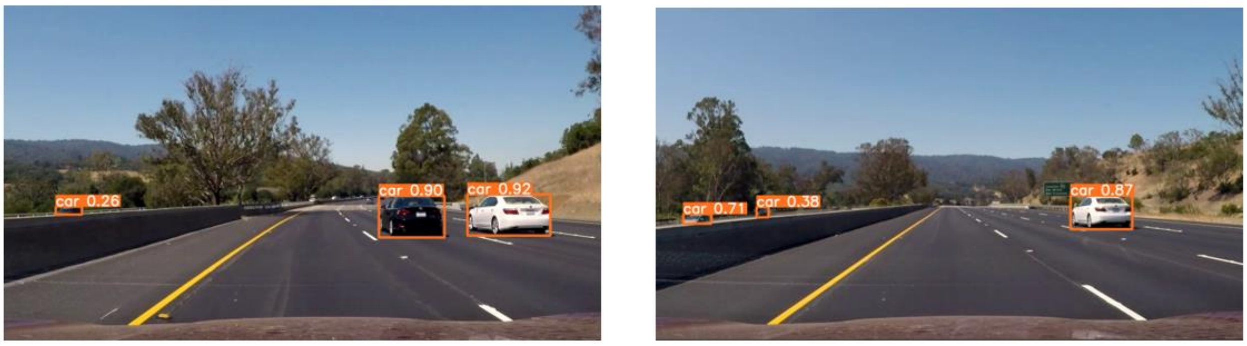

Fig. 7. Vehicle detection using YOLO

The effectiveness of the proposed method is assessed using Tusimple and only 195 frames are sourced from the KITTI dataset as demonstrated in **Table 2**. Using accuracy as the ratio of correctly detected lanes over the whole dataset allows for a direct comparison of the algorithm's performance across different datasets or experimental conditions. It serves as a reliable indicator of the algorithm's ability to correctly identify lane markings, which is essential for the successful implementation of lane detection in real-world applications such as autonomous driving. This straightforward measure simplifies the evaluation process and can be easily interpreted in terms of the algorithm's effectiveness in detecting lanes accurately. Additionally, considering processing time as a performance metric provides insights into the computational efficiency of the algorithm, which is crucial for real-time applications. The method proved impressive accuracy on both datasets, achieving 98.64% on the Tusimple dataset and 96.92% on the KITTI dataset. Furthermore, the method’s efficiency is evident in the average processing times, which were 22.1 ms/frame and 21.9 ms/frame for the Tusimple and KITTI datasets, respectively.

Table 2. Comparison with the previous works

Dataset	Total frames No	Correct recognition No.	False recognition No.	Accuracy %	Average processing time (ms/Frame)
Tusimple	2286	2255	31	98.64	22.1
KITTI	195	189	6	96.92	21.9

Tables 3 provides a clear comparison between traditional methods [3] [11], deep learning approaches [18], and the proposed techniques for lane detection. It considers computational efficiency through processing times and hardware environments used for testing these algorithms. The proposed methods aim to improve accuracy while minimizing computation time compared to existing approaches. Traditional approaches employed by [3] and [11] surpasses deep learning techniques [18] in both accuracy and efficiency. The proposed method outperforms traditional methods, achieving an impressive accuracy of 98.64% and 96.92 % on the Tusimple and KITTI datasets respectively. The proposed method also demonstrates a significantly faster average processing time than other methods, further highlighting its efficiency and real-time applicability. The traditional approaches and deep learning techniques also show competitive performance, with accuracies above 95% and reasonable processing times. Notably, employing a GPU significantly reduces the processing time in some cases, as seen in deep learning [18] and traditional methods [11].

Table 3. Comparison with previous works, HT, Hough Transform, SWS, sliding window search

Method	Algorithm	Applied Dataset	# of samples per dataset	Accuracy %	Average proc. time (s)	Environmental Setting
Traditional methods[3]	HT	1500 Cityspace pictures	1500	95.7	0.06540	i7-6700K and Matlab2016a platforms
Traditional methods [11]	HSL + Sobel filter + SWS	Tusimple	3626 training and 2782 testing videos	97.91 97.91	0.08500 0.0021	Intel Core i5-9300H, CPU@ 2.4 GHz GPU GeForce GTX TITAN X
		KITTI	195	85.13	0.086	Intel Core i5-9300H CPU@ 2.4 GHz
Deep learning[18]	FastDraw ResNet	Tusimple	------	95.2	0.06533	NVIDIA GeForce GTX 1080, GPU
Proposed	Bilateral filter + OTSU + Canny+ HT+SWS	Tusimple	2286	98.64 98.64	0.0221 0.0166	Intel Core i7- 7700K, CPU and 16 GB DDR4 RAM. GPU Tesla K80 12GB
		KITTI	195	96.92	0.0219	Intel Core i7- 7700K, CPU and 16 GB DDR4 RAM.

In this study, we also investigate the time variation associated with frame processing in the context of lane detection. **Fig. 8** depicts a curve representing time intervals in ms across the corresponding frames. Notably, the average time aligns closely with the desired value of 22.1 ms. This analysis sheds light on the temporal dynamics of lane detection algorithms and provides insights for optimizing real-time processing in autonomous driving systems.

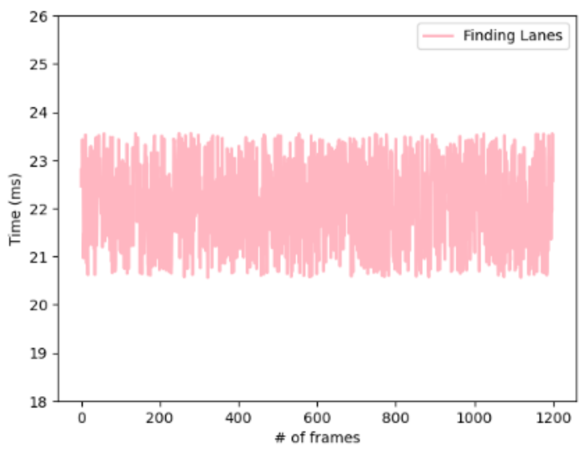

Fig. 8. Time variation over a number of frames

Table 4 presents a concise overview of different methods, their associated accuracy, average processing times, and the environmental settings in which they operate. The proposed method using YOLOv5 achieves the highest accuracy (90.03%) while maintaining competitive processing time (0.89 s).

Table 4. Comparison of object detection algorithms.

Method	Algorithm	Accuracy %	Average proc. time (s)	Environmental Setting
Traditional methods [11]	YOLOv4	81.9 81.9	0.91 0.022	Intel Core i5-9300H, CPU@ 2.4 GHz GPU NVIDIA TITAN
Neural Networks [27]	Faster R-CNN	81.6	2.0	NVIDIA TITAN X and GP106 (DrivePX2)
Proposed	YOLOv5	90.03	0.89	GPU Tesla K80 12GB

[11] utilizes the YOLOv4 algorithm achieving an accuracy of 81.9%, and the average processing time is relatively resonable at 0.91 seconds. Faster R-CNN used by [27] records a similar accuracy of 81.6% but exhibits a longer processing times around 2 seconds. The proposed method exploits YOLOv5 and achieves the highest accuracy at 90.03% and a competitive average processing time of 0.89 seconds. The optimization within YOLOv5 strikes an excellent balance between accuracy and real-time processing demands. For all methods, the remarkable speed enhancement is attributed to the GPU acceleration.

6. DISCUSSION

In this study, we proposed a lane detection algorithm for autonomous driving systems that combines traditional computer vision techniques with deep learning methodologies. The algorithm demonstrated high accuracy of 98.64% and 96.92% on Tusimple and KITTI datasets respectively. The method registers relatively low processing times of 22.1 ms/frame and 21.9 ms/frame for Tusimple and KITTI datasets, showcasing its efficiency and effectiveness in lane detection tasks. When comparing our lane detection algorithm with existing strategies, we observed that traditional approaches and deep learning techniques also exhibited competitive performance, with accuracies above 95% and reasonable processing times. However, our proposed method stood out with the highest accuracy and a relatively low processing time, indicating its efficiency and effectiveness in lane detection applications. Through the integration of advanced curve fitting techniques, such as higher-order polynomials and adaptive curve approximation methods, we aim to improve the precision and robustness of our system in detecting and navigating high curvature radii. By focusing on refining our curvature fitting capabilities, we are confident that the proposed work will be equipped to effectively handle complex driving scenarios, ensuring safe and efficient navigation in real-world environments. The integration of deep learning methodologies, such as the YOLO object detection system, enhances the algorithm's ability to identify vehicles, pedestrians, and obstacles on the road, crucial for decision-making processes in autonomous vehicles. Moving forward, there are several potential areas for further research and enhancement of lane detection algorithms. One avenue for exploration is the adaptation of the algorithm to handle dynamic lane changes, varying road conditions, and complex scenarios. Additionally, conducting experiments on larger and more diverse datasets to validate the algorithm's effectiveness across a wide range of real-world situations would be beneficial for refining its capabilities and ensuring its applicability in various driving contexts.

7. CONCLUSION

This research has made significant strides in the field of autonomous vehicles, particularly in the areas of lane detection, curve fitting, and object detection. The meticulously designed lane detection system, which employs a series of computer vision techniques, has proven to be robust and reliable in identifying and tracking lanes under various driving conditions. The application of a second-order polynomial for curve fitting has provided an accurate mathematical model for representing the curvature of detected lanes. Furthermore, the deployment of object detection methods has improved the comprehensive situational understanding of autonomous driving systems by efficiently identifying and categorizing objects in the driving scene. The proposed methods have been validated through rigorous experiments on the Tusimple and KITTI datasets, achieving impressive accuracies of 98.64% and 96.92% respectively. These results underscore the potential of the proposed methods in revolutionizing transportation through the advancement of autonomous vehicles. The system can collaborate with other autonomous cars and infrastructure, such as traffic management systems and smart cities, to promote sustainable and efficient transportation. Future work will focus on extensive real-world testing to validate the robustness of the proposed methods. Additionally, integration with other ADAS features will be pursued to enhance the autonomous driving system's overall performance and safety.

REFERENCES

[1] Henry, A., Rahesh, R., Das Barman, K., Sujee, R. (2022). Lane Detection and Distance Estimation Using Computer Vision Techniques. In: Khare, N., Tomar, D.S., Ahirwal, M.K., Semwal, V.B., Soni, V. (eds) Machine Learning, Image Processing, Network Security and Data Sciences. MIND 2022. Communications in Computer and Information Science, vol 1763. Springer, Cham. https://doi.org/10.1007/978-3-031-24367-7_2

[2] J. Zhang, T. Deng, F. Yan and W. Liu, "Lane Detection Model Based on Spatio-Temporal Network With Double Convolutional Gated Recurrent Units," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 6666-6678, July 2022, doi: 10.1109/TITS.2021.3060258.

[3] F. Zheng, S. Luo, K. Song, C.-W. Yan, and M.-C. Wang, "Improved lane line detection algorithm based on hough transform," Pattern Recognition and Image Analysis, vol. 28, no. 2, pp. 254–260, 2018.

[4] S. Sultana, B. Ahmed, M. Paul, M. R. Islam, and S. Ahmad, "Vision-Based Robust Lane Detection and Tracking in Challenging Conditions," in IEEE Access, vol. 11, pp. 67938-67955, 2023, doi: 10.1109/ACCESS.2023.3292128

[5] Jia-Qi Zhang, Hao-Bin Duan, Jun-Long Chen, Ariel Shamir, Miao Wang, "HoughLaneNet: Lane detection with deep hough transform and dynamic convolution, Computers & Graphics", Elsevier, Volume 116, 2023, Pages 82-92, ISSN 0097-8493,https://www.sciencedirect.com/science/article/pii/S0097849323001814

[6] Wang, B., Wang, Z., Zhang, Y. (2020). Polynomial Regression Network for Variable-Number Lane Detection. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12363. Springer, Cham. https://doi.org/10.1007/978-3-030-58523-5_42

[7] S. Jung, J. Youn and S. Sull, "Efficient Lane Detection Based on Spatiotemporal Images," in IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 1, pp. 289-295, Jan. 2016, doi: 10.1109/TITS.2015.2464253.

[8] Dorj B, Hossain S, Lee D-J. Highly Curved Lane Detection Algorithms Based on Kalman Filter. Applied Sciences. 2020; 10(7):2372. https://doi.org/10.3390/app10072372

[9] Kumar, S., Jailia, M. & Varshney, S. An efficient approach for highway lane detection based on the Hough transform and Kalman filter. Innov. Infrastruct. Solut.7, 290 (2022). https://doi.org/10.1007/s41062-022-00887-9

[10] H. Zhu, K. -V. Yuen, L. Mihaylova and H. Leung, "Overview of Environment Perception for Intelligent Vehicles," in IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 10, pp. 2584-2601, Oct. 2017, doi: 10.1109/TITS.2017.2658662.

[11] Phat Nguyen Huu, Quyen Pham Thi, Phuong Tong Thi Quynh, "Proposing Lane and Obstacle Detection Algorithm Using YOLO to Control Self-Driving Cars on Advanced Networks", Advances in Multimedia, vol. 2022, ACM digital library, Article ID 3425295, 18 pages, 2022. https://doi.org/10.1155/2022/3425295

[12] M. I. M. Fakharurazi, A. Z. Jusoh, A. L. Asnawi, N. F. A. Malek, K. Abdullah and N. F. M. Azmin, "Object Detection in Autonomous Vehicles," 2023 IEEE 13th International Conference on System Engineering and Technology (ICSET), Shah Alam, Malaysia, 2023, pp. 177-181, doi: 10.1109/ICSET59111.2023.1029

[13] J. I. and Q. Tian, "Adversarial Attack and Defense of YOLO Detectors in Autonomous Driving Scenarios," 2022 IEEE Intelligent Vehicles Symposium (IV), Aachen, Germany, 2022, pp. 1011-1017, doi: 10.1109/IV51971.2022.9827222.

[14] Cao J, Song C, Song S, Xiao F, Peng S. Lane Detection Algorithm for Intelligent Vehicles in Complex Road Conditions and Dynamic Environments. Sensors (Basel). 2019 Jul 18;19(14):3166. doi: 10.3390/s19143166. PMID: 31323875; PMCID: PMC6679325.

[15] Q. Wang, T. Han, Z. Qin, J. Gao and X. Li, "Multitask Attention Network for Lane Detection and Fitting," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 3, pp. 1066-1078, March 2022, doi: 10.1109/TNNLS.2020.3039675

[16] T. Getahun, A. Karimoddini and P. Mudalige, "A Deep Learning Approach for Lane Detection," 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 2021, pp. 1527-1532, doi: 10.1109/ITSC48978.2021.9564965.

[17] J. -M. Guo and H. Markoni, "Deep Learning Based Lane Line Detection and Segmentation Using Slice Image Feature," 2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Hualien City, Taiwan, 2021, pp. 1-2, doi: 10.1109/ISPACS51563.2021.9651012.

[18] J. Philion, “Fastdraw: addressing the long tail of lane detection by adapting a sequential prediction network,” in Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11 574–611 583, Long Beach, CA, USA, June 2019.

[19] Datta, T.; Mishra, S.K.; Swain, S.K. Real-Time Tracking and Lane Line Detection Technique for an Autonomous Ground Vehicle System. In Proceedings of the International Conference on Intelligent Computing and Smart Communication 2019; Springer: Singapore, 2020; pp.1609–1625.https://doi.org/10.1007/978-981-15-0633-8_156

[20] U. Suddamalla, S. Kundu, S. Farkade, and A. Das, “A novel algorithm of lane detection addressing varied scenarios of curved and dashed lane marks,” in Proc. Int. Conf. Image Processing Theory, Tools and Applications (IPTA), Orleans, France, 2015, pp. 87–92.

[21] Andrei, M.-A.; Boiangiu, C.-A.; Tarb ˘a, N.; Voncil ˘a, M.-L. Robust Lane Detection and Tracking Algorithm for Steering Assist Systems. Machines 2022, 10, 10. https://doi.org/10.3390/ machines10010010

[22] Y. Xing et al., "Advances in Vision-Based Lane Detection: Algorithms, Integration, Assessment, and Perspectives on ACP-Based Parallel Vision," in IEEE/CAA Journal of Automatica Sinica, vol. 5, no. 3, pp. 645-661, May 2018, doi: 10.1109/JAS.2018.7511063

[23] Aduen Benjumea and Izzedin Teeti and Fabio Cuzzolin and Andrew Bradley, YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles, ArXiv, volume abs/2112.11798,2021, url=https://api.semanticscholar.org/CorpusID:245385732s

[24] Lakshmi Priya, S.P., Karunya, T., Praveen Kumar, R., Durai Arumugam, S.S.L. (2023). Vehicle Detection in Autonomous Vehicles Using Computer Vision. In: Ranganathan, G., EL Allioui, Y., Piramuthu, S. (eds) Soft Computing for Security Applications. ICSCS 2023. Advances in Intelligent Systems and Computing, vol 1449. Springer, Singapore. https://doi.org/10.1007/978-981-99-3608-3_2

[25] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen and Q. Wang, "Robust Lane Detection From Continuous Driving Scenes Using Deep Neural Networks," in IEEE Transactions on Vehicular Technology, vol. 69, no. 1, pp. 41-54, Jan. 2020, doi: 10.1109/TVT.2019.2949603.

[26] Geiger, A., Lenz, P., Stiller, C. and Urtasun, R. (2013) Vision Meets Robotics: The KITTI Dataset. The International Journal of Robotics Research, 32, 1231-1237. https://doi.org/10.1177/0278364913491297

[27] C. -T. Lin, P. S. Santoso, S. -P. Chen, H. -J. Lin and S. -H. Lai, "Fast Vehicle Detector for Autonomous Driving," 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 2017, pp. 222-229, doi: 10.1109/ICCVW.2017.35.