2CSOE78 - SCIENTIFIC PROGRAMMING

Special Assignment (Odd 2022-23)

TOPIC: TRAFFIC MONITORING SYSTEM

Submitted By:

Mukta Shah (20BEC072)

Maitri Patel (20BEC085)

Devika Pedada(20BEC089)

Under the guidance of:

Prof. Usha Patel

Semester-V

Institute of Technology

Electronics and Communication Department

TABLE OF CONTENT

1. ABSTRACT ……. 3
2. PROBLEM STATEMENT …….. 3
3. FUNCTIONS, LIBRARY AND ALGORITHM USED ……. 3
3.1 OPEN CV……... 3
3.2 BACKGROUND SUBTRACTOR ……... 4
4. PROJECT MODEL ……. 4
4.1 OBJECT DETECTION ……. 4
4.2 OBJECT TACKING AND COUNTING OF VEHICLES ……... 6
4.3 SPEED ESTIMATION …….. 7
5. FLOW CHART ……. 7
6. OUTPUT SCREENSHOT ……... 8
7. RESULTS AND ANALYSIS …….. 8
8. CHALLENGES FACED ……. 8
9. FUTURE SCOPE …….. 9
10. CONCLUSION …….. 9
11. REFERENCES …… 9

1. ABSTRACT

Due to the increase in population, the number of vehicles on the road has increased. At traffic signals there is always a high chance of accidents due to breaking traffic rules by drivers and pedestrians. Therefore, to ensure safety on the road it is important to manage it well and continuously monitor it. However, managing and monitoring traffic is very difficult and it requires human effort. It is difficult and strenuous for a human to monitor how many cars crossed a particular street in a day?

One solution of these problems is to create an automated system that is capable of performing all above mentioned tasks. So, in this project, we have attempted to make a Traffic Monitoring System that detects and counts vehicles on both sides of the road and also measures their speed. The entire project is designed using Image Processing in Python with help of certain libraries of python such as NumPy, Open CV etc.

2. PROBLEM STATEMENT

The current road surveillance cameras and their viewing software can detect and track vehicles in the traffic and is also used for obtaining number plates and identification of drivers. Through this project, we will be able to successfully detect as well as further monitor the traffic according to the number of vehicles as well as estimating its speed.

The objective of this project is to create a traffic radar using Image Processing in Python by using OpenCV. When it comes to tracking the speed of vehicles on a segment of road, the vital steps of this projects is:

- Vehicle Detection
- Vehicle Tracking
- Speed estimation

3. FUNCTIONS, LIBRARY AND ALGORITHM USED

3.1 Library Used - OPEN CV:

- OpenCV is a huge open-source library for computer vision, machine learning, and image processing. It supports a wide variety of programming languages like Python, C++, Java, etc. It is one of the most widely used tools for computer vision and image processing tasks.
- The library comes with more than 2,500 improved aIgorithms, that consists of a wideranging set of both state-of-the-art and cIassic machine learning algorithms and computer vision.
- In our project we have made use of OpenCV for –
    - Reading and writing videos, images.
    - Classify objects
    - ldentify human actions in videos
    - Tracking of moving objects
    - Background subtraction.

3.2 Algorithm Used - Background Subtractor

- Background subtraction is a major pre-processing steps in many vision based applications. Technically, it does the job of extracting the moving foreground from static background. Several algorithms were introduced for this purpose. OpenCV has implemented three such algorithms which is very easy to use. Out of these, we have made the use of Background Subtractor MOG

BackgroundSubtractorMOG

- It is a Gaussian Mixture-based Background/Foreground Segmentation Algorithm. It uses a method to model each background pixel by a mixture of K Gaussian distributions (K = 3 to 5). The weights of the mixture represent the time proportions that those colours stay in the scene. The probable background colours are the ones which stay longer and more static.
- While coding, we create a background object using the function, **cv2.createBackgroundSubtractorMOG()**. It has some optional parameters like length of history, number of gaussian mixtures, threshold etc. It is all set to default values. Then inside the video loop, we use *backgroundsubtractor.apply()* method to get the foreground mask.

4. PROJECT MODEL

The Traffic Mointoring System consists of two major blocks - the Image Processing and Decision Block. The Image Processing and Decision Block consists of the subsequent tasks in sequence: -

1. Acquires video of the traffic on both the lanes of the road.
2. Enhances these acquired video and removes any noise/disturbance present in them using algorithms.
3. Processes the frames to count number of vehicles on each lane at that instant of your time.
4. Finally, using vehicle count determines the no of vehicles in each lane. And then using thresholds determine the speed of the vehicles.

Further about these topics are discussed below:

4.1 Object Detection

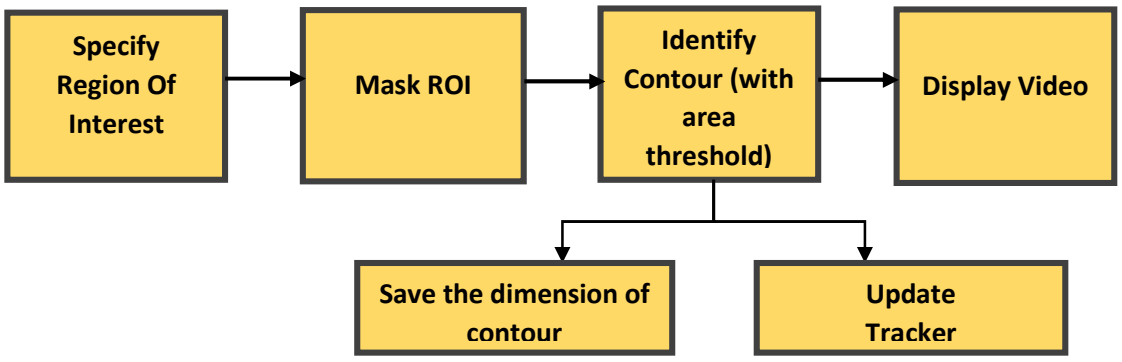

Figure 1. Block Diagram for Object Detection

- The Traffic Mointoring System performs the main functionality of determining the number of vehicles. As the background of the vehicles is stationary (i.e., as the speed camera is stationary) image subtraction is used to detect moving vehicle.
- OpenCV firstly converts the input image from BGR to grayscale, after which the picture undergoes masking, i.e. useless and redundant data is removed using suitable coordinates. After this morphological operations are performed. (Figure 2(a-c))

```
grey=cv2.cvtColor(f[i],cv2.COLOR_BGR2GRAY)
blur=cv2.GaussianBlur(grey,(3,3),5)
```

- Morphological operations are simple transformations applied to binary or grayscale images. More specifically, we apply morphological operations to shapes and structures inside of images.
- Morphological operations "probe" an image with a structuring element. This structuring element defines the neighbourhood to be examined around each pixel. And based on the given operation and the size of the structuring element we are able to adjust our output image.
- So first using Background Subtractor MOG we remove background (black) and differentiate object(white). (Figure 2(d))

```
bgsub= wb.apply(blur)
```

- Then we perform dilation, i.e., we increases the white region in the image or the size of the foreground object increases. This helps in outlining and defining each distinct object seen in the image. (Figure 2(e))

```
dilat=cv2.dilate(bgsub,np.ones((5,5)))
```

- The dataset then undergoes a grid search where most efficient gamma values are retrieved as the output. Kernel uses the most efficient gamma values to get a circular structuring element.

```
kernel=cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
```

- Then we perform close operation. A closing operation is a dilation followed by erosion. It is used to close holes inside of objects or for connecting components together. ((Figure 2(f)))

```
close=cv2.morphologyEx(dilat,cv2.MORPH_CLOSE,kernel)
close=cv2.morphologyEx(close,cv2.MORPH_CLOSE,kernel)
```

- Counter is used for shape analysis and object detection and recognition. It is simply a curve joining all the continuous points (along the boundary), having same color or intensity.

```
cs[i],h =cv2.findContours(close,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
#Syntax:
#contours, hierarchy = cv2.findcontours(img, mode, method)
#      mode – contour retrieval mode
#          cv2.RETR_TREE – retrieves only the extreme outer contours
#      method – contour approximation method
#          cv2.CHAIN_APPROX_SIMPLE – stores only the corner points
```

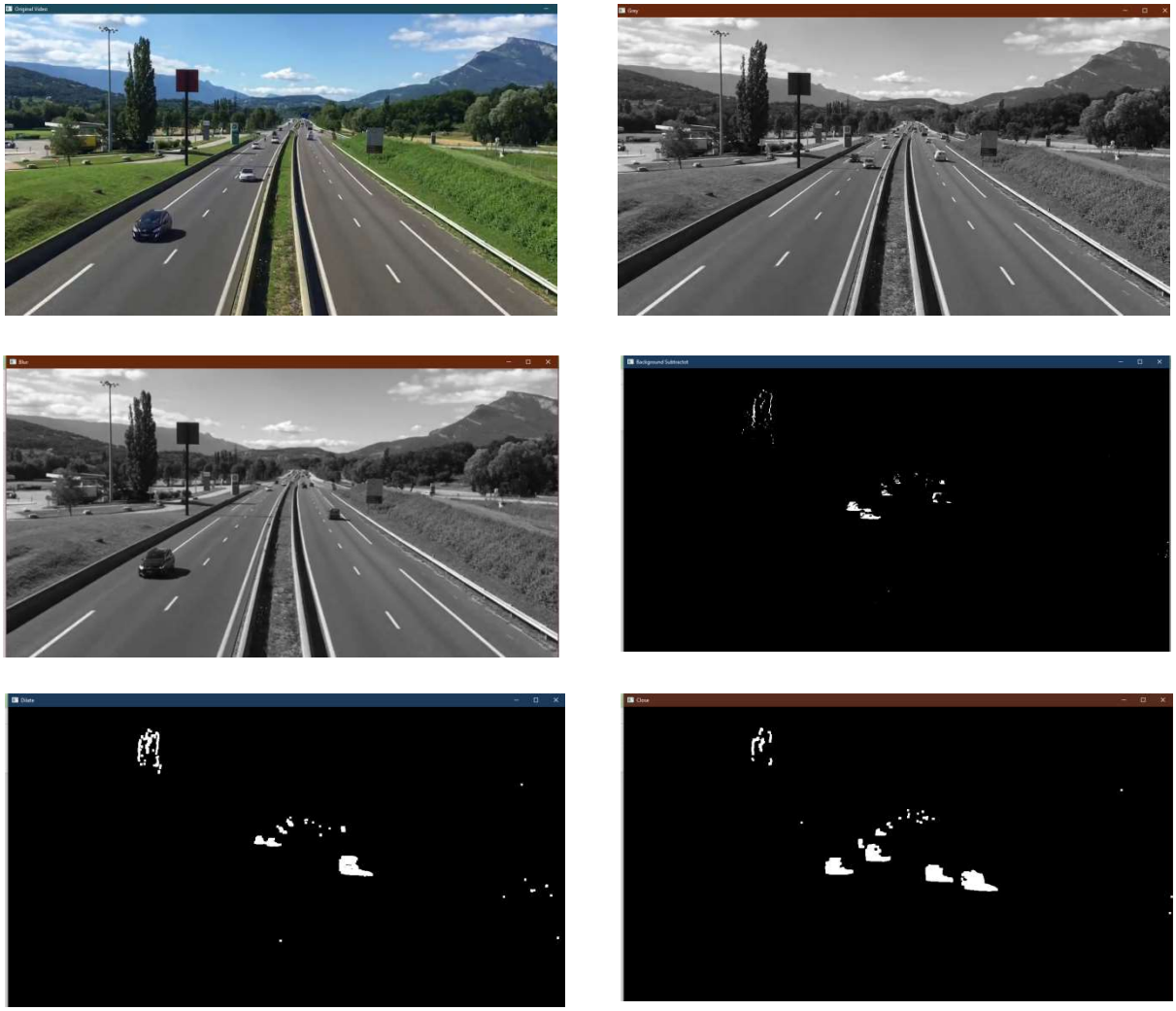

Figure 2. (Left to Right) (a) Original Frame (b) Greyscale image (c) Blurred image (d) Background Subtraction performed (e) Dilation Operation Performed (f) Closing Operation Performed

4.2 Object Tracking and Counting of Vehicles

- Based on the area threshold of number of pixels, the contours are detected.
- Then we draw rectangles around detected objects. *cv2.boundingRect()* is used to draw an approximate rectangle around the binary image. This function is used mainly to highlight the region of interest after obtaining contours from an image.

```
(x,y,w,h)=cv2.boundingRect(c)
```

- The threshold is used to avoid detecting contours of smaller moving objects that are not vehicles.

```
validate_counter=((w>=min_width_rect) and (h>=min_ht_rect))
 if not validate_counter:
          continue
```

- The object is tracked based on the distance between two contours between frames. An ID is assigned to each contour.
- According to the decided threshold we increase our counters separately (but each vehicle has an unique ID)

4.3 Speed Estimation

- Time difference between the position of a vehicle is calculated and the speed is estimated based on a formula.
- The timer starts when the vehicle crosses the first line, and the timer ends when the vehicle crosses the second line.
- The speed is displayed on top of the bounding box only when the vehicle crosses both the lines.

5. Flow Chart

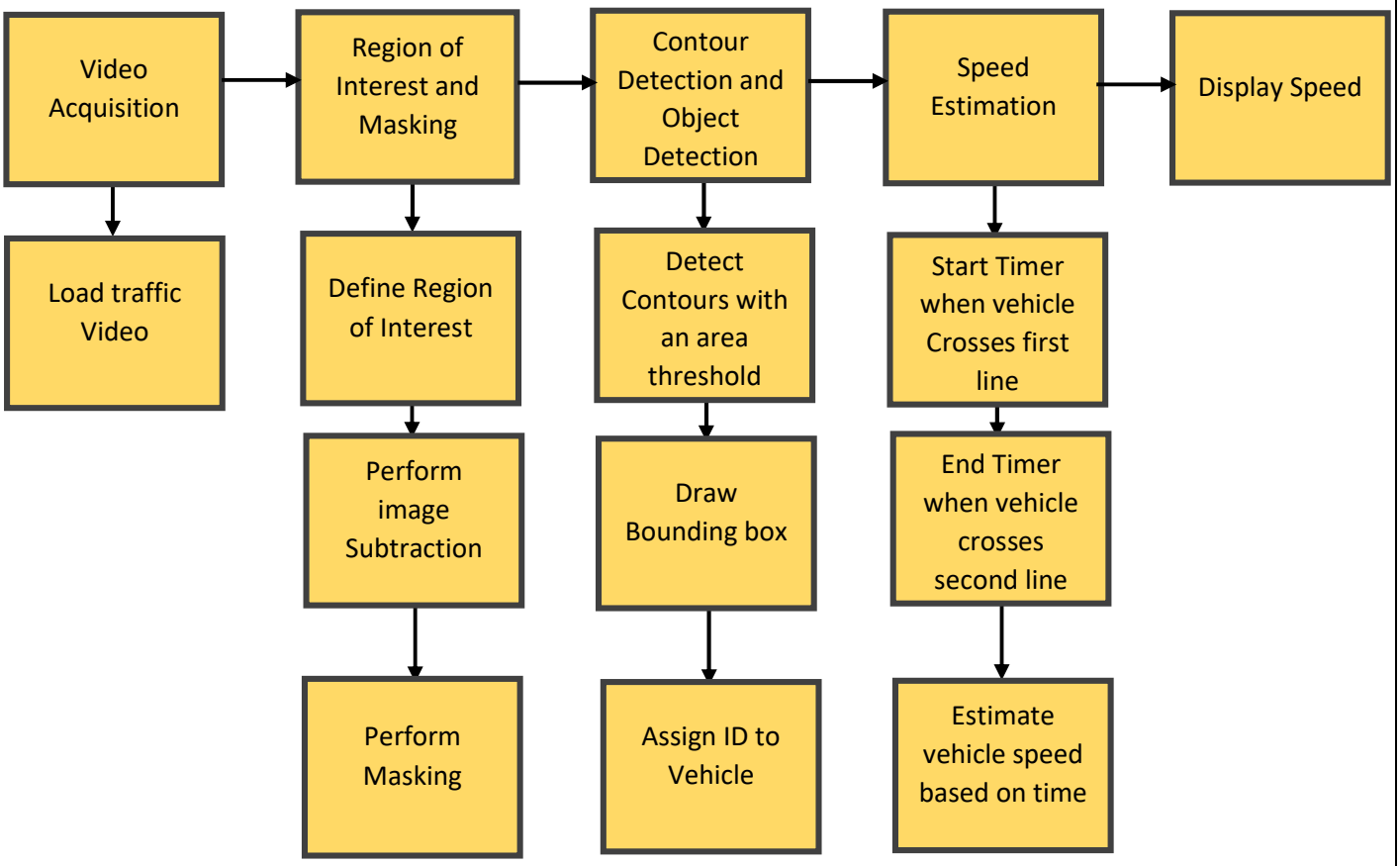

Figure 3. Flow Chart of Traffic Monitoring System

6. Output Screenshots

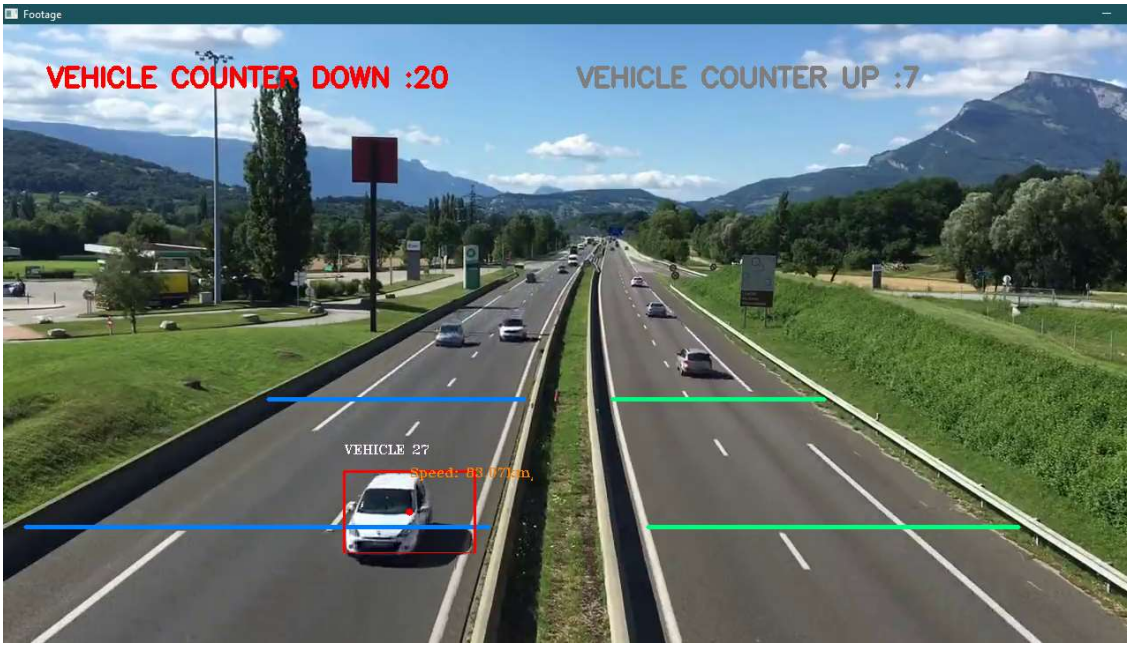

Figure 4. Final Output of the Project

7. Result And Analysis

- This project is successfully able to track vehicles and estimate their approximate speed.
- Accuracy in detecting vehicles is high as long as there is no movement in the camera.
- In his project, we have obtained approximate value of speed. Initially we are getting the speed in pixels per second unit, but later on we have normalised it to have an approximate speed in km/hr.
- Multiple vehicles can be detected. However if two vehicles are moving extremely close to each other, it may be detected as a single object.
- This project requires the camera to be as still as possible, as movement is used to distinguish vehicles from the background.

8. Challenges Faced

- Vehicles moving close to each other are detected as one object.
- The speed is approximate, as the length of the road (i.e., the dimensions of region of interest in real life) present in the video are not known to us.
- Sometimes shadows are also getting detected.
- Not being able to detect the licence plate because of the quality of the video

9. Future Scope

- The objects can be differentiated as cars, trucks or bikes based on their area covered.
- License plate detection can be performed, to give them accurate IDS
- More data about the location can be obtained to have an accurate speed estimation
- The data about the vehicles can be stored in a database for future reference.
- The stored data of the number of vehicles can be used to estimate the traffic density at a specific time of day, which can be useful to construct an adaptive traffic management system.

10. Conclusion

Road safety is an important factor for the police force. And as citizens it is our responsibility to follow rules and maintain safety on our roads. With the ever-growing population it is of top importance that an operative traffic management system which is effective be introduced. The Traffic Monitoring System that has been proposed in this project is one such solution. This project is able to estimate speed of vehicles and count the number of vehicles.

The presented system primarily focuses on the cars. Based on image processing and fixed breaks of time the proposed system can be used in the future to work towards reducing traffic overcrowding, to decrease the probabilities of traffic jams and to provide clearance of road for the vehicles to an extent.

11. References

1. https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html

2. https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_video/py_bg_subtraction/py_bg_subtraction.html#background-subtraction

3. https://www.slideshare.net/alivxlvie/smart-traffic-monitoring-system-report

4. https://www.youtube.com/watch?v=BURNRHK_r9g