DESIGN AND ANALYSIS OF ALGORITHMS

PROJECT REPORT

ON

TOPIC: VEHICLE IDENTIFICATION AND COUNTING USING MACHINE LEARNING

SUBMITTED BY:
AVINASH KUMAR (2230243)

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY (KIIT)

Deemed to be University U/S 3 of UGC Act, 1956

B. TECH IN ELECTRONICS & COMPUTER SCIENCE ENGINEERING
SCHOOL OF ELECTRONICS ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY,
DEEMED TO BE UNIVERSITY
BHUBANESWAR, INDIA

NOVEMBER 2024

CERTIFIACATE

This is to certify that "**Avinash Kumar**" of 3rd year of Electronics & Computer Science has submitted minor project report entitled "**Vehicle Identification And Counting Using Machine Learning**" in partial fulfilment for the award of Bachelor of Technology of KIIT, Bhubaneshwar in session 2024. It has been found to be satisfactory and hereby approved for the submission.

DR. SRICHETA PARUI
(PROF. OF CSE DEPTT.)

DECLARATION

I, **Avinash Kumar**, student of **Kalinga Institute Of Industrial Technology, Deemed To Be University**, hereby declare that the project titled **“Vehicle Identification And Counting Using Machine Learning”** has been carried out by me as part of the coursework for the subject **Design and Analysis of Algorithms** under the supervision of **DR. SRICHETA PARUI**, Department of Computer Science.

I further declare that this project is my original work and has not been submitted for the award of any degree, diploma, or other similar title elsewhere.
All sources of information used in this project have been acknowledged in the references section of this report.

Avinash Kumar
Roll No- 2230243
Date: 3rd November 2024

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to **DR. SRICHETA PARUI** for their continuous support, guidance, and encouragement throughout this project. Their invaluable insights and feedback helped me navigate challenges and improve my understanding of computer vision and human-computer interaction. I also thank my family and friends for their moral support throughout this project. Lastly, I extend my gratitude to the developers and community behind open-source libraries such as OpenCV, and ORB, Machine Learning Algorithm whose contributions made this project possible.

Avinash Kumar
(2230243)

ABSTRACT

In these modern days the use of vehicles has significantly increased, resulting in challenges related to their Planning, monitoring, and control. To overcome this issue, the vehicle detection and counting system is employed. In the proposed work, we designed and implemented a model which takes a video as an input and extracts the frames from that video clip, and processes them to detect and count the vehicles. Those frames are converted into grayscale images on which background subtraction, frame differentiation, contours, and morphological methods are applied for vehicle counting. This system employs a background subtraction algorithm for vehicle detection, which enables accurate results.
There are several applications for the technology used to detect vehicles in recorded video. In this research work, the Oriented FAST and Rotated BRIEF (ORB) method is used to develop a rudimentary model for vehicle detection and counting. The video clip used as input, numerous frames are retrieved, and shadow and backdrop are approximated. To identify every moving object from the estimated backdrop, the subsequent frame is subtracted. Vehicles are identified, categorized, and counted for traffic estimation based on moving objects, utilizing object detection methods and OpenCV. When used with OpenCV, the ORB algorithm offers a potential method for identifying and accumulation the number of vehicles in the video stream. The ORB method is well-suited for tracking objects in dynamic settings since it is especially supportive for feature matching.

Keywords— Vehicle Identification, ORB, Machine Learning Algorithm, Gaussian filter.

INTRODUCTION:

With the rise in the number of vehicles, roads are becoming increasingly congested. To manage this traffic flow and to provide efficient planning for the drivers, an Intelligent Transport System is required. In this regard, vehicle detection and count systems play a crucial role. However, the analysis of videos captured by these systems is typically performed manually, which is time-consuming. To address this issue, there is a growing interest in computer vision and the development of more robust and automatic video based surveillance systems. With advances in storage and computational capabilities, video surveillance systems have become more efficient. One of the advantages of this system is its ability to automatically calculate the number of vehicles by utilizing cameras.

This detection system will get the statistics i.e., the input in real time, and the records can be used for real-time tracking of the volume of traffic at the traffic signal. Detection of a vehicle that is in motion is usually carried out by using different computer vision techniques like background image subtraction. The fundamental idea behind this background subtraction is to construct a background model from the collection of video frames in order to detect the moving objects using both the background estimation and the current frame. This project would benefit two groups firstly the road users and the traffic administrations. On the other side traffic management systems can utilize the traffic information (which includes the count of vehicles in the specific area) in their traffic control systems, resulting in better traffic management.

ORB is used for real-time applications where performance is mandatory. In urban areas, the number of vehicles and traffic conditions has increased, which has led to a rise in the demand for traffic condition classification in urban areas. ORB provides better extraction features than compared to the speed of SIFT.

Several OpenCV techniques are used to detect moving vehicles more accurately. It includes thresholding values, morphology operations and hole filling process. Also Gaussian blur is a smooth method of image smoothing. Detection and count the vehicles automatically when vehicles cross at some arbitrary areas and period of time. But, it is very essential to count, detect and track the vehicles in the certain areas.

LITERATURE REVIEW:

This detection system help us to detect the real time input and record the real time tracking of the volume of traffic at the traffic signal. Detection of a vehicle that is in motion is usually carried out by using different computer vision techniques like background image subtraction. The fundamental idea behind this background subtraction is to construct a background model from the collection of video frames in order to detect the moving objects using both the background estimation and the current frame. This project would benefit two groups firstly the road users and the traffic administrations. On the other side traffic management systems can utilize the traffic information (which includes the count of vehicles in the specific area) in their traffic control systems, resulting in better traffic management. In this regard, vehicle detection and count systems play a crucial role. However, the analysis of videos captured by these systems is typically performed manually, which is time-consuming.

In this system counting is completed in three steps by tracking vehicle movements within a tracking zone called virtual loop. Another video based vehicle counting system. In this system surveillance cameras were used and mounted at relatively high place to acquire the traffic video stream, the Adaptive background estimation and the Gaussian shadow elimination are the two main methods that were used in this system. The accuracy rate of the system depends on the visual angle and ability to remove shadows and ghost effects. The system's incompetency to classify vehicle type is the core limitation of the system.

Some Key Points:

- Background Subtraction Techniques
- Real-Time Tracking and Counting
- Limitations of Existing Systems

The techniques like Background Subtraction techniques and Real –time tracking and counting takes long time to process this problem, to limit this issue a new system was developed known as Limitation of Existing System and to reduce the processing time. Here, I have used the concept of ORB which provides better extraction features than compared to the speed for detection and to count the appropriate number of passing vehicles through a give recorded video. Basically it is only working to test for recoded video not applied for real time approach as it provide us the accurate result we will apply this system to the real time verification . This will help us in future to create a new algorithm for the surveillance of vehicles in more faster and accurate.

PROBLEM STATEMENT:

In these modern days the use of vehicles has significantly increased, resulting in challenges related to their Planning, monitoring, and control. To overcome this issue, Vehicle tracking is the process of locating a moving vehicle using a camera. Capture vehicle in video sequence from surveillance camera is demanding application to improve tracking performance. This technology is increasing the number of applications such as traffic control, traffic monitoring, traffic flow, security etc. The estimated cost using this technology will be very less. Video and image processing has been used for traffic surveillance, analysis and monitoring of traffic conditions in many cities and urban areas. Various methods for speed estimation are proposed in recent years. All approaches attempt to increase accuracy and decrease cost of hardware implementation. The aim is to build an automatic system that can accurately localise and track the speed of any vehicles that appear in aerial video frames.

REQUIREMENTS:

Example Dataset video:

- A recorded video of passing of number of vehicles in highway were there is very much need of surveillance.

A Software To Develop The Code And Algorithm For Performance:

- Python (version 3.x) using vs code as a platform to run this code.

METHODOLOGY:

The implementation of vehicle detection needs to happen in a different setting. In this proposed system uses pre-recorded video as an input dataset, which is then converted into frames, reference backgrounds are extracted, and moving object detection is carried out. The ORB algorithm is used to for multi-object tracking. In order to establish a correlation between the same object and various video frames, the ORB algorithm takes the features of Detected box and compare it. For the purpose of identifying and counting vehicles, taking the pre-recorded traffic flow videos. When the objective of an object recognition task is to locate and identify objects in videos, ORB is frequently used for a variety of Computer Vision applications, the ORB algorithm is a flexible and effective solution, especially when it comes to real-time processing and feature matching. Utilizing a Gaussian filter to convolve a video in order to reduce noise and detail and smooth it out is a commonly used video processing technique known as "Gaussian blur." When it comes to counting vehicles, Gaussian blur can be used to pre-process the input video and improve the precision of later computer vision tasks, like object detection.

First, to start the process the system needed to be initialized by setting up and initializing. Using a camera, a constant sequence of a video is captured and transferred for processing manually. The screen shot of the input video, which was collected from the Kaggle website. After that, OpenCV library as well as the deep learning mechanism has been employed to subtract the background of the captured video frames. Hence forth at the present stage the vehicles are there without the background. This system uses various methods like image enhancement, and image segmentation to capture the speed and movement of the vehicles from the static cameras. We used different approaches to count the vehicles, i.e. by using a convolution neural network that gives final real time results with high accuracy.

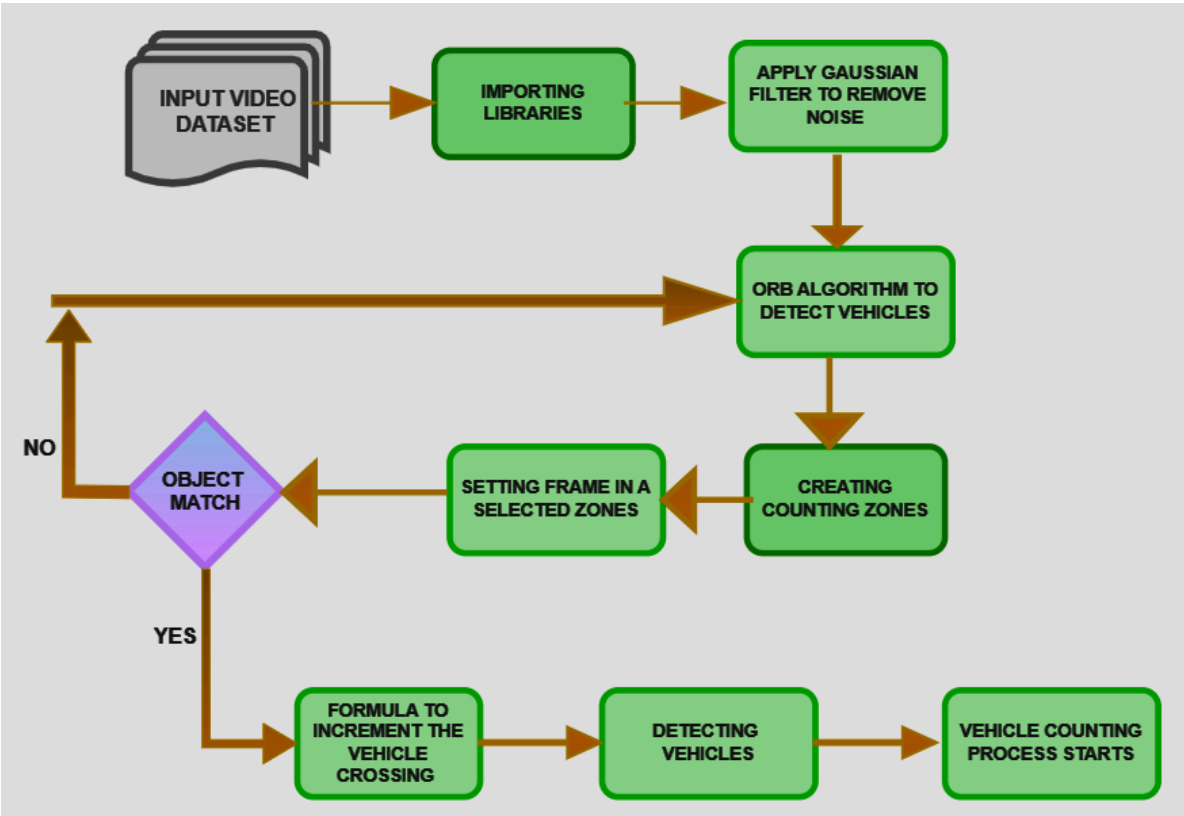

Fig 1: Methodology of vehicle identification and counting using ORB

INPUT VIDEO DATASET:

In order to develop and assess computer vision algorithms for traffic monitoring and analysis, it is imperative to obtain an appropriate surveillance video dataset for vehicle counting. Numerous datasets with a variety of scenarios, camera angles, and environmental conditions are accessible to the public. In this paper, taking a surveillance video to counting the vehicle. The surveillance video includes cars, bikes. In this system counts all the vehicles crossing in the video. It includes the basic transportation infrastructure such as adding more walkways and broader roads. The input video dataset is the more help to check our vehicle counting system.

IMPORTING LIBRARIES:

A popular open-source computer vision processing library, Open-Source Computer Vision Library (OpenCV) used wide range of tools and features for forming computer vision applications. OpenCV can be used for a number of tasks related to vehicle counting, including object detection, tracking, and image processing. Both reading and processing of image and video streams are supported by OpenCV. It has tools for grabbing frames from cameras, video files, and other input sources. OpenCV supports algorithms related to PC Vision and artificial intelligence.

ALGORITHMS USED:

1. GAUSSIAN FILTER:

Gaussian filter used to eradicate unwanted noise in the video streams. When using Gaussian Blur to reduce noise in an image, some frame details are lost in the process. However, even though Gaussian blur lessens detail, it doesn't lower the quality of object detection because moving objects can still be converted to grayscale for far superior outcomes. It utilizes a rectangular pixel group encircling the filtering frame that moves in tandem with the filter being refined. This aids in completing the kernel's missing pixels as well. It completes the task by looping through the entire frame. A common image processing method is the Gaussian filter, which involves convolving an image with a Gaussian kernel to produce effects like feature smoothing, noise reduction, and blurring. The conversion of the RGB image into a grayscale image. The most important part of the first pre-processing stage by which accurate results can be displayed. Basically, the grayscale values are taken from 0 to 255. Background subtraction is the Gaussian mixture-based Background/Foreground Segmentation algorithm. With the help of this approach, it is simple to choose the necessary number of Gaussian mixtures for the pixels. The detection of shadows of the objects is decided by this Algorithm. So, this Algorithm helps in detecting and marking the shadows.

2. ORB ALGORITHM TO DETECT VEHICLES:

The primary uses of the ORB algorithm are feature description and key point detection. It is often used in applications such as image stitching and object recognition. The distinguishing features of images are detected by ORB. The focal points are areas of the picture where texture or intensity changes noticeably. These key points may correlate to particular vehicle features, like corners or distinctive patterns, when it comes to vehicle counting. Following the identification of key points, ORB creates a descriptor for each key point. This descriptor is a condensed representation of the surrounding local image data for the important point. It records the patterns of orientation and intensity, making it possible to match important points between different frames. The OpenCV library consists of the Background SubtractorMOG2 algorithm which is used to apply the background subtraction method. Background subtraction is the Gaussian mixture-based Background/Foreground Segmentation algorithm. With the help of this approach, it is simple to choose the necessary number of Gaussian mixtures for the pixels. The detection of shadows of the objects is decided by this Algorithm. So, this Algorithm helps in detecting and marking the shadows.

3. CREATING COUNTING ZONES:

Determining precise Regions Of Interest (ROIs) within the video frames where need to count the vehicles is the first step in creating counting zones for vehicle counting. To define the counting zones, the drawing functions are used to draw rectangles in the distinct areas of entry and departure. It utilizes ORB algorithm for tracking or vehicle detection
inside the designated counting zones. This could entail models based on machine learning, contour detection, or background subtraction. To attain the appropriate precision and functionality, modify variables like the dimensions or placement of zones. The shape detection and shape recognition of objects is done by Contour which gives the shape guidelines. The accuracy to finding the contour is done by applying Canny edge detection on the binary image. The contour is implemented by using OpenCV which has **cv2.findCountour()command.**

In addition, a motion segmentation process is used to detect vehicles. This process involves analysing and grouping sets of pixels based on the orientation and speed of their movement relative to the background of the scene. By isolating the moving objects in this way, the system can effectively detect and classify vehicles in the video feed.

4. MATCHING FRAMES:

ORB is used to process a series of consecutive frames from a video feed in a vehicle counting application. To find correspondences between frames, the main points and their descriptors are compared. Vehicles are tracked during the video sequence with the aid of this matching process. To distinguish the foreground object from the background frame image, a process called frame differentiation is employed, which involves subtracting two consecutive frames in the image sequence. In addition, a motion segmentation process is used to detect vehicles. The uniform density of the centre of mass of an object is called Centroid.

5. VEHICLE TRACKING:

The algorithm is able to estimate the motion and position of vehicles over time by recognizing and monitoring points that are important to vehicles. The count of the vehicle is increased when the entered in the area of interest depended on the tracking data. After obtaining the bounding box, it must be forced to determine the centroid. It is used to track the movement of the items in the video. The vehicle counting procedure will begin after the bounding boxes have been constructed and the frames match the bounding boxes. In the bounding box, the red dot is placed for counting the vehicles. When the red dot and red line meets its start tracking.

6. **VEHICLE COUNTING:**

When a vehicle enters the allocated area or crosses the line, the counts of vehicles increase one. The counting of vehicle is ensured by reviewing the tracking data, which includes the vehicle's position and direction. The mechanism advances the relevant counter when a vehicle satisfies the counting requirements. By connecting the two ROI points the imaginary line is formed which appears to be diagonal. Vehicle count is increased when the centroid of the vehicle passes the imaginary line in the Region of Interest (ROI). From the vehicle counter is initialized as the variable to store the count of each vehicle when the centroid of the vehicle touches the imaginary line in ROI. Similarly, vehicle counting can be obtained using different machine learning algorithms.

CODE OVERVIEW:

```
import cv2
import numpy as np
cap = cv2.VideoCapture('video.mp4')
min_width_react = 80
min_height_react = 80
count_line_position = 550

# Use createBackgroundSubtractorMOG2

algo = cv2.createBackgroundSubtractorMOG2()
def center_handle(x, y, w, h):
    cx = int(x + w / 2)
    cy = int(y + h / 2)
    return cx, cy

detect = []
offset = 6
counter = 0

while True:
    ret, frame1 = cap.read()
    if not ret:
        break  # Exit if the video has ended

    grey = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(grey, (3, 3), 5)

    img_sub = algo.apply(blur)
    dilat = cv2.dilate(img_sub, np.ones((5, 5)))
    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
    dilatada = cv2.morphologyEx(dilat, cv2.MORPH_CLOSE, kernel)
    dilatada = cv2.morphologyEx(dilatada, cv2.MORPH_CLOSE, kernel)

    # Find contours

    counterShape, _ = cv2.findContours(dilatada, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    # Draw the counting line
```

```
    cv2.line(frame1, (25, count_line_position), (1200, count_line_position), (255, 127, 0), 3)
    for (i, c) in enumerate(counterShape):
        (x, y, w, h) = cv2.boundingRect(c)
        validate_counter = (w >= min_width_react) and (h >= min_height_react)
        if not validate_counter:
            continue

        center = center_handle(x, y, w, h)

 # Check if the center of the vehicle is near the counting line
        if center[1] < (count_line_position + offset) and center[1] > (count_line_position - offset):
            if center not in detect:  # Only count if this vehicle hasn't been counted yet
                counter += 1
                detect.append(center)  # Add to detected list to avoid double counting
                print("Vehicle Counter: " + str(counter))
                # Change the color of the line after counting
                cv2.line(frame1, (25, count_line_position), (1200, count_line_position), (0, 127, 255), 3)

        # Draw rectangle around detected vehicle
        cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
          cv2.putText(frame1, f"Vehicle {counter}", (x, y - 20), cv2.FONT_HERSHEY_TRIPLEX, 1, (255, 244, 0),
2)
        cv2.circle(frame1, center, 4, (0, 0, 255), -1)

    # Display the vehicle count
     cv2.putText(frame1, "VEHICLE COUNTER: " + str(counter), (450, 70), cv2.FONT_HERSHEY_SIMPLEX, 2,
(0, 0, 255), 5)

    # Show the processed frames
    cv2.imshow('Detector', dilatada)
    cv2.imshow('Video Original', frame1)

    if cv2.waitKey(1) == 13:  # Press Enter to exit
        break

# Release resources
cap.release()
cv2.destroyAllWindows()
```

OUTPUT REVIEW:

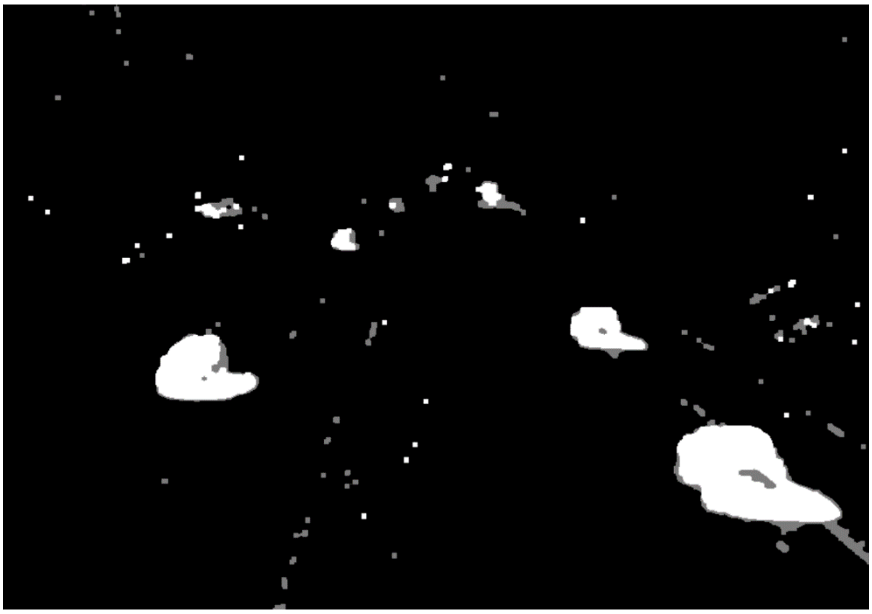

Fig 1: Background Subtraction and Morphological Operation

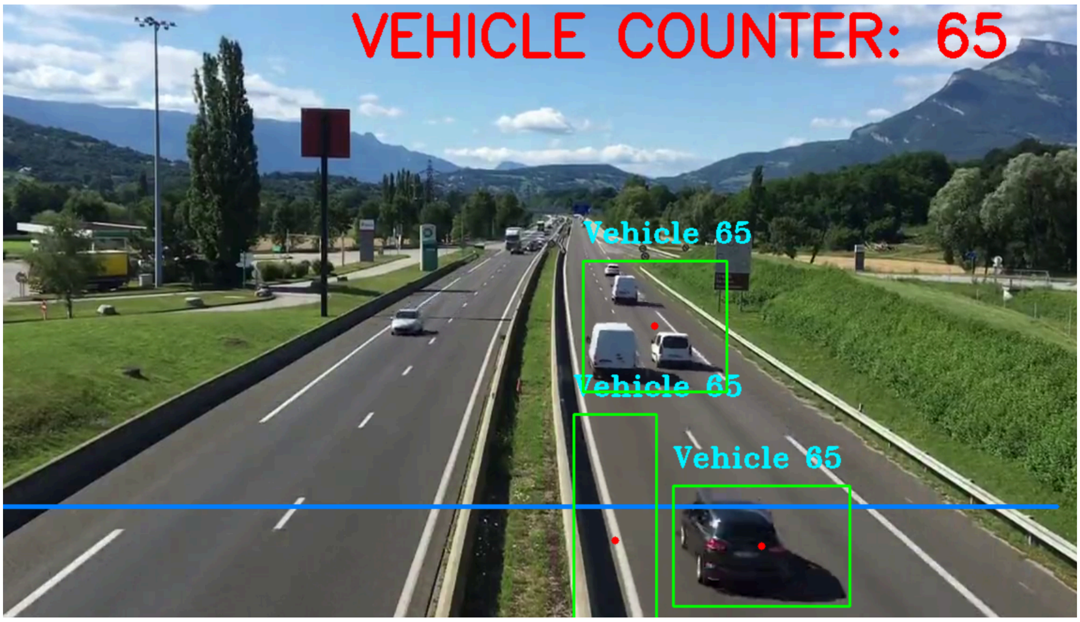

Fig 2: Counting zones of the vehicle counting system and Tracking process of vehicle

```
Vehicle Counter: 1
Vehicle Counter: 2
Vehicle Counter: 3
Vehicle Counter: 4
Vehicle Counter: 5
Vehicle Counter: 6
Vehicle Counter: 7
Vehicle Counter: 8
Vehicle Counter: 9
Vehicle Counter: 10
Vehicle Counter: 11
Vehicle Counter: 12
Vehicle Counter: 13
Vehicle Counter: 14
Vehicle Counter: 15
Vehicle Counter: 16
Vehicle Counter: 17
Vehicle Counter: 18
Vehicle Counter: 19
Vehicle Counter: 20
Vehicle Counter: 21
Vehicle Counter: 22
Vehicle Counter: 23
Vehicle Counter: 24
```

Fig 3: Output of the vehicle counting system

RESULTS AND DISCUSSION:

A one-minute video in the system to demonstrate how the vehicle counting system operates. An ORB (Oriented FAST and Rotated BRIEF) algorithm was employed in this. Ultimately, traffic data is computed. In addition to generating the trajectory and determining the direction in which the vehicle is traveling, object tracking also gathers traffic data, including the quantity of vehicles in each category. The suggested method performs well in terms of accuracy and efficiency when tested against a publically available dataset (Kaggle).
The ORB algorithm provides precise results and accurately sum the number of vehicles where it crosses the frames. It provides excellent accuracy when the wait key is set to 0 and high accuracy when the wait key is set to 100.Utilizing the ORB algorithm, 88 vehicles are counted in one minute by completely consuming the one-minute video to verify the outcome.

CONCLUSION:

This work has provided a thorough explanation of the vehicle detection and counting system that uses OpenCV Python to recognize and count vehicles from traffic surveillance cameras. This paper's major goal is to provide an accurate solution for issues such as being heavily biased in the data set and different types of vehicles having similar sizes and shapes which makes it difficult to classify vehicles and count them. This solution can be used in parking area allocation, traffic monitoring on highways, and traffic control. It is very important to improve the traffic system by using new techniques so that it decreases the road accidents and traffic congestion. In order to detect obstacles and other objects in the lane, this work can be expanded in the future by combining the Haar Cascade classifier with other techniques. Using the extracted objects, an effective way to count vehicles was introduced in the proposed system, where the ORB algorithm is the main component. The methodology used 0 by ORB algorithm to produce results that are precise and highly efficient. The computer vision project is managed using the OpenCV library in the Python programming language. This study offers precise findings based on surveillance footage. Accurate counting also evaluated and a background model was made for a very limited region. Vehicles are interpreted as foreground objects when they travel through this small space. A foreground detector is used to compensate for tiny changes in the real scene when the background is static. The camera's motion is estimated using an image-registration technique, allowing for the detection of a moving vehicle. On top of all that, our framework incorporates an online-learning tracking mechanism that allows us to keep up with changing vehicle shapes and scales in images.

FUTURE WORK:

In vehicle counting, for the possible situation of vehicle missing detection and false detection, vehicle counting based on fusing virtual detection area and vehicle tracking is proposed. Missing alarm suppression module based on vehicle tracking and false alarm suppression module based on bounding box size statistics are designed to avoid vehicle counting errors caused by missing detection or false detection, which further improves the accuracy of vehicle counting. In this framework, the trade-off between accuracy and efficiency can be made according to the requirement by choosing a deep learning object detection model with a suitable performance in accuracy and efficiency as the basic model. Moreover, the proposed framework can improve the accuracy of vehicle counting although the accuracy of vehicle detection is not very high.

All the traffic videos used in this study are shot on straight roads. However, there are other scenarios in traffic surveillance, such as intersections and T-junctions. Although the model in this study has strong performance in straight road scenarios, making the model work well in different scenarios is an important problem to solve. Future work will consider scene adaptation to build a vehicle counting framework for different scenarios.

REFERENCES:

- M.A. Aljamal, H.M. Abdelghaffar, and H.A. Rakha, “Kalman filterbased vehicle count estimation approach using probe data: A multi-lane road case study,” in 2019 IEEE intelligent transportation systems conference (ITSC). 2019.

- https://scholar.google.co.in/scholar?hl=en&as_sdt=0%2C5&as_vis=1&q=+VEHICLE+IDENTIFICATION+AND+COUNTING+OPENCV&btnG=

- K. He, X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

- Kaushal, A. Vehicle Identification GitHub. Retrieved from https://github.com/arpit-kaushal/Vehicle-Detection-and-Counting-using-OpenCV

- M.A. Abdelwahab, “Fast approach for efficient vehicle counting”. Vol:55, Issue:1, Electronics Letters, 2019.

- Karthik Srivathsa D S, Kamalraj R, “Vehicle Detection and Counting of a Vehicle Using OpenCV”, In the International Research Journal of Modernization in Engineering Technology and Science,Vol:03, Issue:05, 2021.

- GeeksforGeeks. (26 Jun, 2024),Vehicle detection using OpenCV Python Retrieved from https://www.geeksforgeeks.org/opencv-python-program-vehicle-detection-video-frame/