International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 06 | June 2022 www.irjet.net p-ISSN: 2395-0072

VEHICLE DETECTION, CLASSIFICATION, COUNTING, AND DETECTION OF VEHICLE DIRECTION AND LANE DETECTION

Ganesh Chowdary Manne[1], Suvarna Abhishek[2], G. Venkata Sai Ram[3], T. Sai Harshith[4], J. Pravalika[5]

[1,2,3,4] *Student, Department of Information Technology, SNIST, Hyderabad, Telangana, India.*
[5]*Professor, Department of Information Technology, SNIST, Hyderabad, Telangana, India.*

Abstract-*Safety and security are two of the most significant characteristics of today's age, and 'automation' is also becoming increasingly important. The major goal of this work is to demonstrate how computer vision and deep learning may be used to detect moving vehicles and count them by classifying them appropriately, as well as to address lane detection using computer vision. When detecting a vehicle, computer vision detects the car in the frame and classifies it using the YOLO deep learning algorithm and as well as determining whether the observed vehicle is moving forward or backward from the camera's perspective. While lane detection uses methodology such as frame masking, canny edge detection, and Hough lines transformation. These systems can be used in self-driving automobiles, driver assistance in smart cars, as well as traffic management and planning, parking management systems, Traffic control and other applications.*

Keywords: Lane, Vehicle, Automation, computer vision, Deep Learning.

1. INTRODUCTION

In the prevailing time, the range of vehicles transferring on roads & highways is growing in a speedy manner, with this the want of tracking and controlling of cars is greater crucial. Surveillance cameras may be visible in lots of public places. Several recordings are being saved and archived over time. Vehicle detection and counting have become an increasing number essential inside the dual carriageway control sector. However, because of the one-of-a-kind sizes of cars, their discovery stays a task that without delay changes the accuracy of car records. The dual carriageway place pictured is first demarcated and subdivided right into a subdivision and adjoining place inside the proposed new subdivision; the technique is essential in enhancing car acquisition. Then, the 2 places above had been located at the YOLOv3 community to decide the kind and place of the car. Finally, vehicle trajectories are detected via way of means of the ORB algorithm, which may be used to decide a vehicle's using conduct and decide the range of various cars. Several dual carriageway-primarily totally based surveillance movies are used to confirm the proposed routes. Test effects affirm that the use of the proposed partitioning technique can offer excessive detection accuracy, mainly for small car detection. In addition, the brand-new method defined in this newsletter could be immensely powerful in judging using and counting cars. This mission has a sensible price this is not an unusual place with inside the control and management of dual carriageway scenes. At present, item-primarily totally based car discovery is split into traditional system imaginative and prescient and deep mastering methods. Conventional system imaginative and prescient structures use the motion of a vehicle to split it from a static heritage image. This technique may be divided into 3 categories: a way to use heritage removal, a way to use non-stop comparison of video frames, and a way to use optical flow. Using the video body variant technique, the distinction is calculated in step with the pixel values of or 3 consecutive video frames. In addition, the frontal circuit is separated via way of means of a threshold. Vehicle detection and records in dual carriageway tracking video sequences are extraordinarily essential for dual carriageway site visitors control and management. With the significant use of site visitor's surveillance cameras, a huge library of site visitors-associated video pictures has been created and gathered for the motive of evaluation while considered from an excessive vantage point, it's miles viable to assume an avenue floor this is in addition away. The goal car's length varies dramatically while considering this attitude and the precision with which a small item a way away may be detected the visibility from the street is poor. In the face of complex digital digicam structures, its miles important to deal with the demanding situations indexed above effectively at the back of the scenes. And positioned them into practice.

2. LITERATURE SURVEY

Many people have published many research papers explaining different applications of OpenCV, NumPy, YOLO, and many detecting algorithms. A paper written by Aharon Bar Hillel, Ronen Lerner, Dan Levi & Guy Raz titled: Recent progress in

road and lane detection: a survey [1], performs a survey on various research papers on the topic of road and lane perception over the last 5 years, in this, they have discussed and elaborated all the techniques. The paper titled: A review of lane detection methods based on deep learning written by, PengLiu SongbinLi, and JigangTang [2], discusses various methods and approaches to performing lane detection. The paper Titled: Robust Lane Detection and Tracking in Challenging Scenarios [3], by ZuWhan Kim addresses some of the challenges which are faced in the traditional lane detection process like lane curvatures, worn line-markings, lane changes. The paper written by Abdulhakam.AM. Assidiq; Othman O. Khalifa; Md. Rafiqul Islam; Sheroz Khan: Real-time Lane detection for autonomous vehicles [4], in this a vision-based lane detection approach is presented that can work in real-time and is resistant to illumination changes and shadows. The paper written by Kuo-Yu Chiu; Sheng-Fuu Lin: Lane detection using colour-based segmentation [5], We offer a new strategy based on colour information that can be used in complex situations. In this way, we start by selecting the region of interest to find the colour limit using a mathematical method. Thereafter the boundary will be used to distinguish between the possible boundaries of the road and the road. The paper: A novel system for robust lane detection and tracking [6], by Yifei Wang; Naim Dahnoun; Alin Achim introduces a route detection and tracking system based on a new route to remove the feature and Gaussian Sum Particle (GSPF) filter. In the paper titled: Lane Detection with Moving Vehicles in the Traffic Scenes [7], written by Hsu-Yung Cheng; Bor-Shenn Jeng; Pei-Ting Tseng; Kuo-Chin Fan in this paper first, route lanes are recognized based on color information. Next, in cars of the same color and trajectory, we use size, shape, and movement information to distinguish the actual route markings. Finally, the pixels in the line marker mask are collected to find the line boundary lines. The paper: An Improved YOLOv2 for Vehicle Detection [8], by Jun Sang; Zhongyuan Wu; Pei Guo; Haibo Hu; Hong Xiang; Qian Zhang; and Bin Cai in this paper it is proposed to solve the problems of existing methods like vehicle recognition, low accuracy, and speed new detection model YOLO v2 Vehicle based on YOLOv2. A real-time oriented system for vehicle detection by [9], Massimo Bertozzi; Alberto Broggi; Stefano Castelluccio: this work presents a system for vehicle detection in a monocular video, it is composed of 2 different engines PAPRICA a parallel architecture, and a serial architecture which runs medium level tasks aimed to detect the position of vehicles. In the paper titled: Vehicle Detection and Tracking Techniques: A Concise Review [10], by Raad Ahmed Hadi, Ghazali Sulong, and Loay Edwar George we present an overview of the image processing techniques and analytical tools used to build these aforementioned applications that involve the development of traffic monitoring systems. Paper titled Real-time multiple vehicle detection and tracking from a moving vehicle [11], written by Margrit Betke, Esin Haritaoglu & Larry S. Davis, a real-time visual system has been developed that analyzes color-coded video cameras on the front-wheel-drive vehicle. The system uses a combination of color, edge, and movement information to detect and track road boundaries, road signs, and other vehicles on the road. A paper titled A survey of vision-based vehicle detection and tracking techniques in ITS written by Yuqiang Liu; Bin Tian; Songhang Chen; Fenghua Zhu; Kunfeng Wang [12], this work presents here a complete review of high-quality video processing strategies for car detection and tracking as well as viewing directions for future research. A paper Vehicle Detection and Classification: A Review [13], by V. Keerthi Kiran, Priyadarsan Parida & Sonali Dash presents a detailed review of vehicle acquisitions and classification methods and discusses various vehicle acquisitions in adverse weather conditions. It also discusses the data sets used to evaluate proposed strategies in various disciplines.

3. METHODOLOGY

3.1. Vehicle Detection, Classification, and Counting Methodology

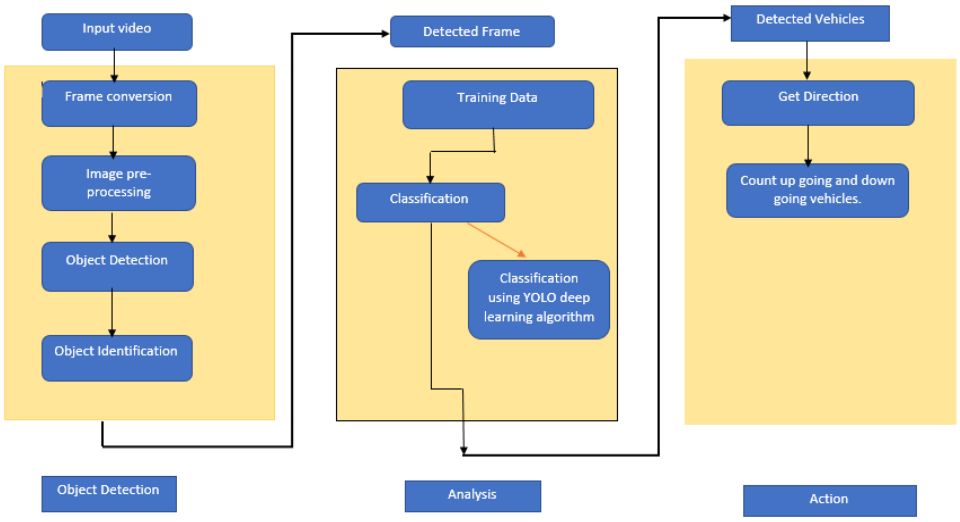

Fig 1: Flow chart for Lane Detection, Vehicle Detection and Classification.

We perform vehicle detection and classification using the OpenCV and python modules. We use the YOLOv3 model along with OpenCV-python. OpenCV is a library of python bindings designed to solve computer vision problems. YOLO stands for the You Only Look Once. It is useful for detecting real-time objects using algorithms. YOLO algorithms work by dividing the image into N grids and each grid has an equal dimensional region of S x S. Each of N grids handles the detection and localization of the objects it contains.

In this project, we will detect and classify the cars moving on the road and also counts the number of cars passing through the road. We need to create two programs for this project. The first one will track the cars using OpenCV. It keeps track of each and every detected vehicle on the road and another one will be the main detection program. The prerequisites required for working with these projects are python-3.0, OpenCV-4.4.0, Numpy, and YOLOV3 pre-trained model weights and config Files.

While performing the vehicle counting. In this process, the First step is, we need to import the necessary packages and initialize the network. In this step, we import the tracker package which is useful for tracking an object using the Euclidean distance concept and also, we need to initialize the line positions that will be used to count vehicles. YOLOv3 is trained on the coco dataset, so we read files that contain all class names and store the names in a list. The coco dataset consists of 80 different classes. Configure the network using cv2.dnn.readNetFromDarknet() function. If you using GPU set the DNN backend as CUDA. We can use the randint function from the random module where we can generate a random color for each class in our data set. With the help of these of these colors, we can draw the rectangles around the objects.

In the second step, we need to read the frame from the video file. Initially, we need to read the video through the video capture object. By performing the read method, we can read each frame from the capture object. By the helpreshapinghape we reduced our frame by 50 percent. The line function is useful for drawing crossing lines in a frame. Then we use the imshow() function to show the output image. In the third step, we preprocess the frames and run the detection. In this we use forward is used to feed the image as input and return an output. Then we perform the postprocess() function to post-process the output. In the fourth step, we perform the preprocessing of the output data.

The network forward has 3 outputs. First, we define an empty class where we store all detected classes in a frame. We use two for loops we iterate through each vector and collect the confidence score and classId index. If the confidence score is greater than our defined confThresold. Then we collect information about the class and store the box coordinate points, class-id, and confidence score in three separate lists. YOLO sometimes provides multiple bounding boxes for a single object, so we reduce thenumber of detection boxes and have taken the best detection box for each class. We use the

NmnBoxes() method to reduce the number of boxes and take only the best detection box for the class. We use cv2.rectangle() we can draw a bounding box around the detected object.

3.1.1. Tracking and counting of vehicles

In the fifth step, Track and count all vehicles on the road, in this stage after getting all detections, we keep track of those objects using a tracker object. The tracker. update() function keeps track of very detected objects and updates the positions of the object. Count_vehicle is a custom function that counts the number of vehicles that crossed the road. Then we need to create the two empty lists to store class id's that enter the crossing line. UP_list and down_list is for counting those vehicle classes in the up route and down route. Then find_center function returns the center point of a rectangle box. In this part, we keep track of each vehicle's position and their corresponding Ids. First, we check if the object is between the up_crossing line and middle crossing line, then id of the object temporality is stored in the up list for up route vehicle counting. In the down route vehicles, we check whether the object has crossed the down line or not. If the object crossed the down line, then the id of the object is counted as an uproute, and we add 1 with the particular type of class counter. We use the circle() method to draw a circle in the frame. Finally get the counts to show the vehicle counting on the frame in real-time.

3.1.2. Architecture

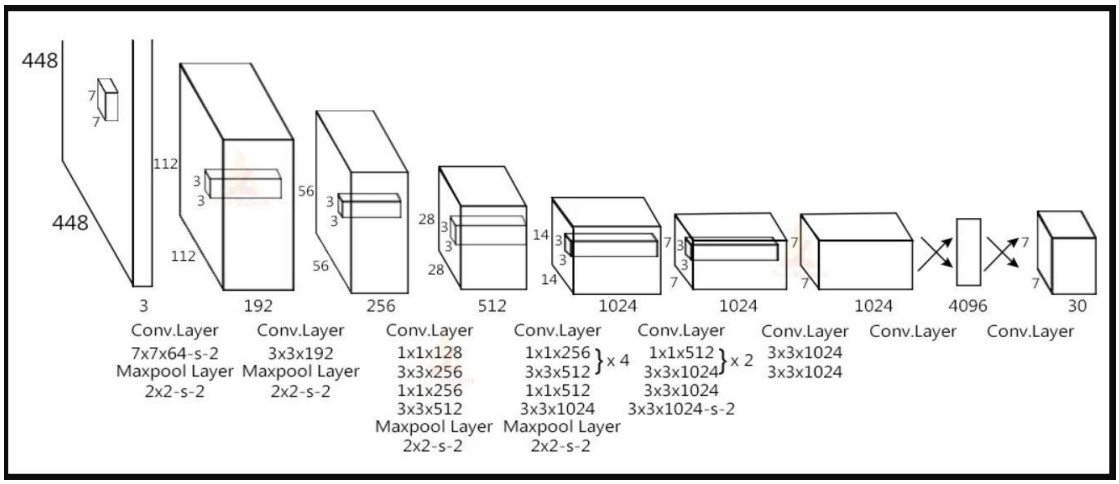

Fig -2: YOLO Architecture

3.2. Lane Detection Methodology

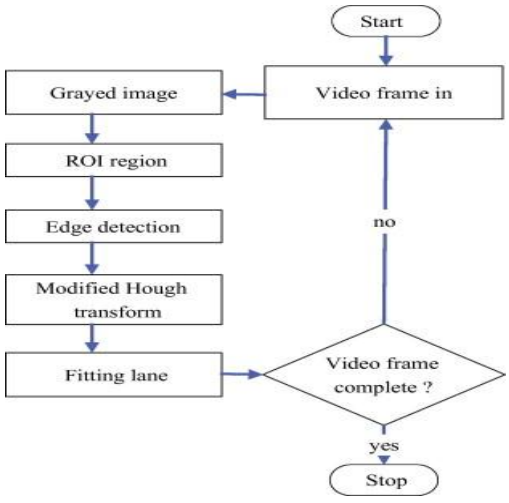

Fig -3: Methodology for Lane Detection

3.2.1. Canny Edge Detection:

Canny Edge Detection is a widely used algorithm in the OpenCV module, as the name suggests this is a method used for detecting edges in any given image. This was developed by John F. Canny in the year 1986. This algorithm is a multi-staged process, below is the stepwise working of the algorithm.

Noise Reduction: Edges are always susceptible to noise, so the first step in the algorithm is to remove this noise by using a 5 * 5 Gaussian filter. This also smoothens the image by reducing the noise

Intensity Gradient Calculation: After the image is smoothened and noise is removed then it is filtered using the Sobel kernel both vertically and horizontally, this helps in deriving the first derivative in the horizontal direction (G_x) and first derivative in the vertical direction (G_y). After this we can calculate the edge gradient and direction of each pixel using the formulae given below:

$$Edge\_Gradient(G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle(\theta) = \arctan\left(\frac{G_y}{G_x}\right)$$

The gradient direction is always perpendicular to the edges, and it is rounded to one of the four angles (0, 45, 90, 180).

Non-Maximum Suppression: After finding the gradient values next, we need to perform a quick scan on the whole image looking for pixels that are unwanted means which are not constituted as edge and remove them. The result you get is a binary image with "thin edges".

Hysteresis Thresholding: The last step in the process is hysteresis thresholding for this process to complete canny edge detection using two threshold values (i.e., upper threshold and lower threshold) those can also be said as MinVal and MaxVal.

The intensity gradient (G) of each pixel is checked with both the threshold values in order to distinguish edge pixels and non-edge pixels. For each pixel (p):

a. If G > MaxVal, then it is considered as an edge pixel.
b. If G < MinVal, then it is considered a non-edge pixel and is discarded.
c. If G > MinVal and G <MaxVal, then this is considered as classified edges and if p is a pixel that is connected to a 'sure-edge' pixel the p is accepted, else p is rejected and discarded.

3.2.2. Hough Lines Transform:

Hough lines transform is a transform used to detect straight lines in an image or video, for this transform to be applied on any image first it is suggested to run an edge-detection pre-process.

Lines in an image space can be expressed with 2 variables:

Cartesian coordinate system: (x, y)

Polar coordinate system: (r, θ)

We use the polar coordinate system to express lines in Hough lines transform, below is the equation we use for the equation of a line.

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right)$$

Arranging the above formulae gives the below one,

$r = x\cos\theta + y\sin\theta$

For each point, (x_0, y_0) in general gives,

$$r_\theta = x_0 . \cos\theta + y_0 . \sin\theta$$

This means that the point (r_0,θ) represents every line passing through (x_0, y_0) .

If for a given point (x_0, y_0) a family of lines plotted, then we obtain a sinusoid., in the plane.

We consider only the points such that r > 0 and $0 < \theta < 2\pi$.

We continue the process for some other points in an image, if any intersection happens in the Θ- r plane, that indicates both the points belong to same line.

These intersections in the plane help us to detect straight lines, i.e., more the intersections more the points of a same line, to consider the detected object to be as a lion a threshold value can be defines which checks for minimum number of intersecting curves in the plane to assign the object as a line. The Hough Line Transform keeps a track of these intersections of curves on the plane for every point in the image when the value increases than threshold then it declares it as a line with (θ,r_0) as intersection point.

3.2.3. WORKING FLOW OF LANE DETECTION

we need to read the video from the camera input, next step is to perform canny edge detection on the read video on every pixel, this can be done by applying gray scaling to images first this turns the image into gray internally, and then the Gaussian smoothing, and blur methods are applied on the grayscale image then the actual canny edge detection is done, and the edges of the whole image are detected. The next step is to identify the region of interest (ROI) according to the requirement of the project, next we need to apply Hough transforms in the ROI after that we have to find the average slope and extrapolate the lane lines in the Hough transforms to apply the image on the video to detect the lanes in the input video

4. RESULTS AND CONCLUSION

4.1 Vehicle Detection Result

As shone in Fig. 4 the vehicles are detected which is represented by the rectangular bounding box. The detected vehicle is classified and the percentage of accuracy is displayed on top of the bounding box, the 2 red lines represent the up and down lines where a vehicle crossing the line is counted as up going or down going vehicle. The pink line is the threshold line. Used to detect the up going and down going vehicles.

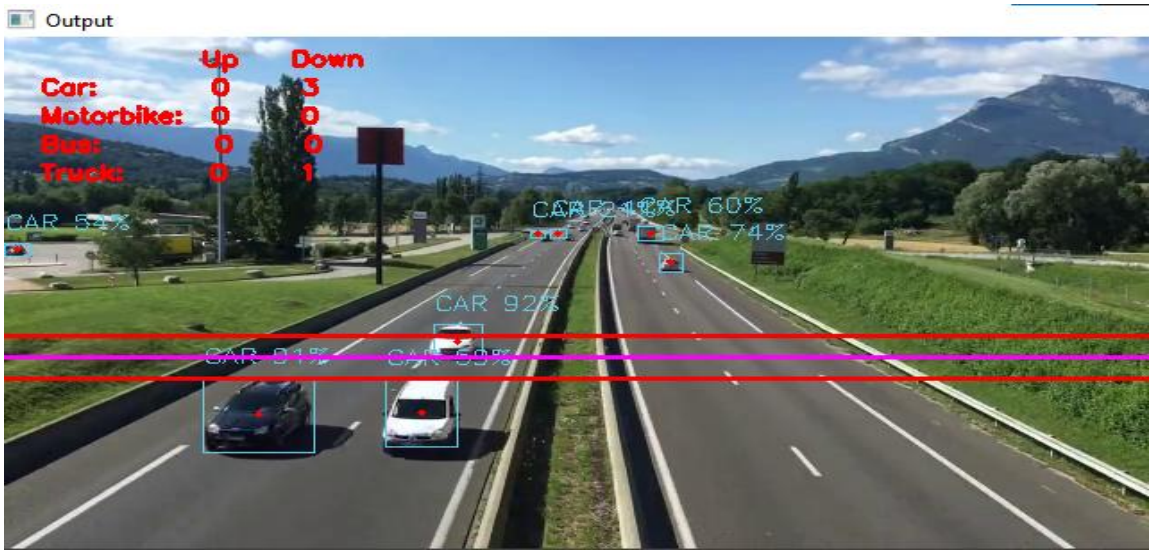

Fig -4: Vehicle detection and count output

4.2. Lane Detection Result

The detected line is represented in reddish orange color in the Fig. 5

Fig -5: Lane detection output

REFERENCES

[1] Aharon Bar Hillel, Ronen Lerner, Dan Levi & Guy Raz. "Recent progress in road and lane detection: a survey", Science Direct.

[2] Jigang Tang, Songbin Li, PengLiu. "A review of lane detection methods based on deep learning.", Elsevier.

[3] ZuWhan Kim. "Robust Lane Detection and Tracking in Challenging Scenarios.", IEEE.

[4] Abdulhakam.AM. Assidiq; Othman O. Khalifa; Md. Rafiqul Islam; Sheroz Khan "Real time lane detection for autonomous vehicles.", IEEE.

[5] Kuo-Yu Chiu; Sheng-Fuu Lin "Lane detection using color-based segmentation.", IEEE.

[6] YifeiWang, NaimDahnoun, AlinAchim "A novel system for robust lane detection and tracking.", Elsevier.

[7] Hsu-Yung Cheng; Bor-Shenn Jeng; Pei-Ting Tseng; Kuo-Chin Fan "Lane Detection with Moving Vehicles in the Traffic Scenes", IEEE.

[8] Jun Sang; Zhongyuan Wu; Pei Guo; Haibo Hu; Hong Xiang; Qian Zhang; and Bin Cai "An Improved YOLOv2 for Vehicle Detection.", MDPI.

[9] Massimo Bertozzi, Alberto Brogg, Stefano Castelluccio "A real-time oriented system for vehicle detection.", Science Direct.

[10] Raad Ahmed Hadi, Ghazali Sulong, Loay Edwar George "Vehicle Detection and Tracking Techniques: A Concise Review.", ARXIV.

[11] Margrit Betke, Esin Haritaoglu & Larry S. Davis "Real-time multiple vehicle detection and tracking from a moving vehicle", Springer.

[12] Yuqiang Liu, Bin Tian, Songhang Chen, Fenghua Zhu, Kunfeng Wang "A survey of vision-based vehicle detection and tracking techniques in ITS", IEEE

[13] V. Keerthi Kiran, Priyadarsan Parida & Sonali Dash "Vehicle Detection and Classification: A Review", Springer.