BOHR International Journal of Computational Intelligence and Communication Network
2023, Vol. 2, No. 1, pp. 1–6
https://doi.org/10.54646/bijcicn.009
www.bohrpub.com

Detection and Tracking of Vehicles Using License Plate Recognition

Navjot Kaur* **and Gurjeet Singh**

Department of Electronics, Amritsar College of Engineering and Technology, Amritsar, Punjab, India
**Corresponding author: navjotkaur980310@gmail.com*

Abstract. The density of vehicles on roads has increased manifold over the last few decades. A seamless system to detect and track a particular vehicle can solve many problems, like traffic congestion, etc. This paper proposes the use of real-time data taken from closed-circuit televisions to detect and track the movements of vehicles. The feature extraction and classification are completely done by the process of faster RCNN (regional convolutional neural networks). The core work is based on region proposal networks. CNN furthers the generation of features; classification is done separately, but it is aided by RCNN, which includes deep learning as well as training the neural network. This is used along with the Kanade Lukas Tomasi algorithm to tack the desired features extracted. This method is simulated on MATLAB software.

Keywords: Bounding boxes, CNN (convolution neural network), detection, faster RCNN (regional convolution neural network), KLT(Kanade Lukas Tomasi).

INTRODUCTION

With the seamless services provided by vehicles, especially private modes of transportation, there has been a rise in the number of people mobilizing themselves through private transportation. This has led to an increase in traffic congestion on Indian roads. Construction of new roads is looked at as a solution, but it is an exhaustive approach. People usually encroach on the area around roads; for example, peddlers, beggars, shopkeepers, etc.

A road cannot be widened infinitely in a finite area. Even in the slight area, folks like to park their automobiles further restricting it. Demand and infrastructure structure seems to mismatch. More edifices for ease of movement cannot be the sole solution to myriad issues [4].

The people's behavior becomes apathetic during the parking of their vehicles. This phenomenon is seen a lot in our daily lives. If a person gets just a little bit of space, they will park their vehicle even if it leads to blocking the traffic on the road.

This also implies people are forced to park their conveyance in shady places too. Obviously, this will encourage the theft of automobiles, especially cars. Reports of missing vehicles have become a frequent phenomenon, so much so that such happenings do not shock society that much. But these things can be sorted by various algorithms, like recurrent neural networks containing a memory component that can be used to keep track of object movement across multiple frames of video [9].

Another important factor to be considered here is the crime that may be committed by using these stolen vehicles. For instance, with increased radicalization, there have been many cases of bomb blasts, and the improvised explosive devices (IEDs) used in these events were usually installed in stolen, abandoned vehicles. This entire situation can lead to an innocent person charged guilty of arson. Also, by the time a person finds their lost vehicle, it may already have been damaged.

Fundamentally, object detection is aimed at finding the objects of interest, which revolves around classification and localizing the identified interested regions. Labels are put on them with rectangular bounding boxes to show the scores of the objects present in the regions of interest [20]. Detection of vehicles has a really broad scope as they can be spotted on the basis of various features that they attribute.

It can be based on texture, geometry, movement, speed, and so on. Thus, the feature classification of these can vary accordingly. With the advent of AlexNet and many other deep learning approaches, a lot of algorithms can be used for this purpose [16].

This paper is focused on solving the problems of similar nature stated above by providing an approach to detect a vehicle from live closed-circuit television (CCTV) footage or video and track it down efficiently and as quickly as possible. The video analytics are done using MATLAB software.

LITERATURE REVIEW

Aerial approaches to vision are considered for detecting modes of transportation of a particular type. A bird's-eye view is one of the ideas but it has not been specifically broadened in that aspect [3]. The basic idea revolves around illumination in images.

Gathering features of interest by eliminating objects in the background like trees, billboards, shops, and so on is important. The core logic is based on whether the objects are dark and the background is lighter or vice versa. Optical vision and the Kanade Lukas Tomasi (KLT) tracker go hand in hand for the fulfillment of the detection task.

A hardware-based approach using various types of sensors can optimize the application [13]. Various types of technologies are used for detecting and tracking the vehicles with devices like light detection and ranging (LiDAR), the global system for mobile (GSM), the geographic information system (GIS), etc. The sensor nodes used for these methods are sometimes prone to damage, and there is a cost factor in their installation. Each has its benefits and disadvantages.

Detection of traffic on the basis of nodes associated with the edges aids in efficiently managing the traffic [6]. It has a YOLOv3 (You Only Look Once) model trained with a great volume of traffic data. After that, the DeepSORT (Deep Simple Online and Realtime Tracking) algorithm is optimized by retraining the feature extractor for multi-object vehicle tracking.

The Jetson TX2 platform is used for implementing the results. It shows efficient detection of traffic flow with an average processing speed of 37.9 FPS (frames per second). The feature extractor used in the DeepSORT algorithm utilizes a dataset related to pedestrians rather than vehicles. Thus, the work by [6] uses the combination of the above algorithms for traffic flow detection in the ITS (Intelligent Transportation System).

Most of the vehicle detection systems do not exclusively focus on night time detection of automobiles. This aspect of detection has been focused on reduced visibility during the night [18]. Geometric aspect on the basis of lights of automobiles of both front and back are considered with multi-camera view.

However, the limitations are pretty obvious, as it is night-time detection, so the contours of lights would not be very accurate. Another aspect to be considered is that maybe a few cars have damaged taillights or headlights.

A comprehensive study of anomaly detections in urban areas, like traffic congestion, people's gatherings, accidents, etc., is mentioned by [12]. Surveillance cameras, GPS systems, urban computing, and spatiotemporal features, form the basis for anomaly detection in this context. Deep learning techniques such as convolutional neural networks (CNNs) can be applied to process data related to city sides.

Vehicle theft detection using microcontrollers like Arduino is considered by Mallikalava et al. (2020). In such detection methods, a lot of cooperation is required from official authorities, which may not always be available due to time and operational framework constraints.

A multi-vehicle scenario can be handled by fusing different modalities of radars, sensors, and so on [7]. Data is preprocessed and fed to neural networks; image and non-image modalities are fused together, along with line-of-sight (LOS) detection by radars and sensors. This improves efficiency for vehicles to be detected where one sensor may not detect but other one might. However, installation and implementation of this may be a costly affair.

The choice of features to be detected can sort out a given problem if they are chosen wisely. A difference in the colors is one of the key factors if a smoky vehicle has to be detected [8]. The difference in pixels is simultaneously used with edge detection.

As, when there will be smoke around a particular part of a vehicle, the edge information for that part would be less as compared to the other part. Three orthogonal planes along with a histogram are used for this. For the entire experiment, focus is only on the back of the vehicle, from which smoke may come, so it is called the "key region."

Also, moving vehicles are differentiated from other areas by using their area difference in pixels. All the samples must meet this criterion, which limits the operation of this algorithm. Moreover, sometimes smoke may look grayish or white, might match with pixels of background making the smoke go undetected. Also, the smoke outlets of many vehicles may not have edges in the first place, altering the assumption of edge difference between real and false detection.

Vehicle detection using surveillance data based specifically on pixels can be another approach to handling this problem [11]. Foreground detection by eliminating the background to selectively spot the vehicles is done by the differentiation of pixels. Frames are distinguished for this purpose by choosing the moving pixels in a frame and discarding the stationary ones.

Hence, moving pixels are assumed to be vehicles. The optical flow vectors are used to generate a threshold to eliminate any other moving objects, but if a car stops moving or maybe the speed is too slow, it may go undetected.

A basic method of using optical character recognition (OCR) using an in-built MATLAB function can be used to detect characters from license plates [17]. It detects the characters only from static frames that is once chosen, which may not be able to read the number plate in live movement of vehicles. This program cannot be scaled further, as OCR may not work accurately if the vehicle is not even detected correctly in the first place. This aspect is not addressed, oversimplifying it by considering all other variables static.

Matching the features one wants to be identified with the standard features stored in a particular database beforehand can help in the recognition of desired attributes [14]. Here, features of a face are detected, particularly the lips of a person, on the basis of already stored data for comparison. Thus, for a threshold, this data is essential, and moreover, feature recognition for twins and bearded men may be a problem.

Categorizing the vehicles based on the model of cars in train datasets can lead to different annotations being considered [19]. These annotations effectively enhance the performance of YOLO v5 (version 5). Simple CNN was used for features, and anchor boxes were deduced with the aid of YOLO v5.

The implementation of the algorithm is done on Pytorch. For CNN, entire image completely is convolved without first localizing separately. The operation is basically based on similar datasets, considering different angles and the changing weather of a particular location. To work with so much specificity, the dataset for training requirements need to be specific too, which may restrict the operation of the algorithm.

Humans are considered objects to be detected [2] for abnormal event detection in online surveillance video. The model is based on an "intelligent video analytics model" (IVAM), also known as "human object detection" (HOD). IVAM is experimented with MATLAB software. The abnormal event detected may be subjectively defined, as there is no standard as to what is really abnormal behavior.

The comparison to the previous frames and actions that are considered abnormal, can change according to these referred frames. This can limit the detection of the event.

Enhanced bat optimization is used to select features for vehicle detection [5]. Just as bats focus on a target to avoid in a particular region, vehicles are pointed out in a region of interest, which is shown here taken in a rectangular geometry. The interference areas between different vehicles are removed by the enhanced convolutional neural network (ECNN) classifier.

This method is based on a threshold generated by examining pixels, which can limit the threshold. This classification is computationally complex, though.

PROPOSED METHODOLOGY

The method that is considered uses real-time data from CCTVs on highways. From the chosen video, all the frames are extracted first.

FRM_SLCN = InputVideo.NumberOfFrames;

Once the video data is extracted, segmentation is done in the frames, as video is nothing but a large number of different frames. This is where the faster RCNN approach is used.

Selective Search

The first step of this technique uses the selective search algorithm. It groups separate regions together based on their pixels. Selective Search is a bottom-up segmentation approach where sampling techniques are diversified considering scores of average best overlap (ABO) and mean average best overlap (MABO) (Ulijlings and van de Sande, 2012). Going by the hierarchical method, different scales of an image are grouped naturally until the entire image is created. The overlap scores are considered to segment the image into parts of similar pixels. It takes into account texture, pixel intensity, etc.

Region Proposals Using Faster RCNN

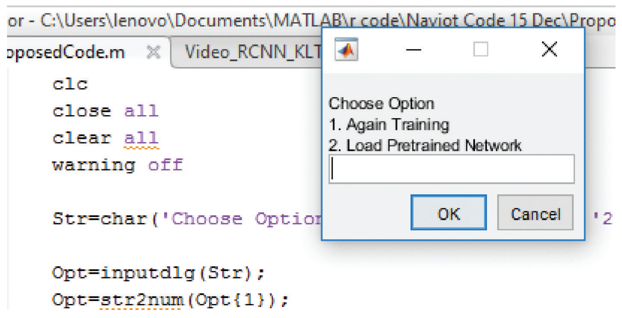
RCNN stands for Regional Convolution Neural Networks. This method consists of two steps. Firstly, the Region Proposal Network (RPN) generates a set of potential bounding boxes for objects in an image.

RPN is basically a CNN trained to identify objects of interest. In this paper, a network is trained to identify vehicles and their license plates. A bounding box is generated around it, along with a score.

In the second stage, a different CNN was used to classify the objects. This stage includes training on a large dataset of properly annotated images with predefined labels of interest.

trainingData = vehicleLicense;
inputLayer = imageInputLayer([32 32 3]);
filterSize = [3 3];
numFilters = 32; (number of filters)

The network also refines the bounding box locations using regression to more accurately enclose the objects in the image. When the entire code is run, a choice can be given to the user whether to load the pretrained network or train the network again (Figure 1). Thus, flexibility can be maintained if the dataset is updated or even changed for the vehicles.



or - C:\Users\lenovo\Documents\MATLAB\r code\Naviot Code 15 Dec\Propo

```
clc
close all
clear all
warning off

Str=char('Choose Optior                                   '2

Opt=inputdlg(Str);
Opt=str2num(Opt{1});
```

Choose Option
1. Again Training
2. Load Pretrained Network

Figure 1. Options in training the network.

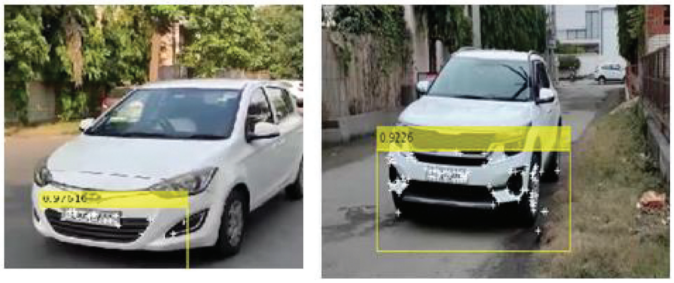


Figure 2. Bounding boxes detecting and tracking in various videos using faster RCNN and KLT.

The Faster RCNN is an object detection algorithm which gives feature vector points of a certain object, usually these denote indirectly to the bounding box coordinates of the object. The prediction is done using a regression model, and the measurement is relative to the anchor box. The coordinates generally are of bottom right and top left corner with width as well as height of the box.

So this helps in the identification of objects by the model. Faster RCNN does not directly produce feature vectors; however, they can be generated indirectly by producing a set of bounding boxes and class probabilities for objects detected. These can be further used by the KLT algorithm to track the desired features for object localization in the frames (Figure 2).

The features are extracted from the training by faster RCNN.

Let {G} be the set of feature vectors generated from Faster RCNN, {g_t} be the feature tracker, and {s_t} be the motion model. The predicted location of each feature at time {t} is given by:

{x_t}={ g_t(G_{t_1})+s_t(G_{t-1},G_t)}

where {g_t} is applied to the set of feature vectors from the preceding frames {G_{t-1}}, for the prediction of location's each feature in the current frame.

Thus, KLT uses this motion model {s_t} to predict location of each tracked feature in the succeeding frame. It is typically a linear motion model.

Simplified Illustration of Proposed Algorithm

- For detection using Faster RCNN, its CNN is run on the input image for bounding box coordinates and class labels of objects detected [15].
  For bounding box regression, we adopt the parameterizations of the four coordinates following:

$$tx = (x-xa)/wa,$$
$$ty = (y-ya)/ha,$$
$$tw = \log(w/wa),$$
$$th = \log(h/ha),$$
$$t{*}\,x = (x{*}-xa)/wa,$$
$$t{*}\,y = (y{*}-ya)/ha,$$
$$t{*}\,w = \log(w{*}/wa),$$
$$t{*}\,h = \log(h{*}/ha),$$

  where x, y, w, and h denote the box's center coordinates and its width and height. Variables x, xa, and x∗ are for the predicted box, anchor box, and groundtruth box, respectively (likewise for y,w,h).
- The accuracy of predicted values of the bounding boxes is depicted using mean squared error (MSE). Its average is used for training the dataset.
- KLT algorithm is operated on objects within the bounding boxes for set of feature vectors (FVs), which are used for object recognition for tracking purposes. For example, if we consider that we have used faster RCNN and KLT to extract FVs for the car and apple in the input image, the output values may look something like this:

  Car: (a1,b1,a2,b2)=(150,70,250,90), FV=[.1,.2,.3.....]
  Apple: (a1,b1,a2,b2)=(100,50,200,150), FV=[.4,.5,.6....]
- Now, these extracted FVs are used for object recognition, as the above FVs can be compared to a set of known FVs of the same objects, and similarity is predicted by a measure called Euclidean distance, which may look like this:

  D=sqrt((.11-.1)^2+(.22-.2)62+(.33-.3)^2+.......)

  This D can be measured with a threshold value for object detection. The above instance simplifies the concept used, which realistically involves complex equations as seen in Shaoqing Ren and Kalming He (2016).

RESULTS AND COMPARISON

The analysis has been carried out in MATLAB software experimentally. The dataset consisting of license plates of various vehicles is taken from https://www.kaggle.com/datasets/andrewmvd/car-plate-detection. The performance parameters like precision, recall, accuracy, etc., have been compared between the proposed faster Regional Convolution Neural Network with Kanade Lukas Tomasi (Faster RCNN with KLT) and the existing enhanced CNN with support vector machine (SVM with ECNN). The comparison is depicted in Figures 3–6.

Precision

Precision basically refers to the ratio of true positive predictions to the total positive predictions, including the false ones. Its formula can be generally stated as in "Towards Data Science, 2020":

$$\frac{\text{True positives}}{\text{True positives} + \text{false positives}} = \frac{\text{No. of correctly predicted positive instances}}{\text{No. of total positive predictions made}}$$

Figure 3 shows the comparison between faster RCNN with KLT and ECNN with SVM for the precision parameter.

As it is visible, the precision ratio in both methods is nearly same, which is theoretically 100%. This value is reached by using the same dataset in both works.

Accuracy

As the term indicates, accuracy defines how accurately an algorithm performs. It is obtained by dividing correct predictions by all the predictions in total. It can be written as in "Towards Data Science, 2020":

$$\frac{\text{True positives} + \text{True negatives}}{\text{True positives} + \text{True negatives} + \text{False positives} + \text{False negatives}} = \frac{\text{No. of correct predictions}}{\text{No. of all predictions}} = \frac{\text{No. of correct predictions}}{\text{Size of dataset}}$$

Figure 4 depicts the accuracy parameter.

The value of accuracy in proposed work is 99.267% and for the existing work it is 74.572%. This clearly shows the edge that proposed work has over existing work.

Recall Parameter

This parameter is calculated by dividing the correctly observed positive samples by the total of these samples. This can be stated as in "Towards Data Science, 2020":

$$\frac{\text{True positives}}{\text{True positives} + \text{False negatives}} = \frac{\text{No. of correctly predicted positive instances}}{\text{No. of total positive instances in the dataset}}$$

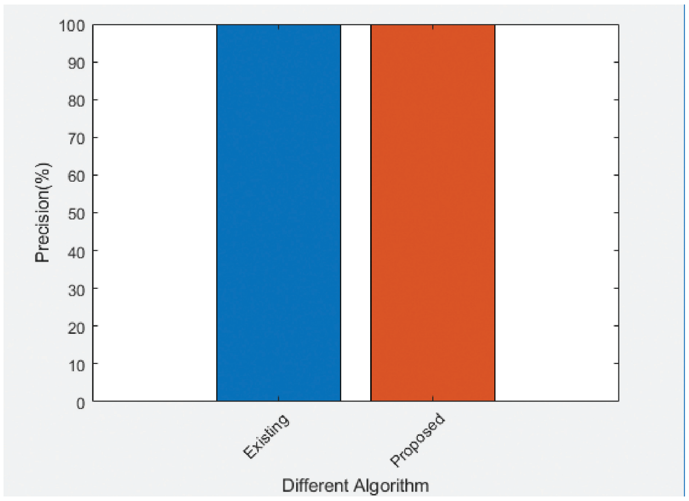


Figure 3. Comparison of precision.

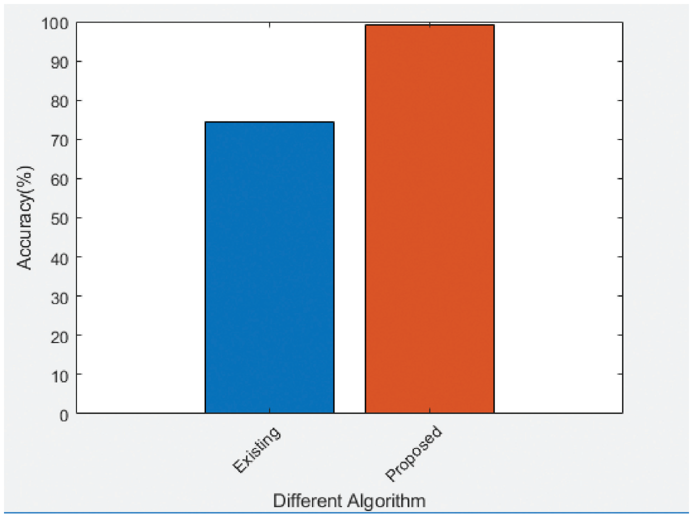


Figure 4. Comparison of accuracy.

Figure 5 shows the comparison for respective considered work for the above parameter.

The value of recall is 99.2% and 74.5% for proposed and existing work, respectively.

F-score Parameter

The F-score is calculated using recall and precision. It includes the average means of these two aforementioned

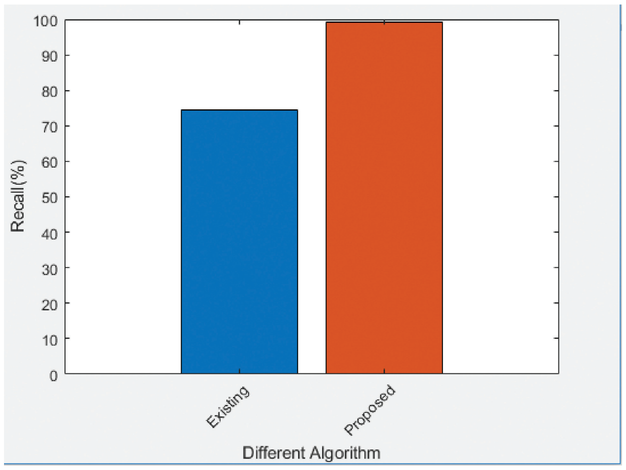


Figure 5. Comparison of accuracy.

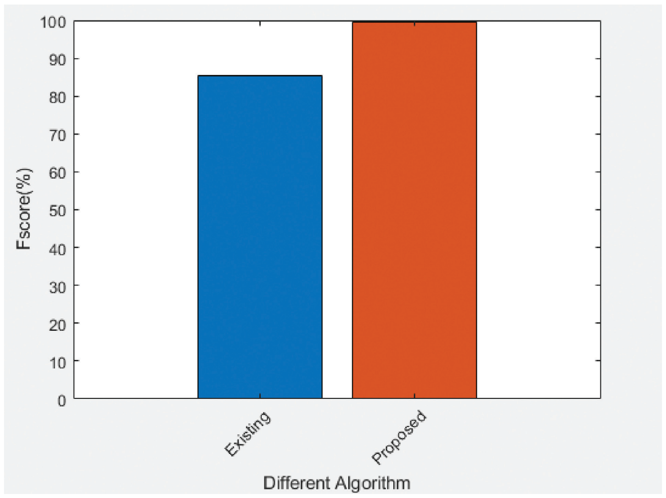


Figure 6. Comparison of F-score.

parameters. It can also be written as in "Towards Data Science, 2020":

$$\text{F-score} = 2^* \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

The comparison of proposed and existing work for the aforementioned parameter is illustrated in Figure 6.

The F-score for proposed work is 99.6% and for existing work is 85.4%. It is clearly visible from the above results that the proposed method shows better parameters. One of the reasons of this is Faster RCNN used instead ECNN of existing work. Existing work basically uses convolution without considering proposals for regions for localization.

Additionally, it uses the list method for tracking. The old and new lists need to be updated every time tracking is done. This may end up making it computationally expensive. The proposed work simultaneously uses region proposal for implementation which is one of the reason for high recall and a more efficient performance.

CONCLUSION AND FUTURE OPTIMIZATION

There are myriad variables and circumstances that can play a part in detecting and tracking a vehicle. Faster RCNN with KLT does this job efficiently as far as license plates are concerned. The biggest window for improvisation is opened by this method, as learning and training of the network can be done with multitudes of datasets.

This basically shows the network is ever ready to learn whatever a user may want to teach it, specifically here, for vehicles. With such high performance parameters, this technique can aid smart automobile system efficiently.

REFERENCES

[1] A Look at Precision, Recall, and F1-Score. (2020, September 12). Towards Data Science. Retrieved from the Towards Data Science website: https://towardsdatascience.com.

[2] A. Balasundaram & C.Chellappan. (2018). An intelligent video analytics model for abnormal event detection in online surveillance video. *Journal of Real-Time Image Processing*, 17, 915–930.

[3] Ahmed Gomaa, Moataz M.Abdelwahab & Mohammed Abo-Zahhad. (2020). Eficient vehicle detection and tracking strategy in aerial videos by employing morphological operations and feature points motion analysis. *Multimedia Tools and Applications*, 79, 26023–26043.

[4] Ameena Padiath, Lelitha Vanajakshi, Shankar C. Subramanian & Harishreddy Manda. (2009). Prediction of traffic density for congestion analysis under indian traffic conditions. *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems*, 1–6.

[5] C. Ranjeeth Kumar & R. Anuradha. (2020). Feature selection and classification methods for vehicle tracking and detection. *Journal of Ambient Intelligence and Humanized Computing*, s12652-020-01824-3.

[6] Chen Chen, Bin Liu, Shaohua Wan, Peng Qiao & Qingqi Pei. (2021). An edge traffic flow detection scheme based on deep learning in an intelligent transportation system. *IEEE Transactions On Intelligent Transportation Systems*, 22(3), 1840–1852.

[7] Debashri Roy, Yuanyuan Li, Tong Jian, Peng Tian, Kaushik Chowdhury & Stratis Ioannidis. (2022). Multi-modality sensing and data fusion for multi-vehicle detection. *IEEE Transactions on Multimedia*, 1–1.

[8] Huanjie Tao & Xiaobo Lu. (2018). Smoky vehicle detection based on range filtering on three orthogonal planes and motion orientation histogram. *IEEE Access*, 6, 57180–57190.

[9] J. Haapala. (2017). Recurrent neural networks for object detection in video sequences. Aalto *Univ., Espoo, Finland, Tech.* Rep. SCI3044, p. 58.

[10] J.R. Rulijlings & K.E.A van de Sande. (2012). Selective search for object recognition, *International Journal of Computer Vision*, 2, 154–171.

[11] K.V. Arya, Shailendra Tiwari & Saurabh Behwal. (2016). real-time vehicle detection and tracking. (2016). *13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, 1–6.

[12] Mingyang Zhang, Tong Li, Yue Yu, Yong Li, Pan Hui, & Yu Zheng. (2019). Urban Anomaly analytics: description, detection and prediction. *IEEE Transactions on Big Data*, 8(3), 809–826.

[13] Pankaj P. Tasgaonkar, Rahul Dev Garg & Pradeep Kumar Garg. (2020). Vehicle detection and traffic estimation with sensors technologies for intelligent transportation systems. *Sensing and Imaging*, 21(29).

[14] Sasikumar Gurumurthy & B.K.Tripathy. (2012). Design and implementation of face recognition system in matlab using the features of lips. *I.J. Intelligent Systems and Applications*, 30–36.

[15] Shaoqing Ren & Kalming He. (2016). Faster RCNN: Towards real-time object detection with region proposal networks. *Computer Vision and Pattern Recognition*, 1(3), ariXv:1506.01497.

[16] Vijeta Sharma, Manjari Gupta, Ajai Kumar & Deepti Mishra. (2021). Video processing using deep learning techniques: a systematic literature review. *IEEE Access*, 9(7), 139489–139507.

[17] Vivek Singh, Yogesh Verma, Tejas Bhavsar, Sandeep Saini & Garvit Gupta. (2021). Vehicle number plate recognition using matlab. *International Journal of Electrical, Electronics and Computers*, 6(3), 24–26.

[18] Xinxiang Zhang, Brett Story & Dinesh Rajan. (2021). Night time vehicle detection and tracking by fusing vehicle parts from multiple cameras. *IEEE Transactions On Intelligent Transportation Systems*, 23(7), 8136–8156.

[19] Yu Zhang, Zhongyin Guo, Jianqing Wu, Yuan Tian, Haotian Tang & Xinming Guo. (2022). real-time vehicle detection based on improved yolo v5. *Sustainability, Multidisciplinary Digital Publishing Institute (MDPI)*, 14(19), 12274.

[20] Z.-Q. Zhao, P. Zheng, S.-T. Xu & X. Wu. (2019). Object detection with deep learning: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*30(12), pp. 3212–3232.