International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Lane Following and Vehicle Detection Using Yolo-V3 Simulink

Ms. A. Sujatha Reddy[1], Haarini Tadiparthi[2], Krishna Haneesha Gangisetty[3], Sreeja Chittampally[4], Prathyusha Bokkala[5]

[1]Assistant Professor, G. Narayanamma Institute of Technology and Science, India. [1]sujathaallipuram@gnits.ac.in,
[2]G. Narayanamma Institute of Technology and Science, India.
[2]tnp.haarini@gmail.com, [3]krishnahaneesha@gmail.com, [4]sreejachittampally92@gmail.com, [5]bokkalaprathyusha02@gmail.com

Doi: https://doi.org/10.55248/gengpi.5.0424.10139

ABSTRACT—

This project presents a novel integration of lane following and vehicle detection utilizing the YOLOv3 algorithm within the Simulink framework. The primary goal is to enhance autonomous driving systems by simultaneously addressing lane discipline and real-time identification of surrounding vehicles.

Through the fusion of YOLOv3's advanced object detection capabilities and Simulink's simulation environment, the proposed solution entails training the

YOLOv3 model with annotated datasets specific to lane markings and vehicle instances. This trained model is then embedded in Simulink, enabling comprehensive evaluations of both lane-following algorithms and vehicle detection mechanisms. The synergistic approach ensures the seamless coordination of steering control for lane adherence and the detection of diverse vehicles, significantly advancing the robustness and safety of autonomous driving systems.

Introduction

Key functions of advanced driver-assistance systems (ADAS) and autonomous driving are vehicle recognition and lane following. These jobs include locating and identifying the cars in a scene as well as keeping an eye on lane markings for safe driving. YOLO (You Only Look Once) v3, a cutting-edge deep learning model renowned for its effectiveness and accuracy in real-time object identification, is a widely used strategy to address these issues.

The basic idea behind YOLOv3, an improvement over its predecessors, is to take an input image, divide it into grid cells, and then estimate bounding boxes and class probabilities for items inside each grid cell. This one-stage object detection technique is appropriate for real-time applications such as autonomous driving because it delivers amazing speed without sacrificing accuracy.

YOLOv3 vehicle detection and lane following is initially implemented by training the model using annotated datasets that contain photos of vehicles with labels and lane markers. Through training, YOLOv3 picks up skills like lane boundary detection and vehicle type recognition, which let it effectively adapt to new situations.

After being taught, YOLOv3 can be used to instantly identify lanes and cars. In order to predict bounding boxes containing cars together with their related confidence scores and class labels, the process starts with feeding the input picture or video stream into the model. The model then splits the image into grid cells. To make lane following easier, YOLOv3 simultaneously detects lane markings and calculates their trajectories.

YOLOv3 vehicle detection entails post-processing the model's predictions to eliminate non-vehicle classes and bounding boxes with low confidence scores. The identified cars represented by the remaining bounding boxes can be further adjusted to reduce redundant detections and increase accuracy by employing strategies such as non-maximum suppression. YOLOv3 uses its capacity to recognize lane lines and calculate their distances from the vehicle to follow lanes. Autonomous vehicles can safely navigate within their lanes thanks to the model's ability to deduce the direction and curvature of the road from the spatial distribution of lane markings and their orientations.

YOLOv3 can be coupled with other sensor modalities, such as LiDAR and radar, to improve perception in difficult situations, such as poor light or bad weather, and to increase the robustness of vehicle detection and lane following. Furthermore, the vehicle's trajectory can be improved and navigation accuracy increased by utilizing contextual information obtained from map data and sensor fusion techniques.

To sum up, YOLOv3-based vehicle identification and lane following provide a strong option for ADAS and autonomous driving applications. Through the utilization of YOLO V3's real-time object detection capabilities, vehicles may travel roads independently and effectively sense their surroundings. YOLO V3 continues to be at the forefront of innovation, propelling the development of intelligent transportation systems toward safer and more effective mobility solutions as deep learning and computer vision research advances.

Methodology

The advancement of advanced driver assistance systems (ADAS) and driverless vehicles depends on vehicle detection and lane following. We will go over the process of using YOLOv3 (You Only Look Once version 3) in MATLAB to implement vehicle recognition and lane following in this succinct presentation.

Vehicle Detection using YOLOv3:

YOLOv3, a cutting-edge object detection algorithm renowned for its speed and accuracy, is used for vehicle detection. The following are the key steps involved:

1. Preparing the Dataset: Gather and categorize a dataset of pictures of cars. The YOLOv3 model is trained using this dataset.

2. Model Training: Train the YOLOv3 model using the labeled dataset. YOLOv3 and other deep learning models can be trained with MATLAB's tools, either from scratch or through transfer learning.

3. Model Evaluation: Evaluate the trained model's performance using metrics like mean Average Precision (mAP), recall, and precision. This stage guarantees that the model can recognize cars in a variety of situations. 4.Integration with MATLAB: For real-time inference, integrate the trained YOLOv3 model with MATLAB. Functions for loading pre-trained models and carrying out object detection on pictures or video streams are available in MATLAB.

Lane Following:

The process of lane following entails adjusting the steering to maintain the car inside the lines. The following is a summary of the MATLAB approach for achieving lane following:

1**. Lane Detection:** To identify lane markers in the recorded video frames or photos, apply computer vision techniques. Polynomial fitting, edge detection, and the Hough transform are common techniques.

2**. Lane Tracking:** It involves estimating the direction and curvature of the lane by following the detected lane markers over a series of frames. For reliable lane tracking, Kalman filters or related methods can be used.

3. **Control Algorithm:** Create a control algorithm that uses the identified lane information to determine the steering commands. For this, PID controllers or model predictive control (MPC) methods are frequently employed.

4**. Integration with Vehicle Actuators:** Use MATLAB to communicate with the steering motors and other actuators in the car to convert the calculated steering orders into motion.

Software description

MATLAB Simulink: The software platform known by its abbreviation MATLAB, or "MATrix LABoratory," is a powerful and adaptable tool that has had a profound impact on data analysis, science, engineering, and mathematics. MATLAB, created by MathWorks, is renowned for its versatility across several fields, interactive environment, computing power, and data visualization features. Its advanced programming language enables users to easily handle challenging mathematical and numerical problems and provides an extensive library of built-in functions that form the basis for investigation, creativity, and problem-solving. MATLAB provides an interactive and user-friendly environment that facilitates rapid algorithm creation, data exploration, and prototyping, whether via its integrated programming environment or command-line interface.

The dynamic modeling and simulation platform Simulink, a potent and essential part of the MATLAB software ecosystem, has completely changed the way scientists, engineers, and researchers design, simulate, and analyze complex systems and control algorithms. Simulink, created by MathWorks, is a vital tool in many different industries, such as aerospace, automotive, robotics, and communications. It offers a graphical and user-friendly environment for modeling and simulating dynamic systems. Simulink's primary strength resides in its ability to visually represent systems using block diagrams and mimic their behavior over time. Here, we explore Simulink's many features and functionalities that have made it a mainstay of system modeling and simulation.

YOLOv3 Algorithm: The YOLOv3 (You Only Look Once version 3) algorithm is a groundbreaking deep learning model for real-time object detection and classification. Developed by Joseph Redmon and Ali Farhadi, YOLOv3 represents a significant advancement in computer vision, with applications spanning from autonomous vehicles and surveillance systems to image and video analysis. The Key features of YOLOv3 algorithm are:

1. Real-Time Object Detection: One of YOLOv3's most notable strengths is its capability for real-time object detection. The algorithm can process images and video streams at impressive speeds, allowing it to analyze a constant stream of data in real-time. This feature is invaluable in applications such as self-driving cars and security surveillance.

2. One-Stage Detection: YOLOv3 is a one-stage object detection model. Unlike traditional two-stage detectors that require separate region proposal and object classification stages, YOLOv3 performs both tasks simultaneously in a single forward pass. This streamlined approach contributes to its speed and efficiency.

3. High Accuracy: YOLOv3 achieves remarkable accuracy in object detection. It can reliably detect and classify objects in a wide variety of scenes, making it a powerful tool for applications where precision is paramount. The algorithm's performance has been further improved through various iterations and fine-tuning.

4. Multi-Scale Detection: YOLOv3 utilizes a multi-scale approach for object detection. It detects objects at different resolutions or scales within a single image. This enables it to detect both small and large objects, making it versatile in a variety of scenarios.

5. Anchor Boxes: YOLOv3 incorporates the concept of anchor boxes. These predefined anchor boxes are used to predict the location and size of objects in the image. The use of anchor boxes improves the accuracy of object localization.

6. Extensive Object Categories: Yolov3 has been pre-trained on a large number of object categories, including commonplace items like vehicles, people, animals, and more. Its broad recognition range allows it to be used in a variety of domains.

7. Community Support and Open Source: YOLOv3 is available as open-source software, enabling academics and developers to improve and expand the model. The YOLO community promotes ongoing development and improvement by offering resources, updates, and pre-trained models.

8. Architectural Advancements: To better capture complex characteristics, YOLOv3 has a deeper architecture with more convolutional layers. This helps explain its remarkable accuracy along with feature pyramid scaling.

9. Customization and Transfer Learning: To make YOLOv3 more suited for certain applications, users can refine it using custom datasets. Developers can use pre-trained models with transfer learning to customize them to meet specific object detection requirements.

10. Real-World Applications: YOLOv3 has found application in a wide range of domains, including autonomous vehicles for pedestrian and vehicle detection, security systems for intruder detection, and robotics for object recognition. Its real-time capabilities have made it invaluable in scenarios where immediate response is crucial

Architecture of YOLOv3: Convolutional Neural Networks (CNNs) are classifier-based frameworks that interact with input pictures as structured arrays of data and seek to identify patterns between them (see picture below). YOLO has the advantage of being much faster than other object detection models while maintaining accuracy. The architecture of YOLOv3 feature detector was influenced by other well-known architectures such as ResNet and FPN (Feature Pyramid Network). YOLOv3's name, Darknet-53, contained 52 convolutions with skip connections like ResNet and a total of 3 prediction heads like FPN, allowing YOLOv3 to process image at a different spatial compression.

Because it allows the model to see the whole image during testing, the prediction process is influenced by the image's total global context. YOLO and other CNN algorithms "score" the regions according to how similar the images are to the pre-established classes.Positive detections of the class that the high-scoring regions most closely resemble are reported. For instance, by "looking" into the areas where the score is high relative to established classes of cars, YOLO can identify the various types of vehicles while operating on a real-time traffic stream.

SqueezeNet serves as the foundation for the YOLOv3 algorithm, which is the one employed in this project.

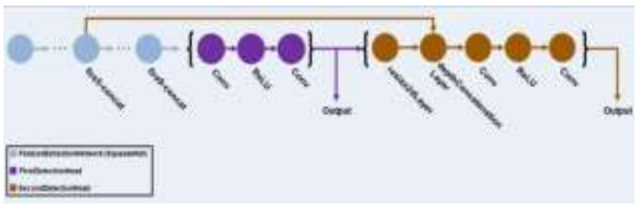

Figure-1 Architecture of YOLOv3 Algorithm

SqueezeNet: SqueezeNet is a remarkable and compact deep learning architecture that has gained significant attention in the field of computer vision. It is known for its efficiency and ability to perform image classification with high accuracy while minimizing the computational and memory resources required. When incorporated into the YOLOv3 (You Only Look Once version 3) algorithm, SqueezeNet becomes a pivotal component that contributes to real-time object detection and classification, making the YOLOv3 algorithm even more efficient and suitable for resource-constrained applications.

Researchers at DeepScale first created SqueezeNet in response to the growing demand for effective deep learning models that could function on hardware with constrained memory and processing power. It might be difficult to implement traditional deep neural networks on gadgets like smartphones, embedded systems, or Internet of Things (IoT) devices because of their many layers and parameters. This problem is addressed by SqueezeNet, which prioritizes model compression without sacrificing performance.

An essential idea of SqueezeNet is the "fire module." The architecture is constructed from these fire modules. The two primary parts of any fire module are the "expand" and "squeeze" layers. In order to compress the information, the squeeze layer mostly uses 1x1 convolutional filters to lower the number of input channels. To capture a wide range of characteristics, the expand layer then combines 1x1 and 3x3 convolutional filters. SqueezeNet is distinguished by its ability to extract features that are more compact and effective as a result of this combination.

Additionally, SqueezeNet adds bypass connections to training, enabling effective gradient flow. These linkages lessen the total computing load, preserve significant traits, and aid in the learning process. SqueezeNet is one of the smallest deep learning architectures thanks to its design principles, proving that smaller models may still attain competitive accuracy.

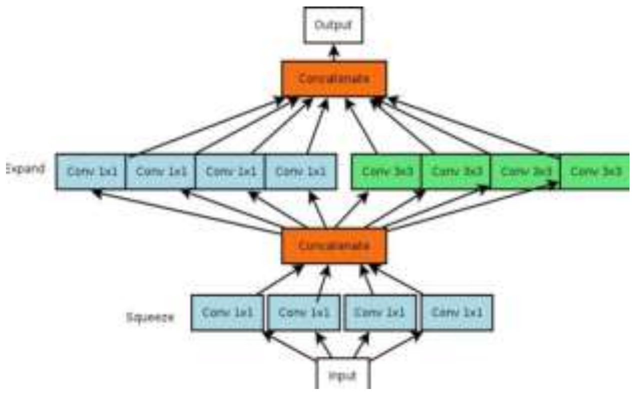

Figure-2 Architecture of SqueezeNet

III. Vehicle detection

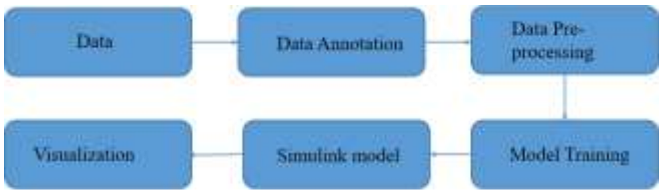

Figure-3 Block diagram of Vehicle Detection

1. Loading Data: Gathering the required data is the initial stage in creating a lane-following and vehicle-detection system. Usually, a camera mounted on a vehicle captures photos or video frames that make up this data. The system uses these photographs as its input data. The quantity and quality of the data are crucial since they have an immediate effect on the system's functionality. The data pipeline must be configured in this stage in order to gather and supply photos to the system for additional processing.

2. Data Annotation: After the data has been loaded, data annotation comes next. Data annotation is the process of identifying and categorizing interesting objects in the pictures. You would annotate the lanes and automobiles for a lane following and vehicle detection system. Usually, to do this, bounding boxes are drawn around the objects, their locations are noted, and labels are applied. Since accurate data annotation provides ground truth for model training, it is essential to the training of a machine learning model.

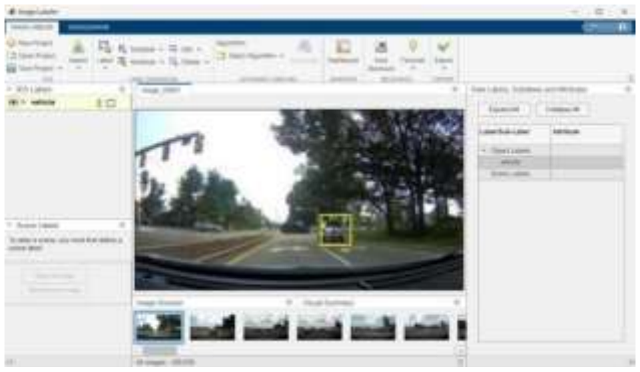

Figure-4 Data Annotation using Image Labeller

3. Data Preprocessing: Data preprocessing is an essential step in any computer vision project. It involves several tasks, such as resizing images to a consistent size, normalizing pixel values, and augmenting the data to improve model generalization. In the context of Simulink and YOLO-V3, data preprocessing may also include converting the annotated data into a format suitable for model training. This step ensures that the data is in the right form to be used in the subsequent training and simulation processes.

4. Model Training: Model training is the heart of the system development process. In this step, you use the annotated and preprocessed data to train a YOLO-V3 model. Training involves adjusting the model's parameters and learning the features that allow it to accurately detect lanes and vehicles in the images[10]. The training process typically consists of multiple iterations, with the model being updated based on its performance. The goal is to optimize the model's ability to identify and locate objects of interest in the data. The 60% of data is given as Training data and 40% of data is given as Testing data.

5. Simulink Model: In this step, you create a Simulink model to implement the trained YOLO-V3 model for real- time lane following and vehicle detection. The Simulink model includes components for image input, data preprocessing, model inference, and post-processing of model outputs. It allows you to interface the trained model with live camera data and process it to make real-time decisions about lane following and vehicle detection. The Simulink model is designed to work in a closed-loop system, ensuring that the vehicle responds to the detected objects appropriately.

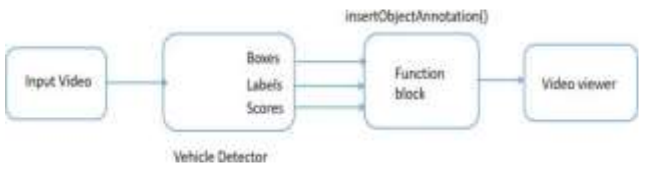

Figure-5 Simulink Model Diagram for Vehicle Detection

6. Visualization: Visualization is the final step in the process, and it is essential for monitoring and evaluating the system's performance. Visualization components in Simulink can be used to display real- time video feeds with annotated lane markings and detected vehicles. Visual feedback is vital for debugging, verifying the system's correctness, and fine-tuning its parameters. Additionally, visualization can help provide insights into how the system is making decisions and whether any adjustments are needed

IV. Lane Detection

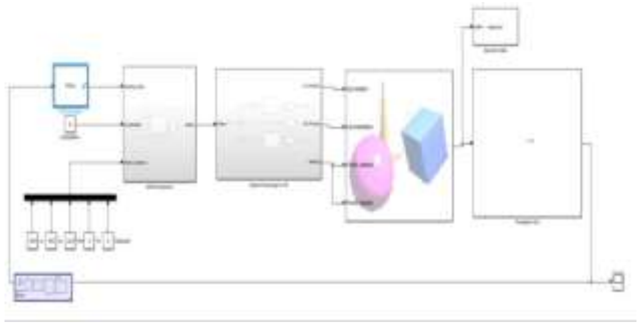

Figure-6 Block Diagram for Lane Detection

Since YOLOv3 is primarily intended for object detection tasks rather than lane detection, utilizing it for lane detection is not a usual technique. A real-time object identification system called YOLO (You Only Look Once) can recognize objects in pictures or video frames and generate bounding boxes around them.

But rather than object detection, lane detection usually uses computer vision techniques like edge detection, Hough transform, or deep learning-based segmentation algorithms.

The network design and training data would need to be changed if you were especially interested in using YOLOv3 for lane detection in order to recognize lane markers as objects. In order to predict the bounding boxes of lane markings, this method would need identifying lane markings in the training dataset and adjusting YOLOv3's output layer.

This is a simple overview of how to use YOLOv3 for lane detection:

1. Prepare the dataset by gathering a collection of roadside photos or videos that have lane markers noted on them. Lane markers should be annotated as objects with bounding boxes.
2. Formatting Training Data: Create a YOLOv3 file from the annotated dataset. The bounding box coordinates and the class (lane markings) for each image should be specified in the accompanying text file.
3. YOLOv3 Modification: To support lane detection, adjust the YOLOv3 network design. In the final layer, change the number of output channels to correspond with the number of classes (lane markings).
4. Training: Apply normal object detection training procedures to the annotated dataset to train the modified YOLOv3 model.

5. Evaluation: To gauge the accuracy of the trained model's lane detection, test its performance on a different validation dataset.
6. Conclusion: To detect lanes in fresh photos or video frames, apply the YOLOv3 model that has been trained. Analyze the model's output to retrieve and visualize the coordinates of the lane markings.

   Remember that while YOLOv3 can be modified for lane recognition, it might not be the best or most effective method when compared to conventional computer vision techniques created especially for this purpose. Alternative techniques like deep learning-based semantic segmentation networks designed for lane identification tasks or edge detection in conjunction with lane fitting algorithms can be investigated, depending on your needs and limitations.

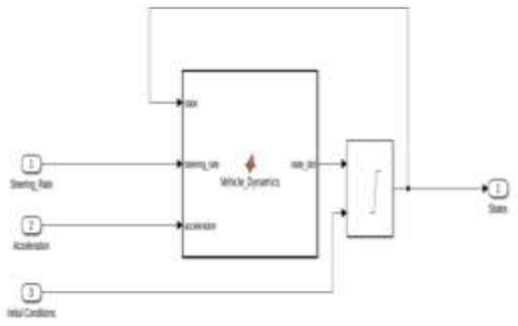

Figure-7 Block Diagram for Vehicle Dynamics

The above models the kinematics of a vehicle, providing the rates of change of its position, orientation, velocity, and steering angle based on the input parameters.

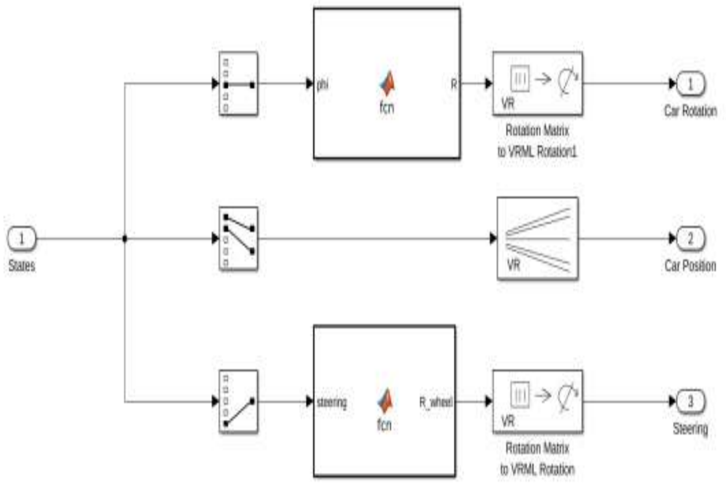

Figure-8 Block Diagram for Signal Processing for VR

The above function generates a 3x3 rotation matrix for a specified angle around the y-axis.

function_steering calculates a combined rotation matrix to model orientation changes with steering input, involving successive rotations around the x-axis and y-axis.

V. Results and discussions

1. Results Of Vehicle Detection

This project model provides a visually informative and actionable result. The primary output is the visual representation of the input video stream with yellow bounding boxes drawn around the detected vehicles. These bounding boxes serve as a visual indicator of where the system has identified vehicles within each frame of the video. This immediate visual feedback is essential for system operators, users, and developers to understand how well the object detection model is performing in real-time.

In addition to the yellow bounding boxes, the class name "CAR" is displayed within or near each bounding box. This class labeling is a crucial element that not only informs the viewer about the detected object but also enhances the interpretability of the output. It provides clear and unambiguous information about the type of object that has been detected, contributing to the overall usability of the system. The model's aptitude to correctly classify these quality attributes holds profound implications for the medicinal herb industry, where the potency of leaves is intricately tied to their quality.

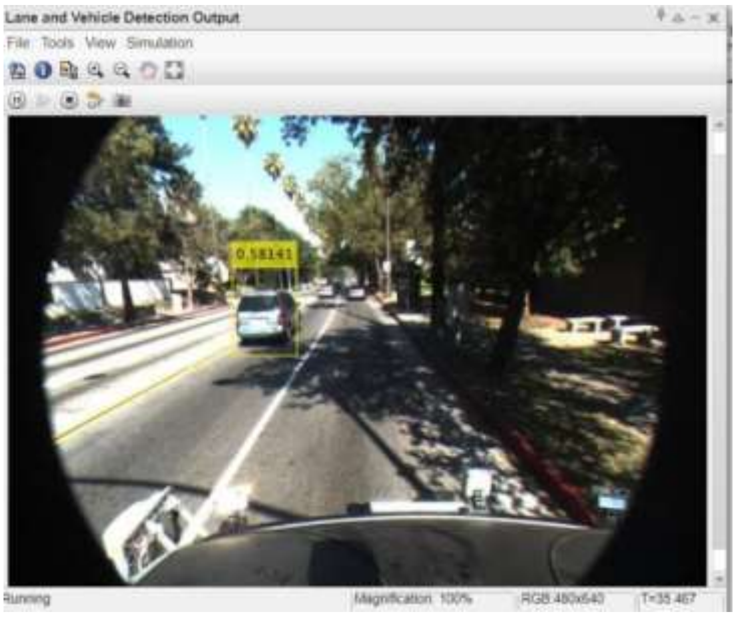

Figure-9 Output after Detecting a Vehicle

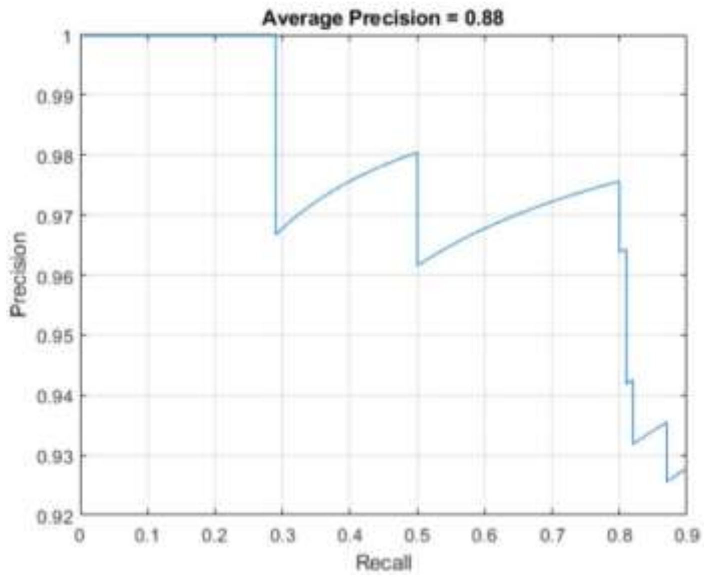

Figure-10 Vehicle Detection with Precision Accuracy

An element of quantification is added to the result by the precise accuracy that is computed and presented along with the identified cars. A statistic called precision assesses the degree of accuracy of the object detecting system provided accurate predictions. It offers information about the degree of certainty the system has in its classifications. Users are better able to evaluate the accuracy of the detections and base their decisions on the system's confidence level when this precise accuracy is displayed.

The output video is set up to loop, with ten seconds of playback between each loop. This loop function lets users observe the system's performance over an extended length of time by ensuring continuous monitoring and real-time assessment of vehicle detection. It is particularly helpful in situations when continuous attention to detail is necessary, like in autonomous cars or surveillance.

An input video with a series of 295 images, or frames, is the source of the output video. The object detection procedure is based on these input frames. The model analyzes each frame in real-time using the YOLOv3 object detection method, which makes it possible to identify and annotate automobiles.

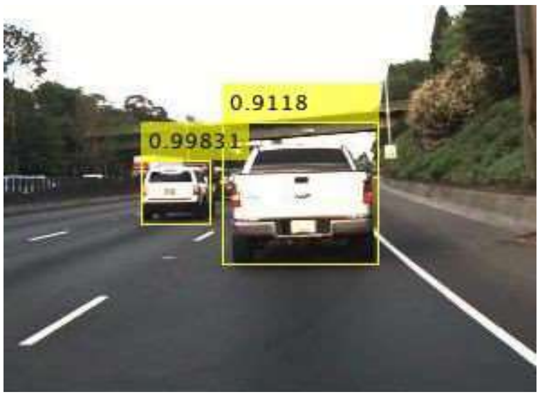

Figure-11 Analysis of Average Precision

VI. Results Of Lane Detection

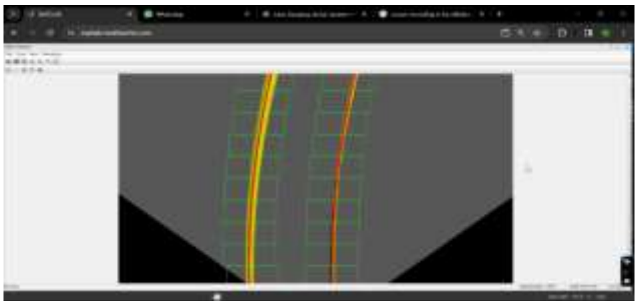

Figure-12 Output of Lane Detection in Bird View

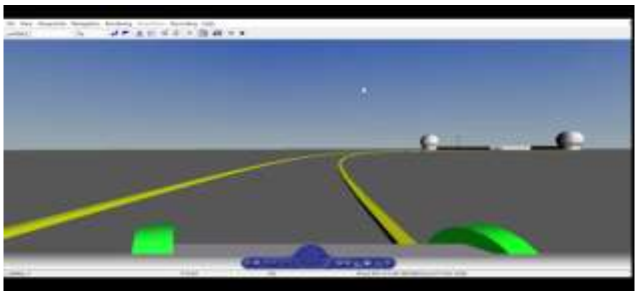

Figure-13 Output of Lane Detection in front view

VII. Future Scope:

1. Increased Accuracy and Speed: Upcoming studies may concentrate on optimizing the YOLOv3-based vehicle detection and lane detection algorithms' accuracy and speed. To improve performance in real-time applications, this could entail developments in network designs, optimization strategies, and training procedures.

2. Robustness to Challenging Conditions: Systems for lane detection and vehicle detection that can withstand difficult environmental factors like bad weather, dim lighting, occlusions, and changing road conditions are becoming more and more necessary. Subsequent advancements could concentrate on enhancing the resilience and dependability of YOLOv3-based algorithms in scenarios like these.

3. Integration with Autonomous Vehicles: To enable safe and dependable autonomous navigation, there is a growing need for precise and effective vehicle recognition and lane detection systems as autonomous vehicle technology advances. Algorithms based on YOLOv3 may be essential to the advancement of autonomous car perception.

4. Multi-Object Tracking: Using YOLOv3-based vehicle detection, future research may investigate methods for multi-object tracking. This would include keeping track of associations between detections over successive frames, anticipating the trajectories of many vehicles, and tracking them over time.

5. Adaptation to Dynamic situations: Future development of YOLOv3-based vehicle recognition and lane detection systems could enable them to adapt to dynamic situations in which lane configurations and vehicle presence are subject to change. Algorithms that can track vehicles in real time and update lane models dynamically would be needed for this.

VIII. Conclusion

To sum up, YOLOv3-based vehicle and lane recognition offers a viable method for boosting road safety, enabling autonomous driving, and advancing transportation infrastructure. With its speed and precision, YOLOv3 is a good option for real-time vehicle and lane marking recognition in a variety of settings.

YOLOv3 uses deep learning approaches to provide reliable and effective vehicle detection, which makes it possible to accurately identify many objects at once. Furthermore, by modifying YOLOv3 for lane detection, important details about the geometry of the road may be extracted, which helps with lane-keeping and navigation.

Even while YOLOv3 offers a strong basis for lane and vehicle identification, there is still room for improvement. Subsequent studies may concentrate on strengthening the algorithm's resistance to difficult circumstances, like bad weather and intricate traffic patterns. Furthermore, by combining YOLOv3 with additional sensor modalities and fusion techniques, perception capacities can be improved, leading to a more thorough comprehension of the world.

All things considered, YOLOv3's use in lane and vehicle detection holds enormous potential for the advancement of transportation technology. We can get closer to creating transportation systems for society that are safer, more effective, and more intelligent by continuously improving and optimizing these algorithms.

IX. References

[1]. Yi-Qi Huang, Jia-Chun Zheng, Shi- Dan Sun , Cheng_x0002_Fu Yang and Jing Liu, "Enhanced YOLOv3 Algorithm and Its Application in Traffic Flow Detections", Appl. Sci. 2020, 10,3079,doi:10.3390/app1009307.

[2]. Muhammad Fachrie,"A Simple Vehicle Counting System Using Deep Learning with YOLOv3 Model", DOI:10.13140/RG.2.2.15026.56001, 2020.

[3]. Lecheng Ouyang, Huali Wang, "Vehicle target identification in complex scenes dependent on YOLOv3 Algorithm", AMIMA 2019.

[4]. Riaz, A., Khan, A. A., & Zafar, M. F. (2020). "Vehicle detection and tracking using deep learning: YOLO vs. faster R-CNN". In 2020 International Conference on Innovative Computing (ICIC) (pp. 1-6). IEEE.

[5]. Lee, S., Kim, H., Kim, J., & Ko, B. C. (2017). "Vehicle detection based on YOLO deep neural network". In 2017 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 147-150). IEEE.

[6]. Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., ... & Adam, H. (2015). "An empirical evaluation of deep learning on highway driving". arXiv preprint arXiv:1504.01716.

[7]. Tang, W., & Tan, T. (2018). "Lane detection and tracking using deep learning based object detection in a driving environment". In 2018 IEEE Intelligent Vehicles Symposium (IV) (pp. 1078-1083). IEEE.

[8]. Goyal, S., & Singhal, N. (2019). "Real-time vehicle detection and counting system using YOLO". In 2019 International Conference on Automation, Computational and Technology Management (ICACTM) (pp. 1-6). IEEE.

[9]. Srivastava, A., Jindal, R., Jain, S., & Saurabh, K. (2020). Vehicle detection and classification using YOLOv3. In 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3) (pp. 1-6). IEEE.

[10]. Redmon and Farhadi A, "YOLOv3: An Incremental Improvement," Preprint, submitted April 8, 2018. [Online]. Available: https://arxiv.org/abs/1804.02767.

[11]. Krizhevsky A, Sutskever I, and Hinton G.E, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, May 2017.

[12]. W. Min, M. Fan, X. Guo, Q. Han, "A new approach to track multiple vehicles with the combination of robust detection and two classifiers," IEEE Trans. Intell. Transp. Syst., vol. 19, pp. 174–186, 2018.

[13]. Elert N.S, 2020. Programming possibilities using MATLAB Simulink embedded coder on the example of data analysis from ahrs module, Journal of Physics Conference Series 1507 082042.

[14]. S. Agarwal, J. O. Terrail, F. Jurie, "Recent advances in object detection in the age of deep convolutional neural networks," arXiv preprint arXiv:1809.03193, 2018.