Real-time Traffic Signs and Lane Detection

Harsh Kumar Chaudhary
Department of Electrical Engineering
Fr. C. Rodrigues Institute of Technology
Vashi, Navi Mumbai
harshkc2000@gmail.com

Khaliluddin R. Sarkazi
Department of Electrical Engineering
Fr. C. Rodrigues Institute of Technology
Vashi, Navi Mumbai
khaliluddin.sarkazi@gmail.com

***Abstract*—Automation of vehicles is a fast growing field and detection of driving lanes, traffic signs and signals are important parts of it. This paper introduces a methodology used for detection and tracking of traffic signs and signals along with lane detection. A convolutional neural network based model is used for traffic signs and signals detection. For driving lane detection, a simple computer vision algorithm is implemented. It was shown that real-time detection can be achieved, even on HD images with average Frames Per Second (FPS) of around 35.**

***Keywords*—lane detection; neural networks; computer vision; object detection; hough transform; object tracking; autonomous vehicles.**

I. Introduction

The need for independent cars has extensively multiplied inside the beyond few years because of growing site visitors tiers and progressively busier roads across the globe. In a few decades, these systems will become more complicated to provide complete autonomy of the vehicle. Mainly, there are three important elements in the development of such autonomous systems, which are lane, signs/signals and obstacle detection. Modern-day vehicles are typically ready with some sensors, and both front and rear video cameras. Their original reason became parking and using assistance, both of which become completed by visual or sound alarms. Front cameras may be used for warning of close by pedestrians, bicycles and different automobiles in addition to for detection of different types of limitations. With modern advances in technology and the auto industry it is reasonable to assume that vehicle computer systems can be extra effective and able to provide greater assistance, also regarding driving lanes and signs/signals detection. For such advanced systems, it is necessary that the car computers are able to process data at a high speed for sufficing the real time needs. A device capable of performing this kind of task might be very valuable. It may be used as an assistant for drivers, alerting them approximately the presence of some particular sign (e.g., a keep right sign) or a high-risk situation (e.g., riding at a better velocity than the maximum velocity allowed).

Current breakthroughs inside the subject of machine learning, allowed the implementation of efficient object recognition algorithms. For traffic sign detection and traffic signal detection, an important issue is proper management of datasets for less false negatives and false positives, therefore ensuring the safety of participants in traffic. Implementing a Convolutional Neural Network (CNN) based model for such a task was found to be the most effective. Convolutional Neural Networks (CNN's) are the most effective methods, which provide the quality result in object detection and recognition, therefore exquisite overall performance is finished inside the task of Traffic Signs and Traffic Signals detection and recognition. Therefore, we employed the novel YOLOv4-tiny architecture[1] which has demonstrated to be an accurate and fast method for real-time object detection. In our case, we trained a YOLOv4 tiny model for obtaining much higher inference speeds at a cost of lower accuracy, so that real-time speeds can be achieved without much computational costs.

For lane detection and tracking, there are particularly two varieties of vision-based techniques namely, feature-based and model-based approach. Here, we used a feature-based approach for lane detection analysis. In most of the cases the big challenges for the lane detection with tracking are the decreased visibility due to terrible weather conditions, disconnectivity of traces, loss of clarity of lane markings, shadows, illumination and mild reflection. A model based approach would be preferred for overcoming these challenges but implementing it along with the CNN model would add on to the computational costs significantly. For lane detection, roads can be categorized into two groups on the basis of their marking:

(i) Structured roads: Roads with lane markings

(ii) Non-structured roads: Roads without lane markings

Lane detection on structured roads is comparatively easier than non-structured roads. In our case, we have focused mostly on structured roads and for that a simple and effective computer vision algorithm using Canny Edge Detector and Hough transform is implemented.

To provide a common platform to both OpenCV (Computer vision algorithm library) and YOLO, we used Tensorflow by

Google. Tensorflow is a high level library that allows researchers and developers to easily build state-of-the-art Machine Learning powered applications.

This paper is prepared as follows. In Section II, the info of network architecture for real-time item detection is given as well as the database that is used for training. Section III offers information about the technique and experimental outcomes of the lane detection algorithm. Section IV offers end feedback.

II. Traffic signs and signals detection

A. Object Detection Network

YOLO (You Only Look Once) is an object detection approach with deep neural networks as a backbone, which is designed for instant and accurate object detection. It is designed to simultaneously predict class probabilities and multiple bounding boxes for those boxes. YOLO takes the entire image into consideration while training, so it is able to consider contextual information about the objects. YOLO is refreshingly simple. This unified model has several benefits over traditional methods of object detection.

YOLO v4, with the reduced density of the convolutional layers, is another version referred to as YOLO v4-tiny. It was proposed by Alexey Bochkovskiy [1]. Therefore, inference speeds are significantly increased (about 540% faster than YOLOv4), but with the cost of reduced detection accuracy. YOLO takes an entire photograph and divides it into square grids, and then every grid predicts some bounding boxes. Now, image classification and localization are applied on each grid. It then predicts the bounding boxes and their corresponding class probabilities for objects (if there are any). Finally, Intersection over Union (IoU) and Non-Max Suppression (NMS) techniques are applied on the predicted boxes to obtain a single prediction per object.

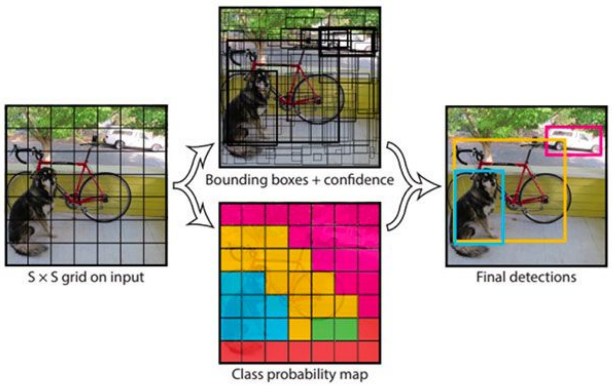

Fig. 1. Working process of YOLO model

YOLOv4 tiny is extremely fast and has shown significant object detection accuracy improvements over it's previous versions. Table I. shows that previous versions of YOLO tiny have a lower value of mAP (mean Average Precision) on the COCO dataset compared to YOLOv4 tiny.

TABLE I. Performance comparison of YOLO tiny versions

Detector	*Number of layers*	*FLOPS*	*FPS*	*mAP value*	*Used set of data*
YOLO v1-tiny	9	-	155	52.8	VOC-data
YOLO v2-tiny	16	5.42	244	23.6	COCO-data
YOLO v3-tiny	22	5.57	220	33.2	COCO-data
YOLO v4-tiny	36	6.9	371	40.2	COCO-data

YOLOv4 tiny consists of a total of 36 layers. The architecture for the same is shown in Fig. 2.

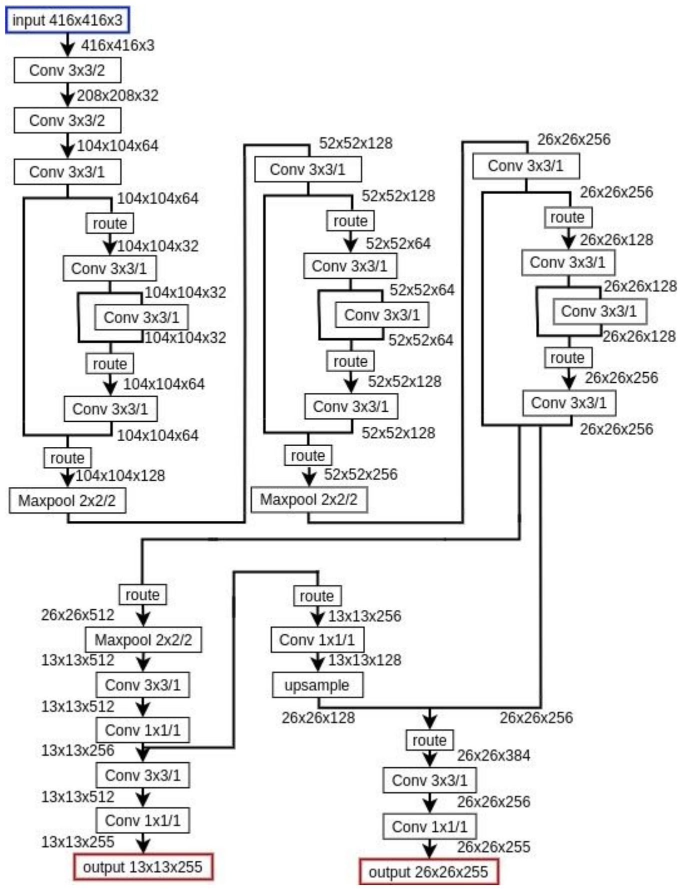

Fig. 2. The YOLOv4 tiny network architecture

B. Traffic signs and signals database

For training the YOLOv4 tiny model, two detection datasets from two different sources were combined into a single database. For traffic signs, we used The German Traffic Sign Detection Benchmark (GTSDB) database [2]. It consists of a total of 900 images which is splitted into 600 training images and 300 testing/evaluation images. Each image in this dataset has a resolution of 1360x800 pixels and the colour format is RGB. The traffic signs in this dataset are grouped into 4 categories: Prohibitory, Danger, Mandatory, Other. Detailed information about each category is given below:

(i) The Prohibitory category consists of speed limit, no overtaking, no traffic both ways, no trucks signs.

(ii) The Danger category consists of priority at the next intersection, school crossing, danger, uneven road, bend left, cycles crossing, bend right, snow, bend, slippery road, construction, road narrows, traffic signal, pedestrian crossing, animals signs.

(iii) The Mandatory category consists of go right or straight, go left, go right, go straight, go left or straight, roundabout signs, keep left, keep right.

(iv) The Other category consists of give way, restriction ends , no entry signs, priority road, stop.

For traffic lights detection, we used The Tsinghua–Tencent traffic light (TTTL) dataset. The TTTL dataset contains over 16,000 high definition images covering numerous driving scenes. The dataset was splitted into 85% training images and 15% testing/evaluation images. It consists of a total of 6 categories: Red, Red left turn, Green, Red pedestrian, Green forward and Other.

TABLE II. Number of instances for each major class in Tsinghua–Tencent traffic light dataset

Class	***Instance number***
Red	4558
Green	3873
Red left turn	2748
Green forward	908
Red pedestrian	1549
Other	2539

For our purpose, we combined both the datasets into a single one and re-organized the categories into 6 final groups: Prohibitory, Mandatory, Danger, Green, Red and Other. The annotations for both the datasets were converted into the YOLO annotation format.

C. *Methodology*

For training of YOLOv4-tiny, the publicly accessible implementation based totally on the Darknet framework was used[6]. There-trained weights based on the ImageNet database were used to initialize the model. Also,the training sets were used for calculating the anchors during the training process. Input image size was set to 416x416 pixels. Batch size was set to 64 with a learning rate of 0.00261. A total of 14000 iterations were done to train the model. Training was done completely on a single Nvidia Tesla K80 GPU on Google Colab which enabled the use of multiple CUDA cores for faster training of the model.

D. *Results*

After training for 14000 iterations over 4.5 hours, we determined the mean Average Precision (mAP @ 50) for the measure of accuracy. We also determined recall, precision, F1- score and averaged intersection of union (IoU) in Table III. Fig. 3. shows the loss chart generated during the training process.

TABLE III. Results of YOLOv4 tiny model

average	***YOLOv4 tiny***
Precision	0.56
Recall	0.62
F1-score	0.59
BFLOPS	6.797
IoU	41.69
mAP^{50}	55.33

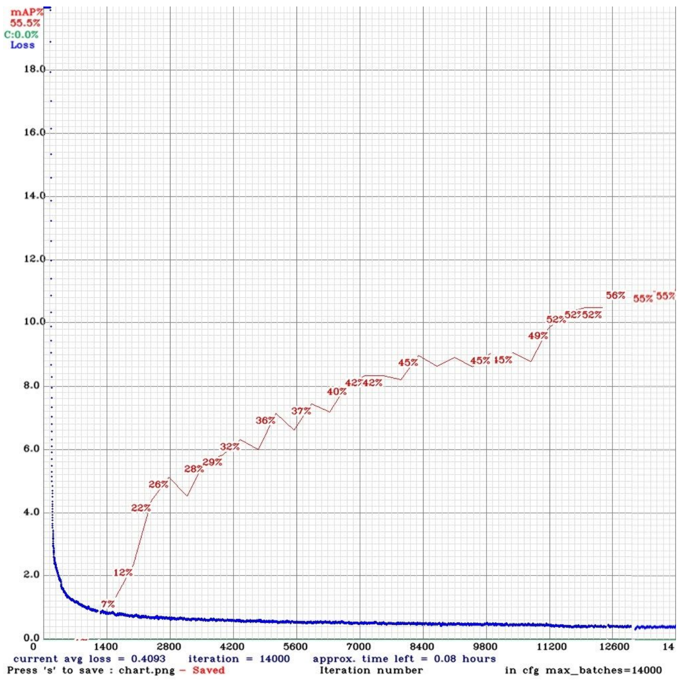

Fig. 3. mAP and Loss chart

Here, as expected the accuracy isn't that great but considering the small and lightweight structure of the YOLOv4 tiny model, it can be considered as a good achievement as it provides really high inference speeds i.e. it takes around 0.01584 seconds to perform detection on a single image. Such speeds easily suffices our real-time detection needs.

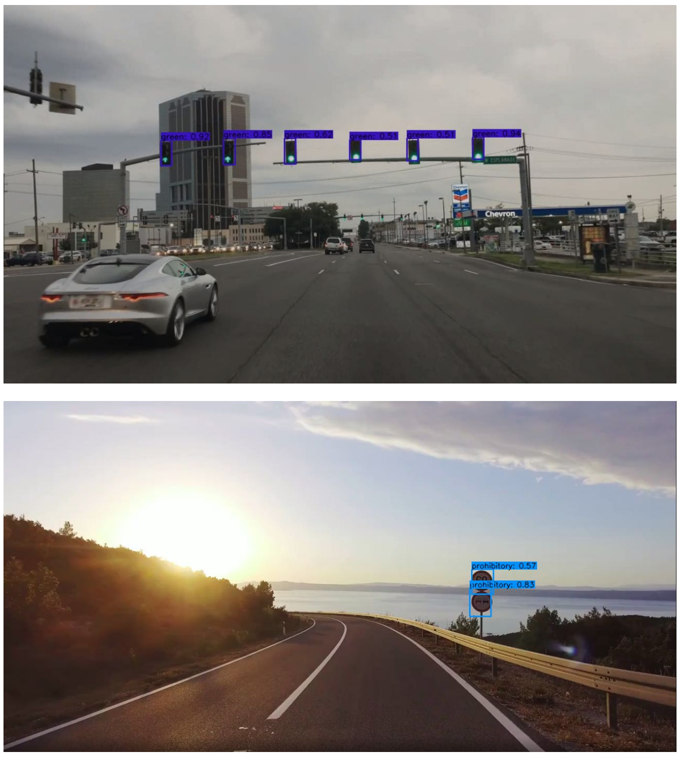

Fig. 4. Examples of YOLOv4 tiny detection and recognition.

III. Driving lane detection

The other part of our work was to design a simple yet reliable algorithm to identify the lane markings on the road in real time.For the same we used OpenCV which is the short form of Open Source Computer Vision Library which is an open source platform containing libraries for various computer vision and machine learning techniques . The video captured by a camera can be broken down into individual frames of images and various computer vision techniques can be used on those images.

To get precision in detecting lanes, various algorithms were applied in order to remove unwanted noises and limit our area of interest to the required area which in our case were the lanes.

The flowchart in Fig.5. provides the series of algorithms we applied on the image to obtain results

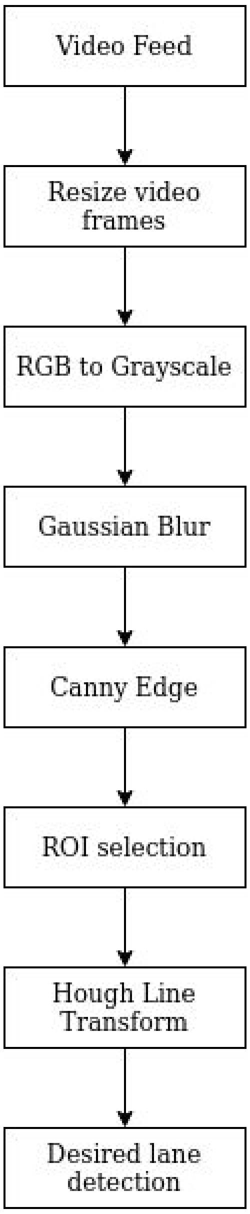

Fig. 5. Lane detection algorithm flowchart

A. RGB to Grayscale

A grayscale image basically has different shades of the color grey. The reason for converting images in grayscale is that we need to provide less information per pixel for the same image in RGB. Thus making computations easier and faster.Ther gray image has equal intensities of the 3 fundamental colors hence we need to provide only one value instead of three.

The intensity of ‘gray’ in an image is specified in a 8-bit integer which gives us 256 different shades of gray with extremes being white and black.

Grayscale images are hence commonly used in various image capturing hardware equipment as they can only support 8-bit images, at the same time the grayscale image is completely efficient in image processing applications and significantly reduces the processing load.

Fig 6(a) shows the conversion of test image from RGB space to Grayscale

B. Gaussian Blur

The Gaussian blur is an image processing technique that uses a Gaussian function (which also expresses the normal distribution in statistics) for calculating the transformation to apply to each pixel in the image to blur the image.

We use Gaussian blur in order to reduce noise from the image in different lighting conditions.It acts as a low pass filtering out the high frequency noises which can cause disturbances.

The formula for Gaussian function in two dimensions is:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where x is the distance on the x axis ie.horizontally from the origin, y is the distance on the y axis ie. vertically from the origin, and σ is the standard deviation in Gaussian distribution. When applied in two dimensions, this formula produces a surface whose contours are concentric circles with a Gaussian distribution from the center point.

Fig 6(b) shows the application of Gaussian Blur on the Grayscale image.

C. Canny Edge Detection

To determine the edges of the line, the sharp contrast between the colors of the lane wrt the surrounding roadways used and canny edge detector algorithm was implemented.After converting the image into grayscale canny edge detector is used to extract the edges . A threshold (upper limit) value is defined, and the edges shorter than a threshold value are removed, and the remaining edges are then combined to form line segments.

The Process of Canny edge detection algorithm can be broken down to 5 fundamental steps as follows:

1. Apply Gaussian filter in order to remove high frequency noises which can cause discrepancies.
2. Finding the intensity gradients of the image
3. Non-maximum suppression technique is applied to get rid of false response to edge detection
4. Applying double threshold to determine points that can be potential edges.
5. By the means of hysteresis the weak points which are not connected and not a part of solid edges are removed

Fig 6(c) shows the application Canny edge detection algorithm on the blurred image

D. Region of Interest Selection

The complete test frame under consideration contains numerous lines which can be detected by the algorithm causing disruption, hence before performing the lane detection algorithm we need to first separate a region of interest. We specify four coordinates making a trapezoidal region of interest while masking the rest of the image black.

The lanes are hence present within the area of interest.Each test frame is priorly reshaped to a standard value in order to avoid any discrepancies.

E. Hough Line Transform

After applying the various above stated algorithms to our video frames we finally have to detect the lanes in the image. The Hough transform algorithm is used for the same.. The biggest advantage of the Hough transform algorithm is that it is limitedly affected by noise in an image and uneven lighting conditions. The Hough transform algorithm is applied to pixels in each frame of the video. The algorithm extracts the coordinates of the line in parametric form and is stored in a 2-D array $C(\sigma, \theta)$. Where σ can be represented as:-

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

Where σ is the distance between the detected line and origin while θ is angle from the x-axis in counter-clock wise direction, and x y is the coordinate value of a pixel. The range of θ is kept between 0 and 90 . The value associated with accumulator cell array is determined by θ , and the resulting σ are incremented by 1, whenever $\sigma = x\cos\theta + y\sin\theta$ is computed for θ . Once the desired lines are determined by setting up proper threshold and σ values the values are mapped back onto the original image to display the lines on them.

Fig 6(d) show the Green line detecting the lanes on the road using hough line transform.

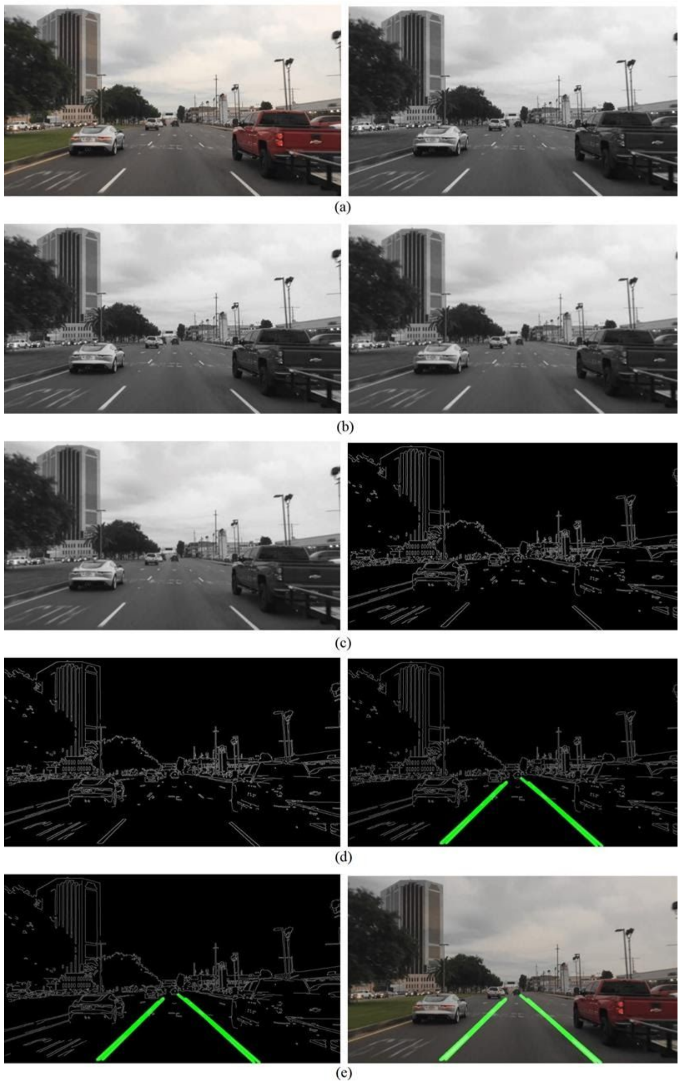

Fig. 6. Applying image processing algos on the test image.

IV. Scope of improvement

For traffic signs and signals detection, we have used YOLOv4-tiny as we had limited computational power. If a decent GPU is available, then the YOLOv4 model can be used for significantly higher detection accuracy as compared to YOLOv4-tiny.

As for the lane detection, discrepancies in lanes can cause a disruption in the detection of the lanes , extreme dark conditions can also cause a failure in the detection. To improve the accuracy further we can use machine learning to train a model on various datasets in a variety of lighting conditions to provide a robust and reliable lane-detection solution.

V. Conclusion

In this study, computer vision and deep learning techniques were used for traffic signs, signals and lane detection and tracking purposes which can be implemented in driving assistance systems in autonomous vehicles.We proposed an effective algorithm for detection and tracking of the lane. The proposed algorithm is straightforward which is mainly based on a series of algorithms the main ones being the canny edge and hough transform. For traffic signs and signals detection, we trained a robust YOLOv4 tiny model to detect multiple classes quickly and as accurately as possible. It enables this system to work effectively in real-time using deep learning techniques. On an average computation system consisting of a decent GPU, it can provide upto 30 FPS which fulfills the need of a fast inference system for real-time application.

In a nutshell the proposed algorithm is lightweight and robust with low computation requirements hence can be easily implemented as compared to other conventional techniques.Further, the algorithm was tested on various videos and successful results were obtained.

References

[1] Alexey, Joseph Redmon, Stefano Sinigardi, cyy, Tino Hager, Vinjn Zhang, … 7FM. (2020, May 15). AlexeyAB/darknet: YOLOv4 pre-release (Version darknet_yolo_v4_pre). Zenodo. http://doi.org/10.5281/zenodo.3829035

[2] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," The 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, 2013, pp. 1-8, doi: 10.1109/IJCNN.2013.6706807.

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, J. and A. Zisserman, "The PASCAL Visual Object Classes (VOC) Challenge", International Journal of Computer Vision, 88(2), 303- 338, 2010

[4] P. Adarsh, P. Rathi and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2020, pp. 687-694, doi: 10.1109/ICACCS48705.2020.9074315.

[5] Lu, Y., Lu, J., Zhang, S. et al. Traffic signal detection and classification in street views using an attention model. Comp. Visual Media 4, 253–266 (2018). https://doi.org/10.1007/s41095-018-0116-x

[6] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779–788, 2016.

[7] H. Bilal, B. Yin, J. Khan, L. Wang, J. Zhang and A. Kumar, "Real-Time Lane Detection and Tracking for Advanced Driver Assistance Systems," 2019 Chinese Control Conference (CCC), Guangzhou, China, 2019, pp. 6772-6777, doi: 10.23919/ChiCC.2019.8866334.

[8] A. de la Escalera, L. E. Moreno, M. A. Salichs and J. M. Armingol, "Road traffic sign detection and classification," in IEEE Transactions on Industrial Electronics, vol. 44, no. 6, pp. 848-859, Dec. 1997, doi: 10.1109/41.649946.

[9] A. Avramović, D. Tabernik and D. Skočaj, "Real-time Large Scale Traffic Sign Detection," 2018 14th Symposium on Neural Networks and Applications (NEUREL), Belgrade, 2018, pp. 1-4, doi: 10.1109/NEUREL.2018.8587013.

[10] B. Novak, V. Ilić and B. Pavković, "YOLOv3 Algorithm with additional convolutional neural network trained for traffic sign recognition," 2020 Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2020, pp. 165-168, doi: 10.1109/ZINC50678.2020.9161446.

[11] A. A. Assidiq, O. O. Khalifa, M. R. Islam and S. Khan, "Real time lane detection for autonomous vehicles," 2008 International Conference on Computer and Communication Engineering, Kuala Lumpur, 2008, pp. 82-88, doi: 10.1109/ICCCE.2008.4580573.