E-ISSN 2581 – 7957
P-ISSN 2277 – 3916
CVR Journal of Science and Technology, Volume 24, June 2023
DOI: 10.32377/cvrjst2412

Real Time Video based Vehicle Detection, Counting and Classification System

K. Sai Keerthan[1], R Vishal[2], T. Tanay[3], M. Vasavi [4]
[1]UG Scholar, CVR College of Engineering/CSE Department, Hyderabad, India
Email: saikeerthankothawadla@gmail.com
[2] UG Scholar, CVR College of Engineering/CSE Department, Hyderabad, India
Email: vishalrambathri02@gmail.com
[3] UG Scholar, CVR College of Engineering/CSE Department, Hyderabad, India
Email: tanaychowdary@gmail.com
[4]Sr.Asst.Professor, CVR College of Engineering/CSE Department, Hyderabad, India
Email: m.vasavi@cvr.ac.in

Abstract: **City planners have wrestled with traffic challenges for a long time. Better techniques for simplifying the process and analyzing traffic are currently being developed. Both the quantity of vehicles at a specific location over a specific time period and the kind of vehicles can be taken into account for traffic analysis. Such devices have been created for decades, but the bulk of them use sensors to identify the moving cars, such as a couple of proximity sensors to determine the direction of the driving vehicle and to count the number of moving vehicles.**

These systems are highly effective and have matured, however they are not very cost-efficient. The problem is that such systems demand routine maintenance and calibration. Creating a vision-based vehicle counting and categorization system is the aim of this project. In order to do feature extraction and be able to identify and count the vehicles, this system takes still pictures from video. The cars are then categorized by comparing the contour regions to the predetermined values. The comparison of two classification algorithms is the work's significant contribution. Utilizing both the Bag of Features (BoF) approach and contour comparison (CC), classification has been achieved.

Index Terms: **Background learning, Foreground extraction, Vehicle classification, YOLO algorithm, COCO dataset.**

I. Introduction

There is a greater need for effective management and monitoring of road traffic as a result of the growth of road networks, an increase in the number of vehicles, and, most importantly, the size of vehicles. Intelligent traffic surveillance systems are crucial for modern traffic management, but conventional approaches like wireless sensor networks, inductive loops, and EM microwave detectors are expensive, heavy, and difficult to install without causing traffic disruptions. Instead, video-based surveillance systems can be an effective alternative. Video surveillance systems are now more accessible and efficient because of advancements in storage capacity, processing power, and video encryption techniques.

The videos that these surveillance systems stores are normally examined by humans, which takes time. The need for more reliable, automatic video-based surveillance systems is essential. Although the primary function of a traffic surveillance system are to detect, monitor, and categorize cars, they can also be used to carry out more complicated operations like lane recognition and driver behavior recognition. Systems for monitoring traffic can be used for a number of things, such as identifying people, detecting unusual behavior, detecting accidents, detecting car theft, monitoring parking places, and detecting accidents. Hardware and software are the two main components of a traffic monitoring system.

The hardware component is a stationary camera installed on the side of the road that records the video stream, and the processing and analysis is handled by software component. These systems may be portable and include a CPU built into the camera for instantaneous processing and evaluation, or they may be nothing more than cameras that send the video stream to a processing hub. A variety of techniques have been employed to develop systems that can recognize, count, and categorize cars and may be used for traffic monitoring in intelligent transportation systems. These systems are discussed in this section along with details on their creation.

For traffic monitoring, computer vision technology is widely used. A crucial element of ITS is the development of computer vision technologies over video-based traffic monitoring for spotting moving cars in video feeds. Vehicle tracking and detection using computer vision technologies has been the subject of extensive research. In 2005, Hasegawa and Kanade unveiled a technique for identifying and categorizing moving objects according to their type and color. A series of photographs of a particular place were made available during this process, and vehicles were located using these images. Using Python and OpenCV, Nilesh et al. (2013) created and constructed a system for identifying and counting moving automobiles. Using background subtraction, picture filtering, image binary, and segmentation techniques, it can identify and count moving objects like cars in real time or from recorded movies. Using Python and OpenCV, as well as an adaptive background subtraction technique in conjunction with virtual detector and blob tracking technologies, Da Li et al. (2014) developed a real-time moving vehicle identification, tracking, and counting system. The virtual detector creates a collection of rectangular regions in each input image frame, and the blob tracking technique creates input image frames with the absolute difference between the background image and foreground blobs representing the moving cars. The techniques outlined above have some

Received on 02-04-2023, Revised on 11-04-2023, Accepted on 26-04-2023.

limitations, including how they handle shadows and how they obstruct many vehicles that appear in a single area. In order to create a device that can be used for traffic surveillance in intelligent transportation systems, a variety of approaches have been used.

II. Literature survey

For a system that counts vehicles in real time using video, Tursun, M., and Amrulla, G. [10] developed an efficient virtual loop technique. To determine how many vehicles pass through, they deployed sophisticated traffic monitoring cameras placed along roads. This technology counts in three phases by monitoring the movement of the vehicle inside a tracking area known as the virtual loop. Lei, M., et al. [11] suggested yet another car counting mechanism based on video. Automatic feature estimation and Gauss shadow elimination were the two main methods used in this system, which employed surveillance cameras positioned at relatively high elevations to gather the traffic video feed.

The visual angle and the system's capacity to eliminate shadow and ghost effects are what affect the accuracy rate. The system's fundamental flaw is its inability to categorize vehicle types. Bas et al. [12] The authors introduced a method for counting vehicles through video analysis. The approach utilizes an adjustable bounding box size for tracking and detecting vehicles, based on an estimation of camera distance. A boundary is often defined in a picture for both inbound and outbound directions in order to identify the Region of Interest (ROI). The system is unable to monitor moving cars as they change directions, despite advancements to deal with specific weather situations.

N.C. Mithun et al. [13] The authors suggested a system for detecting and classifying vehicles that relies on time spatial images and the use of virtual detecting lines spread out. A two-step K closest neighbor (KNN) approach is used to classify automobiles using shape-invariant and texture-based criteria. Experiments reveal that the suggested method has higher accuracy and a lower error rate than earlier methods since it accounts for different illumination settings. A system for recognizing and classifying automobiles at busy urban crossroads was proposed by Habibu Rabiu. [14]. The system uses background elimination, the Kalman filter method, and a Linear Discriminant Analysis classifier to accurately classify cars in order to detect and track them.

The most crucial stage of a video-based traffic monitoring system is the initial phase which is vehicle recognition since it has a significant impact on other algorithms like tracking and classifying the vehicles. For this reason, it's crucial to accurately recognize and separate from the background moving item. Foreground detection uses many techniques, such as frame n differencing [15]. The most basic foreground identification and segmentation technique is frame differencing, it relies on the intimate association of the series of moving images.

Collins [16] proposed a more accurate frame differencing technique that computes the foreground by comparing the differences between numerous frames as opposed to simply the initial one. Gibson [17] created the optical flow field method, a different technique. Wu, K., et al. postulated that mode velocity within a picture is represented by optical flow [18]. The Optical Flow approach treats the Detecting Area Image as a vector field of velocity, with each vector representing the momentary change in a pixel's position relative to its surroundings. Another technique for identifying foreground objects is the average model. [19]. The background value of a pixel in the average model is the average grey value of that pixel over a set of frames.

Friedman, N., and S. Russell [20] GMM was suggested by Stauffer, C., and W.E.L. Grimson [21, 22] improved for tracking in real-time. The backdrop is assumed to be more apparent than any foreground areas in the Gaussian Mixture Model. A nonparametric background model based on kernel density estimation was proposed by Elgammal [23]. The kernel density estimation method chooses the data with the highest probability density as the background after evaluating video data samples using kernel functions. Images are portrayed by the bag of features model as random groupings of local features. The bag of words representation used in text-based information retrieval served as the model for the term "bag of features."

III. Implementation

The system is made to perform vehicle detection, recognition, and tracking in video frames. It then divides the identified vehicles into three separate size groups. The system is made up of three modules: Vehicle categorization, Foreground extraction, and Background learning.

Background Learning:

Background Learning is used to create a background model of the video scene by analyzing the pixels in each frame of the video over a period of time. This background model is then used to identify any changes in the scene, which could indicate the presence of a moving object.

The first module in the vehicle detection system is responsible for learning about the background in the video feed. This is important for identifying the moving objects or foreground. The module extracts frames from the video and uses image processing algorithms to learn about the static backgrounds in the scene. For instance, in a traffic scenario captured with a stationary camera, the background would include static objects such as buildings and road signs. By analyzing the differences between the background and foreground, the system can accurately detect and track vehicles in the video feed. The use of image processing algorithms allows the system to learn about the background and identify changes in the foreground in real-time.

Foreground Extraction:

Foreground Extraction uses the background model to subtract the background from each frame of the video, leaving only the foreground pixels, which correspond to the moving objects in the scene. This module is used to isolate the vehicles in the video frames.

The second module in the vehicle detection system comprises three steps: background subtraction, image enhancement, and foreground extraction. First, the background is subtracted to make the foreground objects visible. This is done by assigning binary 0 to the pixels that correspond to static objects in the scene. Next, to acquire accurate foreground object contours, picture enhancing techniques like noise filtering, dilation, and erosion are used.

These techniques help to improve the quality of the foreground objects and remove any artifacts or noise. Finally, the module outputs the foreground extraction, which is a set of contours representing the moving vehicles in the video frames. These contours will then be used in the subsequent module for vehicle classification based on their size and shape.

Vehicle Classification:

Once the foreground extraction module is applied in the vehicle detection system, proper contours of the vehicles in the video frames are obtained. From these contours, features such as contours aspect ratio, area, size, and solidity are extracted. These features are then used to classify the vehicles based on their size and shape. For example, the system could classify vehicles into small, medium, and large categories based on their total length and area. Alternatively, it could classify vehicles based on their shape, such as car vs. trucks using aspect ratio as a classification parameter. The extraction of these features is crucial in accurately identifying and categorizing the vehicles in real-time.

IV. WORKING

Steps for execution:

1. Importing the necessary libraries and setting up the vehicle detection, counting, and categorization system are the first steps in the code.
2. The pre-trained object identification model is then loaded using the YOLOv3 method.
3. The video stream is then captured and examined using OpenCV frame by frame.
4. The vehicles in each picture are recognized using the object detection model.
5. The monitored vehicles are recognized using a centroid tracker approach, which gives each one a unique ID and monitors its movement over time.
6. Following that, the system counts the number of vehicles passing through a certain Region Of Interest (ROI) and classifies them based on type (car, bus, truck, etc.).
7. A pre-trained machine learning model that recognizes the type of vehicle based on image features is used to do the categorization.
8. The system then updates the count and classification data for each car, displaying it on the output video stream.

The completed video stream is finally exported to a file for additional use. Overall, the system uses deep learning and computer vision algorithms to provide real-time vehicle recognition, counting, and categorization for a video stream.

Details of each Step:

1. **Import required libraries**: First, the necessary libraries, including argparse, NumPy, and OpenCV, are imported.
2. **Initialize the camera**: The OpenCV function named Video Capture is then used to initialize the camera.
3. **Initialize required variables:** There are fixed variables for frame count, width, height, and font.
4. **YOLOv3 network should be loaded:** The pre-trained YOLOv3 model is loaded using the cv2.dnn.readNet function.
5. **Describe the classes and colors:** Both the categories of objects that the model can recognize, and the colors connected to them are specified.
6. **Specify the output layers**: The output layers of the YOLOv3 network have been identified.
7. **Start an endless loop**: The camera's frames are started to be recorded in an endless loop.
8. **Image preprocessing:** Scaling and normalizing the pixel values are two steps in the preprocessing of the recorded frame.
9. **Perform network inference:** To find objects, the preprocessed image is passed via the YOLOv3 network.
10. **Postprocess the detections:** Postprocessing removes low confidence and non-vehicle detections from the YOLOv3 network's detections. The threshold for low confidence detections is set to 0.3. This means that any detected object with a confidence score below 0.3 will be considered as a low confidence detection and ignored.
11. **Count vehicles:** The number of detected vehicles for the current frame is counted and shown on the output.
12. **Vehicle classification:** The aspect ratio of each vehicle (car, truck, or bus) determines its classification.
13. **13. Define bounding boxes and labels for each vehicle:** Each detected vehicle has bounding boxes and labels placed around it, and on the output, each vehicle's type is indicated.
14. **Show the results**: The show function in OpenCV is used to display the result frame on the screen.

V. DATASET AND CLASSES

The dataset used is COCO. The COCO (Common Objects in Context) dataset is a widely used benchmark dataset in computer vision. It includes over 200,000 diverse images with 80 objects categories, annotated with masks, bounding boxes, category labels. The dataset is popular for training and evaluating object detection algorithms like YOLO. It has also spurred competitions and challenges, driving advancements in the field of computer vision. Overall, the COCO dataset serves as a standard resource for benchmarking object algorithms and promoting research in the field.

The total number of classes in the provided code depends on the implementation and dataset used to train the YOLO model. In this code, the number of classes is set to 80, which is a common value used for the COCO dataset. The COCO dataset is widely used for object detection tasks and includes 80 different object classes.

However, based on the COCO dataset used for training, which includes 80 object categories, the classes could potentially include a wide range of common objects and vehicles, such as:

Person, Bicycle, Car, Motorcycle, Bus, Truck, Traffic Light, Stop sign, Pedestrian, Motorbike, Caravan, etc.

In the provided code, if a vehicle is not part of the classes defined in the COCO dataset, the YOLO model may classify it as a generic class like "object", "unknown", or "other". As a result, the code could potentially detect and count vehicles that are not included in the COCO dataset, but it will not provide specific class labels for those vehicles. The model's behavior for unclassified vehicles may vary depending on the implementation and configuration of the YOLO algorithm being used. It's important to consider the limitations of the dataset and model when using the code for vehicle detection and classification tasks.

VI. Results

Figure 1. It shows the first step for executing the project where a command is passed in the command prompt to start the GUI.

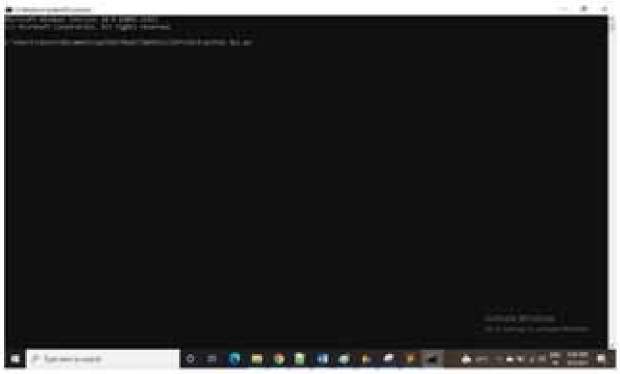

Figure 1. Command prompt in which command executed.

Figure 2. Graphical User Interface is opened after the command executed in the command prompt which displays the option of choosing the video to process in the project. We should click on the button "Upload A Video" located in the bottom of the window, which then displays the window of folder to select a video.

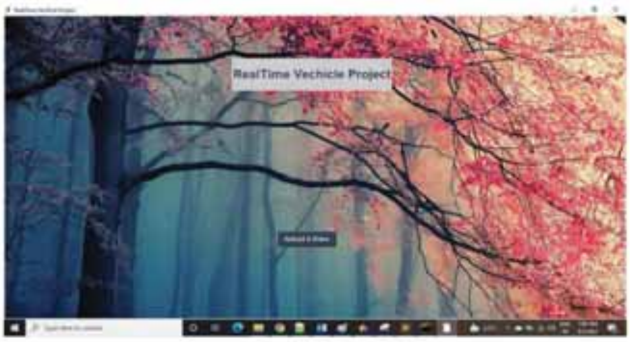

Figure 2. Graphical User Interface

Figure 3. Display of GUI after the selection of the video to be analyzed in the project. We should click on the "Get RealTime Vehicle Reading" option to start analyzing the video and start detecting and counting of the vehicles in the video.

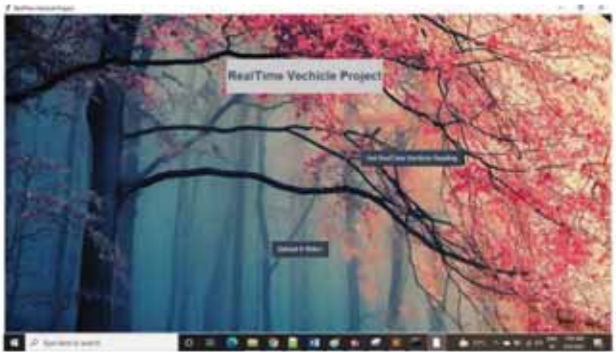

Figure 3. GUI after selection of the video to be processed.

Figure 4. After clicking on the "Get RealTime Vehicle Reading" option, it displays all the background details of the execution like number of frames per second captured and the count of the vehicle in the command prompt.

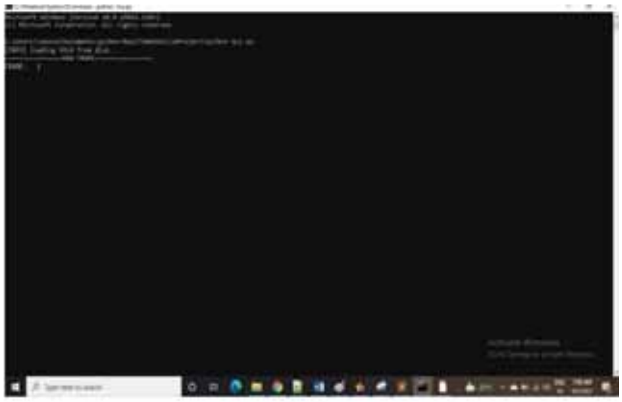

Figure 4. Background details of the execution.

Figure 5. Finally, it displays a video which consists of the traffic, the count of the vehicles moving on the road and the classification of vehicle captured from the input video.

Figure 5. Output video with the count of vehicles

VII. Conclusions

The proposed fix makes use of the OpenCV bindings and is written in Python. Various sources of traffic camera footage are being utilized. The user selects the area of interest to be looked into using a straightforward interface, and image processing methods are then applied to count the number of vehicles and categorize the cars. We developed a video-based vehicle monitoring system for real-time traffic data collection. Python, OpenCV, and the Background Subtraction Yolo algorithm were used to create the system. In the suggested model we studied the traffic flow in the day and night. Additionally, we evaluated the various shadows cast by moving autos.

REFERENCES

[1] I. Alam, M. F. Ahmed, M. Alam, J. Ulisses, D. M. Farid,Shatabda, and R. J. F. Rossetti, "Pattern mining from historical traffic big data," in IEEE Technologies for Smart Cities (TENSYMP 2017). IEEE, July 2017, pp. 1–5.

[2] I. Alam, D. M. Farid, and R. J. F. Rossetti, "The prediction of traffic flow with regression analysis," in Emerging Technologies in Data Mining and Information Security, ser. Advances in Intelligent Systems and Computing, A. A., D. P., M. J., B. A., and D. S., Eds. Springer, Singapore, 2019, vol. 813, pp. 661–671.

[3] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Effect of information availability on stability of traffic flow: Percolation theory approach," Transportation Research Procedia, vol. 23, pp. 81–100, 2017.

[4] A. Csikos, T. Charalambous, H. Farhadi, B. Kulcs ´ ar, and H. Wymeersch, ´ "Network traffic flow optimization under performance constraints," Transportation Research Part C: Emerging Technologies, vol. 83, pp. 120–133, 2017.

[5] M. Zhou, X. Qu, and X. Li, "A recurrent neural network based microscopic car following model to predict traffic oscillation," Transportation Research Part C: Emerging Technologies, vol. 84, pp. 245– 264, 2017.

[6] F. B. Ghorghi and H. Zhou, "Traffic control devices for deterring wrongway driving historical evolution and current practice," Journal of Traffic and Transportation Engineering, vol. 4, pp. 280–289, 2017.

[7] A. Abadi, T. Rajabioun, and P. A. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 653–662, 2015.

[8] S.-Y. Cheung, and P.P. Varaiya, "Traffic surveillance by wireless sensor networks: Final report", PhD diss., University of California at Berkeley, 2006.

[9] S. Oh, S. Ritchie, and C. Oh, "Real-time traffic measurement from single loop inductive signatures", Transportation Research Record: Journal of the Transportation Research Board, (1804), pp. 98-106, 200

[10] M. Tursun, and G. Amrulla, "A video based real-time. vehicle counting system using optimized virtual loop method", IEEE 8th International workshop on Systems Signal Processing and their Applications (WoSSPA), 2013.

[11] M. Lei, D. Lefloch, P. Gouton, K. Madani, "A video based real-time vehicle counting system using adaptive background method", IEEE International conference on Signal Image Technology and Internet Based Systems (SITIS'08), pp. 523-528, 2008.

[12] Bas, Erhan & Tekalp, A. & Salman, F.. (2007). Automatic Vehicle Counting from Video for Traffic Flow Analysis. 392 - 397. 10.1109/IVS.2007.4290146.

[13] N.C. Mithun, N.U. Rashid, and S.M. Rahman, "Detection and classification of vehicles from video using multiple time-spatial images", IEEE Transactions on Intelligent Transportation Systems, vol 13, no 3, p. 1215-1225, 2012.

[14] H. Rabiu, "Vehicle detection and classification for cluttered urban intersection", International Journal of Computer Science, Engineering and Applications, vol 3, No.1, p. 37, 2013.

[15] M. Seki, H. Fujiwara, and K. Sumi, "A robust background subtraction method for changing background", Fifth IEEE Workshop on Applications of Computer Vision, 2000.

[16] R.T. Collins, et al., "A system for video surveillance and monitoring", VASM final Report, Robotics Institute, Carnegie Mellon University, 2000, pp.1-68.

[17] J.J. Gibson, "The perception of the visual world", 1950.

[18] K. Wu, et al., "Overview of video-based vehicle detection technologies", 2011.

[19] G. Yang, "Video Vehicle Detection Based on Self Adaptive Background Update", Journal of Nanjing Institute of Technology (Natural Science Edition), vol 2, p. 13, 2012.

[20] N. Friedman, and S. Russell, "Image segmentation in video sequences: A probabilistic approach", Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence, 1997, Morgan Kaufmann Publishers Inc.

[21] C. Stauffer, and W.E.L. Grimson, "Learning patterns of activity using real-time tracking", IEEE Transactions on pattern analysis and machine intelligence, 2000. Vol 22, no. 8, pp. 747-757, 2000.

[22] C. Stauffer, and W.E.L. Grimson, "Adaptive background mixture models for real-time tracking", IEEE Computer Society Conference Computer Vision and Pattern Recognition, 1999.

[23] A. Elgammal, D. Harwood, and L. Davis, "Nonparametric model for background subtraction", European conference on computer vision, Springer, 2000.