National Education Society ®

JNN College of Engineering

Sawalanga Road, Navule, Shivamogga -277204

(Accredited by NBA, NAAC 'B' & Certified by UGC 2f & 12B)

Department of Electronics & Telecommunication Engineering

PROJECT PHASE 2 SEMINAR

"TRAFFIC RULES VIOLATION RECOGNITION FOR TWO-WHEELER USING DEEP LEARNING"

Presented by:

Kavya T N 4JN17TE017

Pooja H 4JN17TE029

Sneha Mohan Hebbar 4JN17TE017

Sushma S 4JN17TE047

Guide:

Ms. Rashmi M Hullamani

Assistant Professor

Contents :

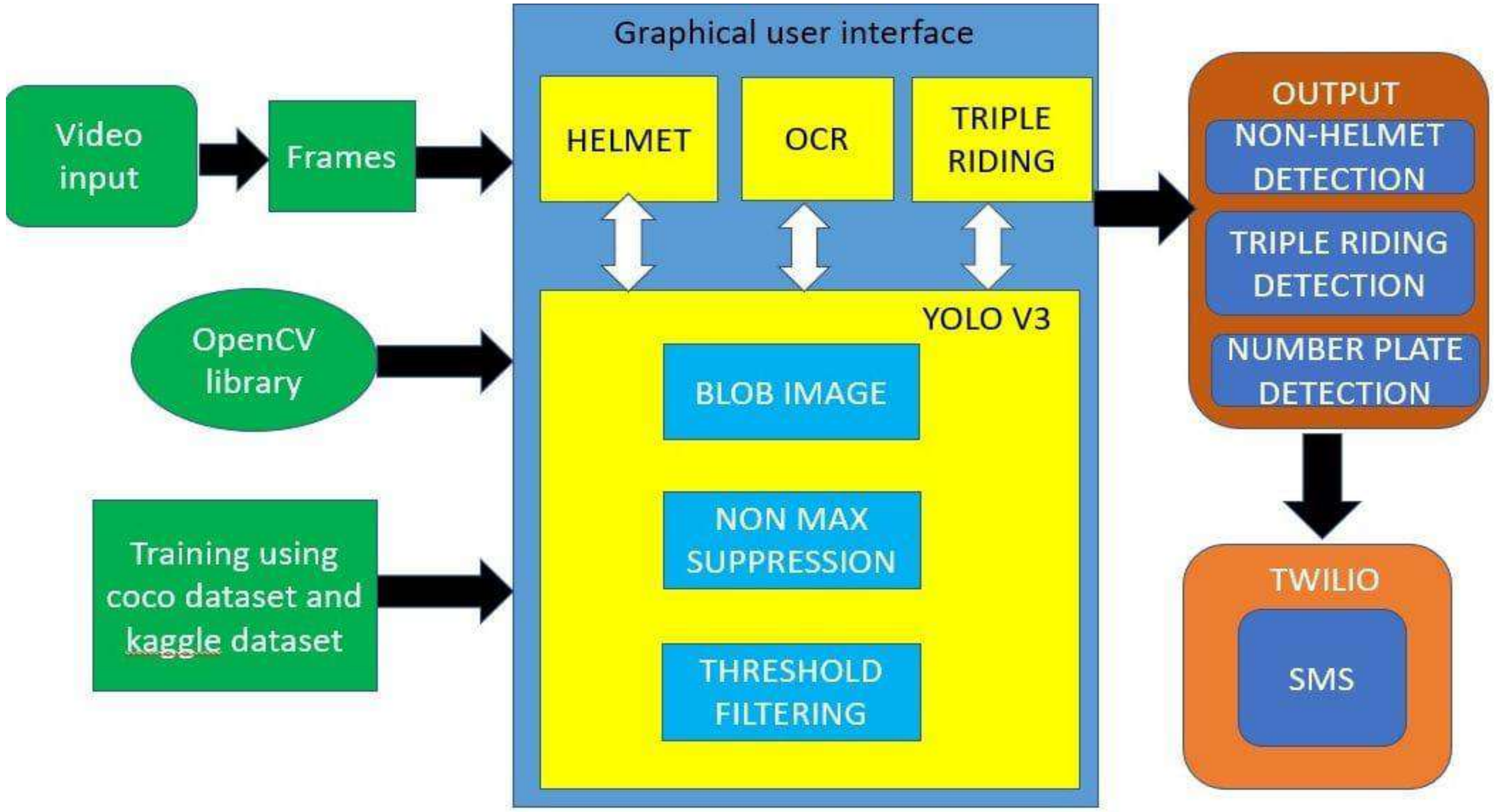
Aim of the project :

- We usually prefer motorbikes over other vehicles as it is significantly less expensive to run, less demanding to park and adaptable in rush hour gridlock.
- In India, in excess of 118 million individuals are utilizing bikes. Since lot of bikes travel along with other vehicles wearing headgear is critical to decrease the danger of injuries.
- we propose an approach where we identify the motorcycle riders without headgear, who are triple riding and who are using mobile phones while driving by utilizing surveillance videos in real-time.
- Our approach also proposes a system where a message will be sent to the concerned authority about the vehicle details of the person who violates the above-mentioned rules with the proof.

INTRODUCTION :

- Bike is an extremely mainstream method of transportation in India. However, there is a high risk involved due to lack of protection.
- To decrease the involved risk, it is highly desirable for motorcycle riders to use helmet. That’s why the government has made it a punishable offense to ride a bike without helmet.
- The drawback of the current method where human intervention is required can be solved by our proposed method.
- Nowadays eveyone is moving towards automation and many countries have implemented automatic traffic surveillance sytem.
- Here we are developing a system in which we are using a more efficient way which helps us in getting better results.

Block Diagram :



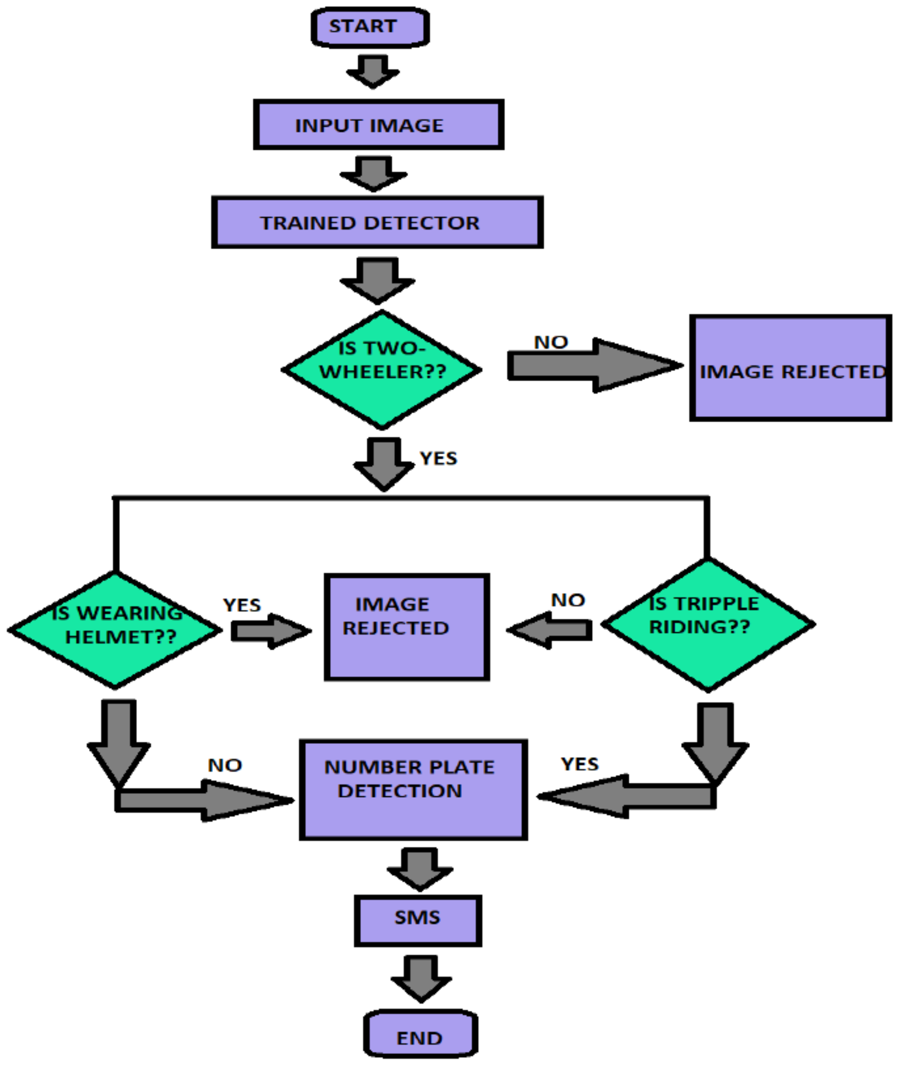
Objectives :

The main objective of the project work is to develop a system for

- Detection of persons on the motorbikes who are not wearing the helmet and triple riding while crossing the signals and junctions .
- Sending an SMS to the concerned authority about the vehicle details of the person who violates the above-mentioned rule.
- To improvise the metric(accuracy,speed and efficiency) to yield better results

Design :

Flow Diagram

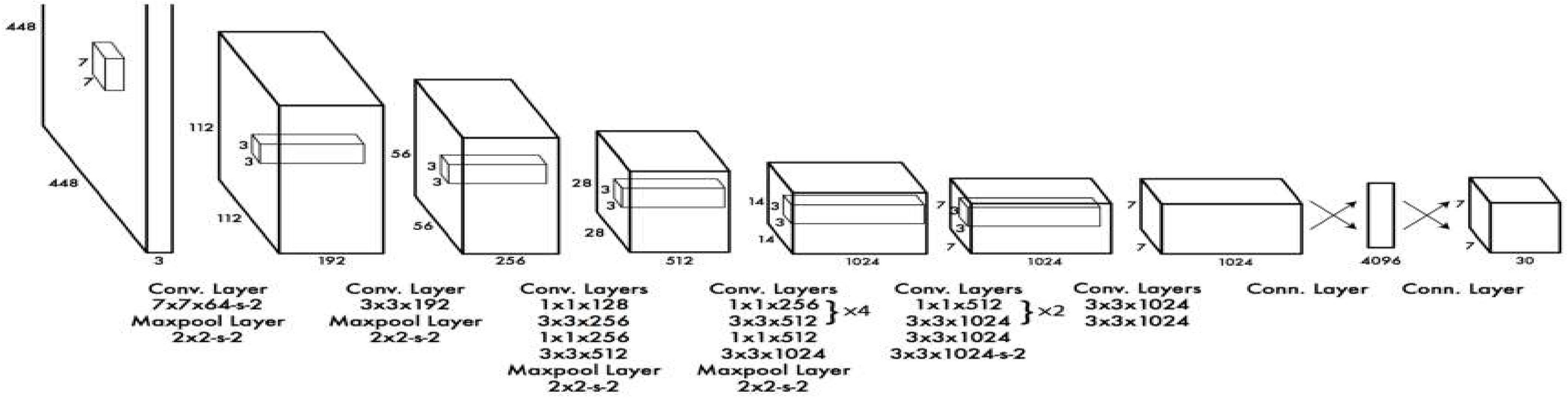


STEP1 - A real time input video is fed to the model.
STEP2 - The model is previously trained with a dataset. Then the real time input is compared with dataset.
STEP3 - If the two-wheeler is detected in the input the process continues else the image is discarded.
STEP4 - After two-wheeler is detected, the pretrained model detects the rider wearing helmet and triple riding. If the rules are violated it goes to the next step else the image is discarded.
STEP5 - When the rules are violated an SMS alert will be sent to the concerned authority.

Algorithm :

THE YOLO MODEL

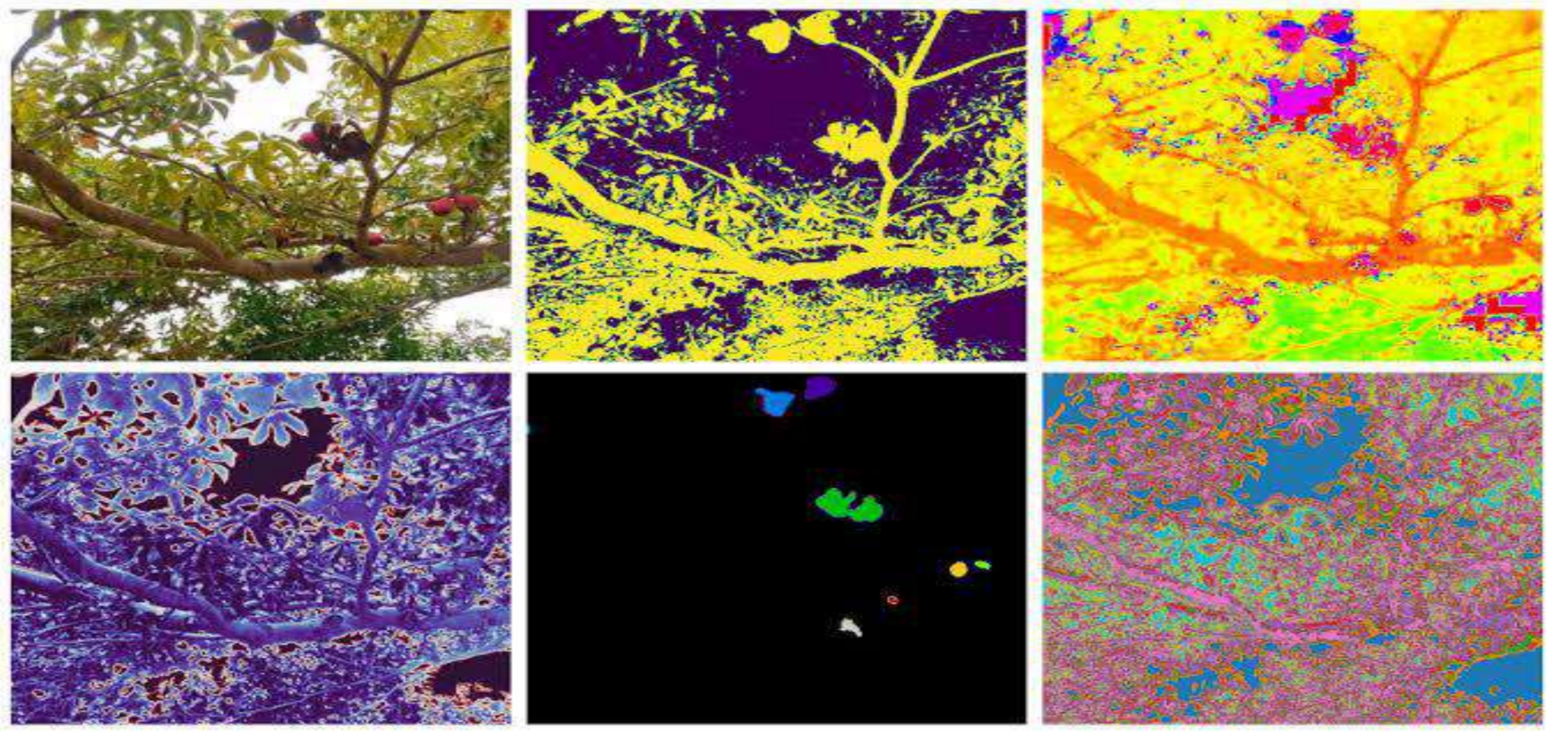
- YOLO calculates the probabilities and bounding boxes for objects in a single forward pass. The way this is done is splitting the image into grids and detecting objects for each of the boxes.
- However, this is not done sequentially, and is actually implemented as a convolution operation.
- For example: if
- Input image size is (448,448,3)
- Number of classes to predict = 25   and Grid size = (7,7)
- Then, each label vector will be of length 30 (Pc, Bx, By, Bw, Bh, +25 classes).



448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

BLOB IMAGE

- Input Video will be converted into frames and the number of frames per second depends on the processor.
- YOLO architecture needs blob images and hence normal images are converted into blob images for processing.
- When compared to the phase 1 now we are able to get better efficiency because we used YOLO V3 algorithm.

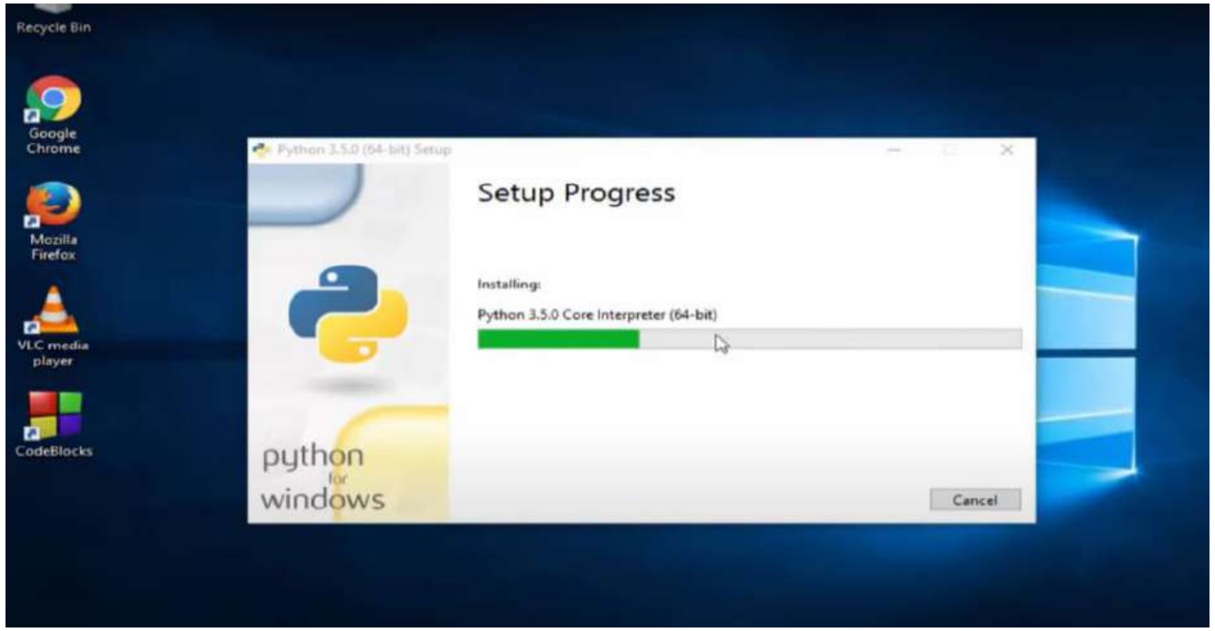


Software implementation :

Python

- Python is a widely used general-purpose, high-level programming language. Its syntax allows the programmers to express concepts in fewer lines of code when compared with other languages like C, C++ or java.

- To get started installing Python, visit Python.org and downloaded the version 3.8.3. windows x86-64 executable installer by setting correct path. Then install the libraries pandas, requests, pillow, and OpenCV in command prompt by giving the command pip install.



Setup Progress

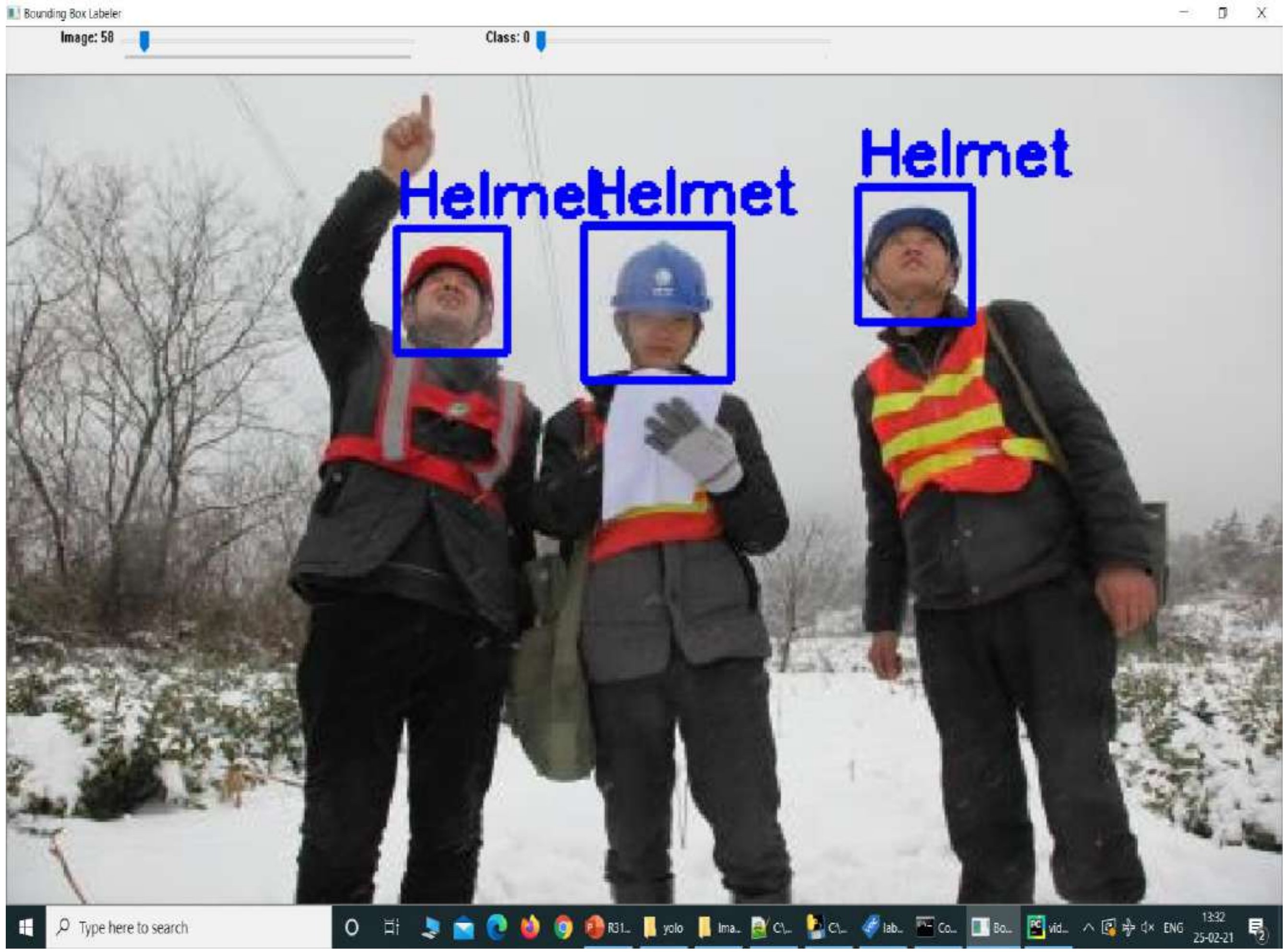
Installing:

Python 3.5.0 Core Interpreter (64-bit)

Fig: Python Installing

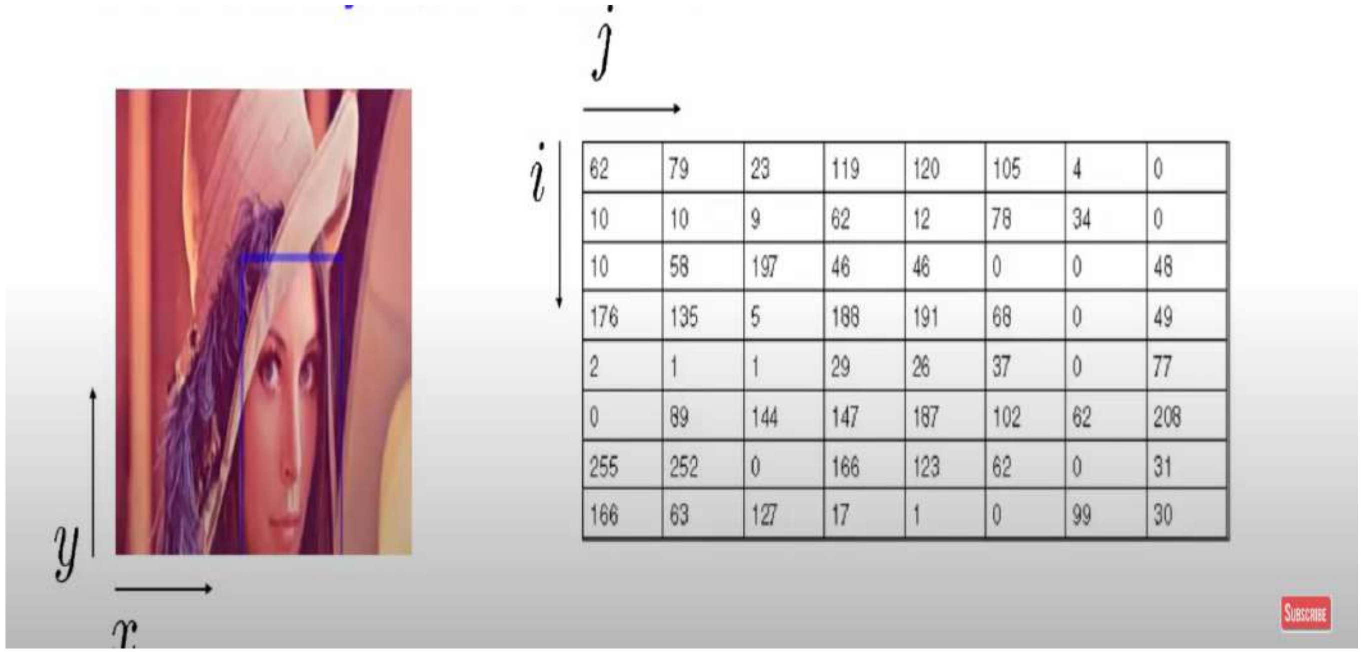
Open CV :

- Open CV is an image processing library created by Intel and later supported by Willow Garage and now maintained by Itseez.
- Available on Mac, Windows, Linux.
- Works in C, C++ and Pyhton.
- Open Source and free.
- Easy to use and install.

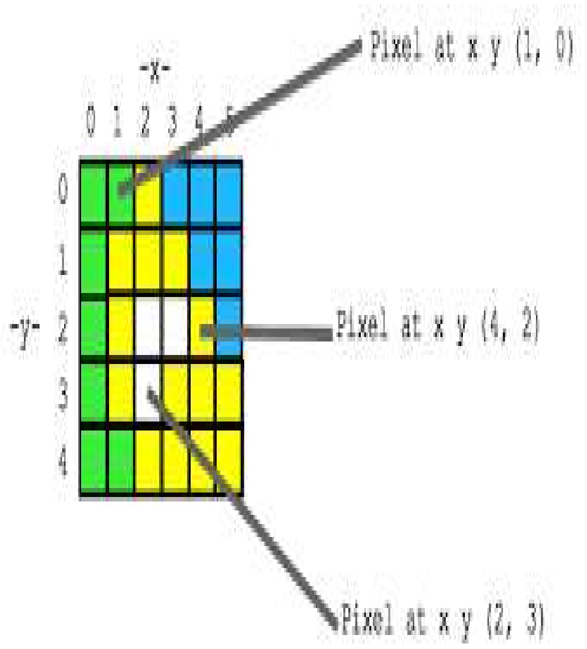


Digital Images

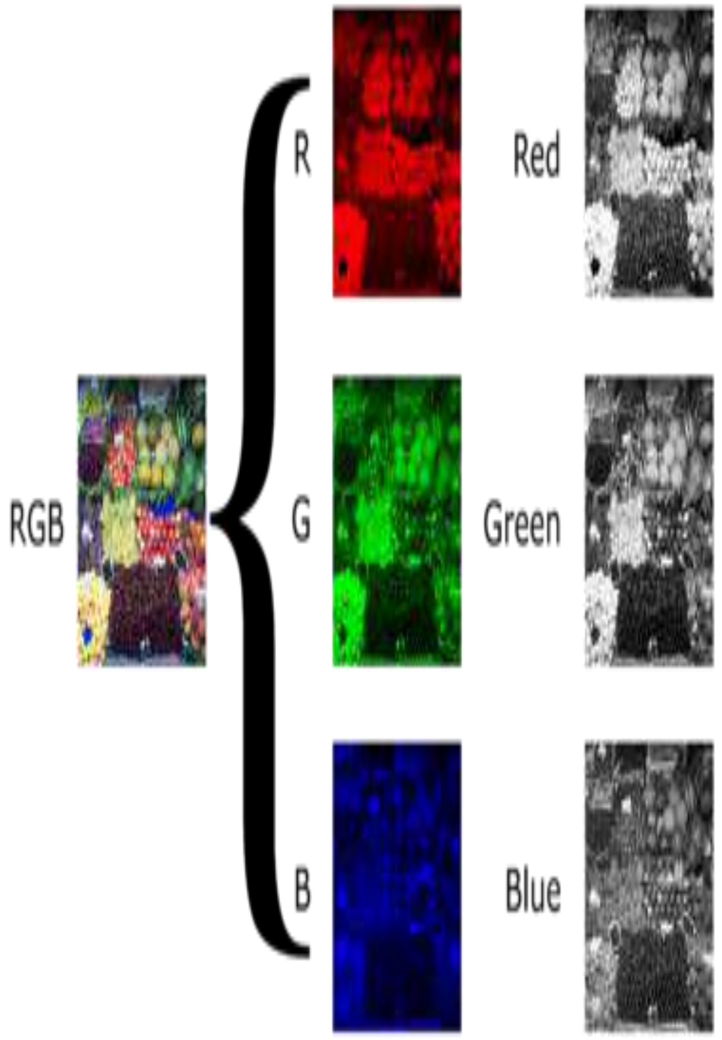
- Digital images are typically stored in matrix.
- There are many different file formats.



62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30



How do computers see images?



There are two types digital images

1. **Grayscale Images**
2. **Colored Images**

1.Grayscale Images:

In this each pixel represents the intensity of only one shade that is how bright or dark the pixel is. In other word it is said that it has only one channel.

2. Colored Images:

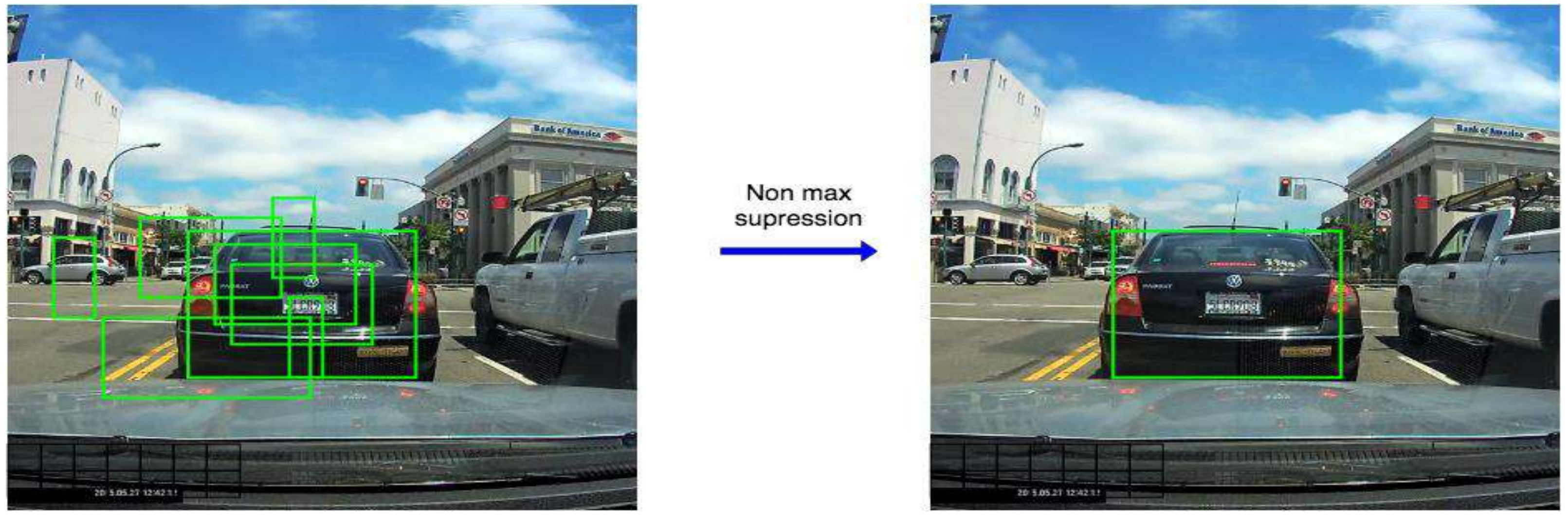
In this we have three channels they are R, G, B (Red, Green, Blue)

Numpy :

- Numpy is a highly optimized library for numerical operations.
- Array Structure is important because digital images are 2D arrays of PIXELS.
- All the OpenCV array structures are converted to numpy arrays.
- We can use more convenient indexing system rateher than using for loops.

Non-max suppression and threshold filtering

- Non-max suppression makes use of a concept called “intersection over union” or IoU.
- The first step, quite naturally, is to get rid of all the boxes which have a low probability of an object being detected.
- It takes as input two boxes, and the name implies, calculates the ratio of the intersection and union of the two boxes.



OCR

- Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo or from subtitle text super imposed on an image.

COCO (Common Objects in Context)

This program uses the COCO (Common Objects in Context) class list, which has 25 object categories.

When the detected images matches the objects list in coco class, the object number gets incremented by 1 in the program.

Graphical user interface:

- A GUI (graphical user interface) is a system of interactive visual components for computer software.
- A GUI displays objects that convey information, and represent actions that can be taken by the user.

Tkinter

- It is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications.
- We create buttons in the GUI page for each applications.

Source code :

- **Install Python IDE of version 3.8.3 on Windows 10 +**

- First, import the all the messages from tkinter

```
from tkinter import *

from tkinter import messagebox

top = Tk()
```

Declare a variables and initialize the color, height, width and size

```
C = Canvas(top, bg="blue", height=250, width=300)

filename = PhotoImage(file = "resized_test.png")

background_label = Label(top, image=filename)

background_label.place(x=0, y=0, relwidth=1, relheight=1)
```

Import window from tkinter and assign the title, geometry and configuration.

```
from tkinter import *
window=top
root=window
window.title("project1")
window.geometry("3500x900")
window.configure(background="#E4287C")
```

Labelling will be done by declaring a variable called lbl and labelling the name with color, size, width, height of rectangle for triplericing.

```
lbl=Label(window, text="Triple Ride and Helmet Detection" ,fg="black"  ,width=25  ,height=1,font=('times', 30, 'italic bold underline'))
lbl.place(x=400,y=10)
```

Then import os and define time()

```
import os
def time():
    os.startfile("yolo_detection_webcam1.py")
def time1():
    os.startfile("Helmet_detection_YOLOV3.py")
def time2():
    os.startfile("yolo_detection_images.py")
def time3():
    os.startfile("yolo_detection_webcam.py")      #start python file
def time4():
    os.startfile("helmet.py")
def time5():
    os.startfile("live1.py")
def time6():
    os.startfile("yolo_detection_images4.py")
def time8():
```

Import the Python libraries, and declare threshold filtering and NMS (Non Maxima Suppression) values

```
        import numpy as np
        import cv2                                              #importing libraries
        import cv2 as cv

        confidenceThreshold = 0.0
        NMSThreshold = 0.6                                      #ranging of pixel
```

Take the dataset inputs from coco and other dataset files

```
        modelConfiguration = 'cfg/yolov3.cfg'
        modelWeights = 'yolov3.weights'

        labelsPath = 'coco.names'
labels = open(labelsPath).read().strip().split('\n')
```

Then VideoCapture() will help to capture the input video

```
video_capture = cv2.VideoCapture("om1.mp4")

(W, H) = (None, None)
count = 0
```

writing the while loop and if loop for repeating the process of capturing video

```
while True:
    ret, frame = video_capture.read()
    fram1=frame
    #frame = cv2.flip(frame, 1)
    if W is None or H is None:
        (H,W) = frame.shape[:2]
```

Non Maxima Suppression will be done in this part of the code.

```
        #Apply Non Maxima Suppression


        detectionNMS = cv2.dnn.NMSBoxes(boxes, confidences, confidenceThreshold, NMSThreshold)
        if(len(detectionNMS) > 0):
          for i in detectionNMS.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])
(w, h) = (boxes[i][2], boxes[i][3])


            color = [int(c) for c in COLORS[classIDs[i]]]
 print(labels[classIDs[i]])
            if labels[classIDs[i]]=="motorbike": #since my detector only has 1 class
              cv2.imwrite("tripleride//framet%d.jpg" % count, frame[y-200:y+h, x-20:x+w])
```

Finally when video capture is over, release the video capture and destroyAllWindows

```
        video_capture.release()
        cv2.destroyAllWindows()
```

Further declarations and assigning values are required for the video and image input and output frames.

```
    except:
        print("completed video")

def time9():
    os.startfile("yolo_detection_images6.py")
```

```
def time7():
    import requests
    import base64
    import json
    from glob import glob
    import pandas as pd
    import time
    import os
    def ocr(IMAGE_PATH):
        SECRET_KEY = 'sk_fa7d3dcec0363bdfb6ac3e06'
        with open(IMAGE_PATH, 'rb') as image_file:
            img_base64 = base64.b64encode(image_file.read())
```

```
 l=set(l)
    print(l)
    for text in l:
        raw_data = {'date':[time.asctime( time.localtime(time.time()))],'':[text]}
        #raw_data = [time.asctime( time.localtime(time.time()))],[text]
df = pd.DataFrame(raw_data)
        df.to_csv('data.csv',mode='a')
    os.startfile('data.csv')
btn1=Button(window, text="Helmet input Video", command=time  ,fg="blue"  ,bg="orange"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))
btn1.place(x=0,y=150)
```

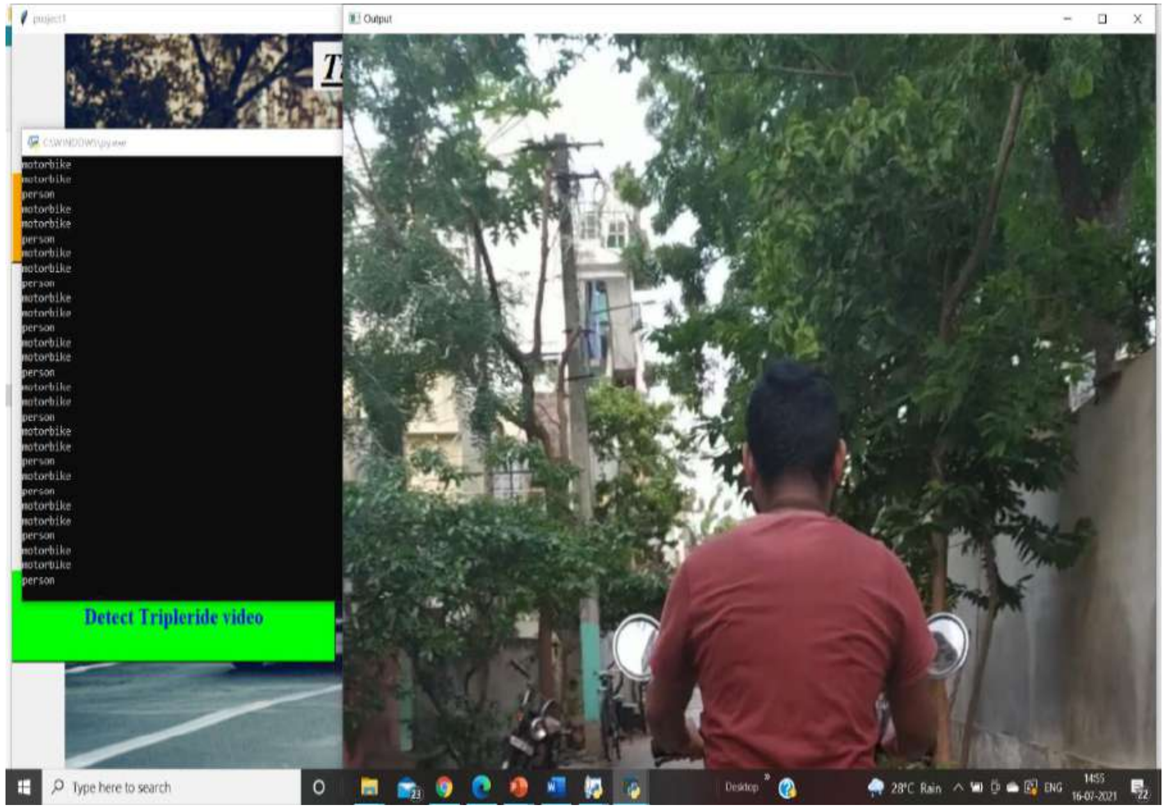
```
btn1=Button(window, text="Detect Persons", command=time2  ,fg="blue"  ,bg="orange"  ,width=30  ,height=3 ,activebackground
= "white" ,font=('times', 18, ' bold '))

btn1.place(x=500,y=150)


btn1=Button(window, text="Detect Helmet", command=time1  ,fg="blue"  ,bg="orange"  ,width=30  ,height=3 ,activebackground
= "white" ,font=('times', 18, ' bold '))

btn1.place(x=980,y=150)


btn1=Button(window, text="TripleRide input Video", command=time8  ,fg="blue"  ,bg="White"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))

btn1.place(x=150,y=280)


btn1=Button(window, text="Detect Persons", command=time9  ,fg="blue"  ,bg="White"  ,width=30  ,height=3 ,activebackground
= "white" ,font=('times', 18, ' bold '))

btn1.place(x=900,y=280)
```

```
btn1=Button(window, text="Detect NumberPlate", command=time6  ,fg="blue"  ,bg="White"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))
btn1.place(x=150,y=450)

btn1=Button(window, text="Detect Text NumberPlate", command=time7  ,fg="blue"  ,bg="White"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))
btn1.place(x=900,y=450)

btn1=Button(window, text="Detect Tripleride video", command=time3  ,fg="blue"  ,bg="#00ff00"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))
btn1.place(x=0,y=580)
```

```
btn1=Button(window, text="Detect helmet video", command=time4  ,fg="blue"  ,bg="#00ff00"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))

btn1.place(x=500,y=580)


btn1=Button(window, text="traffic Detection", command=time5  ,fg="blue"  ,bg="#00ff00"  ,width=30  ,height=3
,activebackground = "white" ,font=('times', 18, ' bold '))

btn1.place(x=980,y=580)


window.mainloop()
```

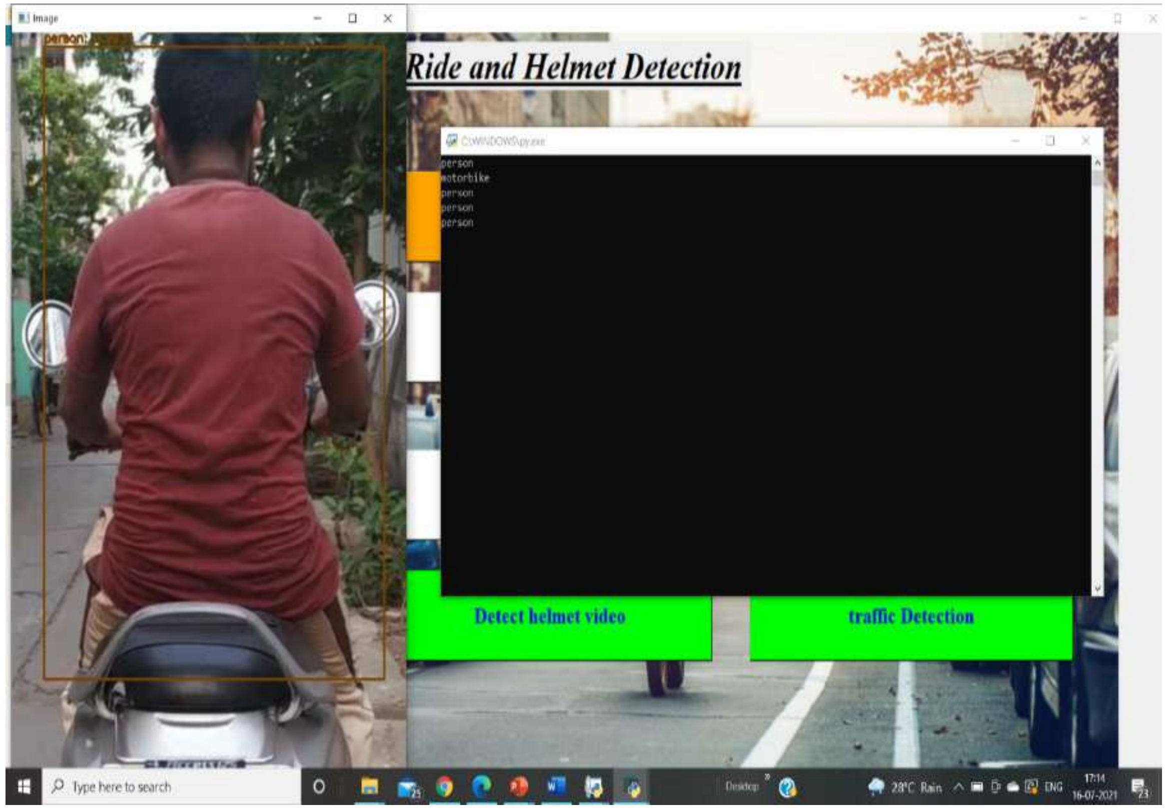
Result :

Helmet Input Video



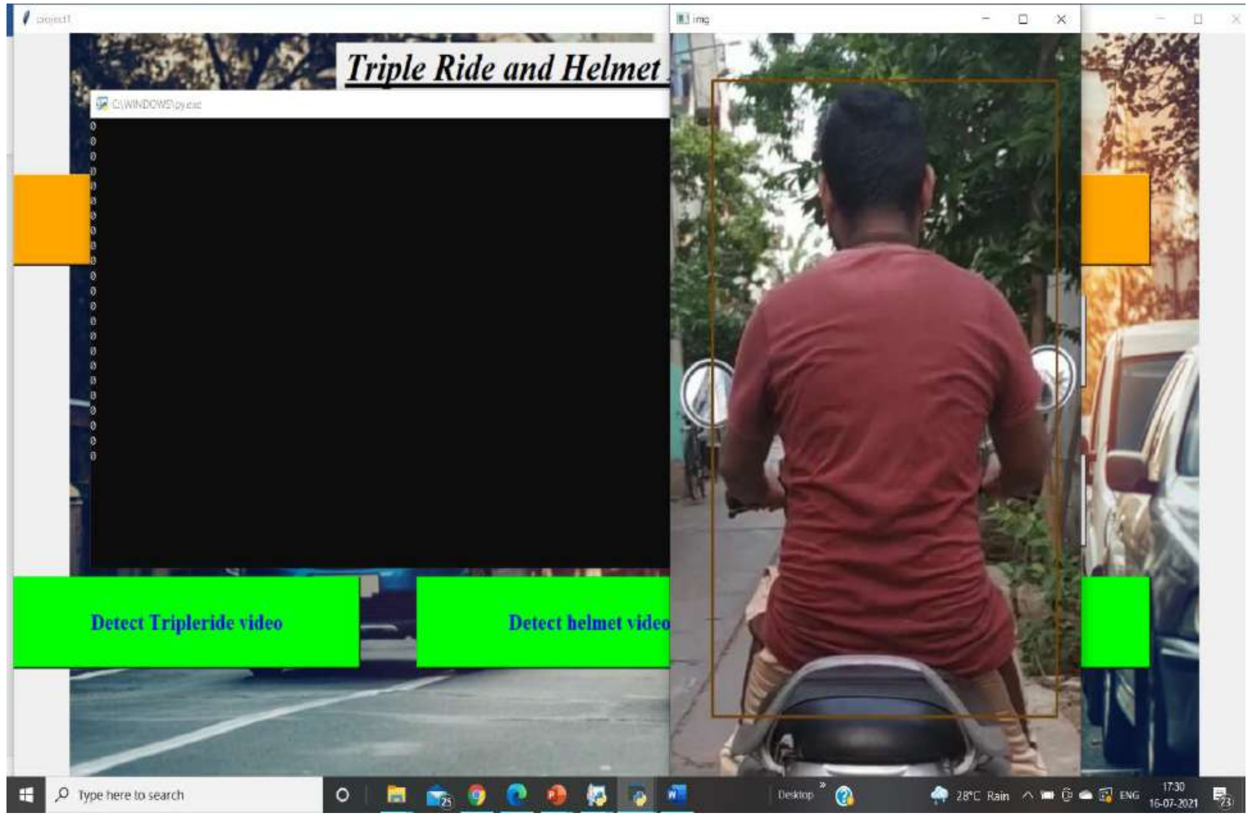
```
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
motorbike
person
motorbike
motorbike
person
motorbike
motorbike
person
```

Detect persons



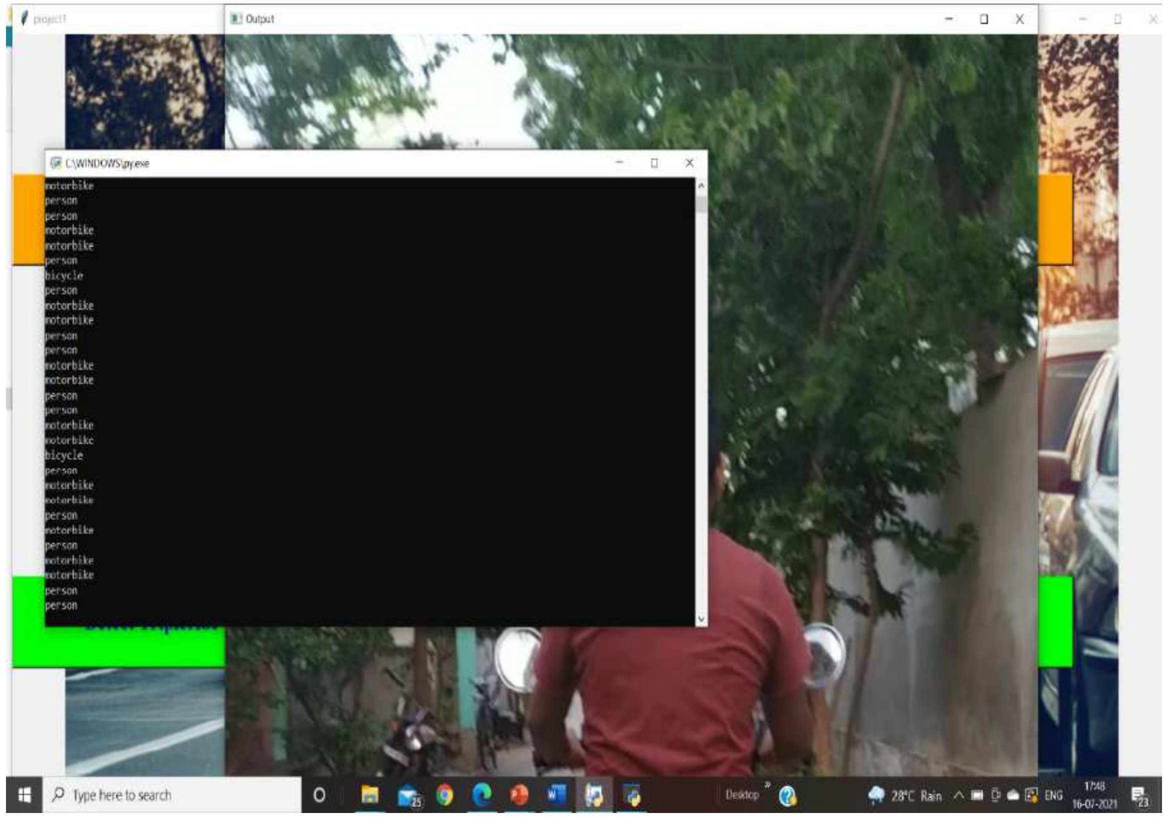
```
person
motorbike
person
person
person
```

Detect Helmet



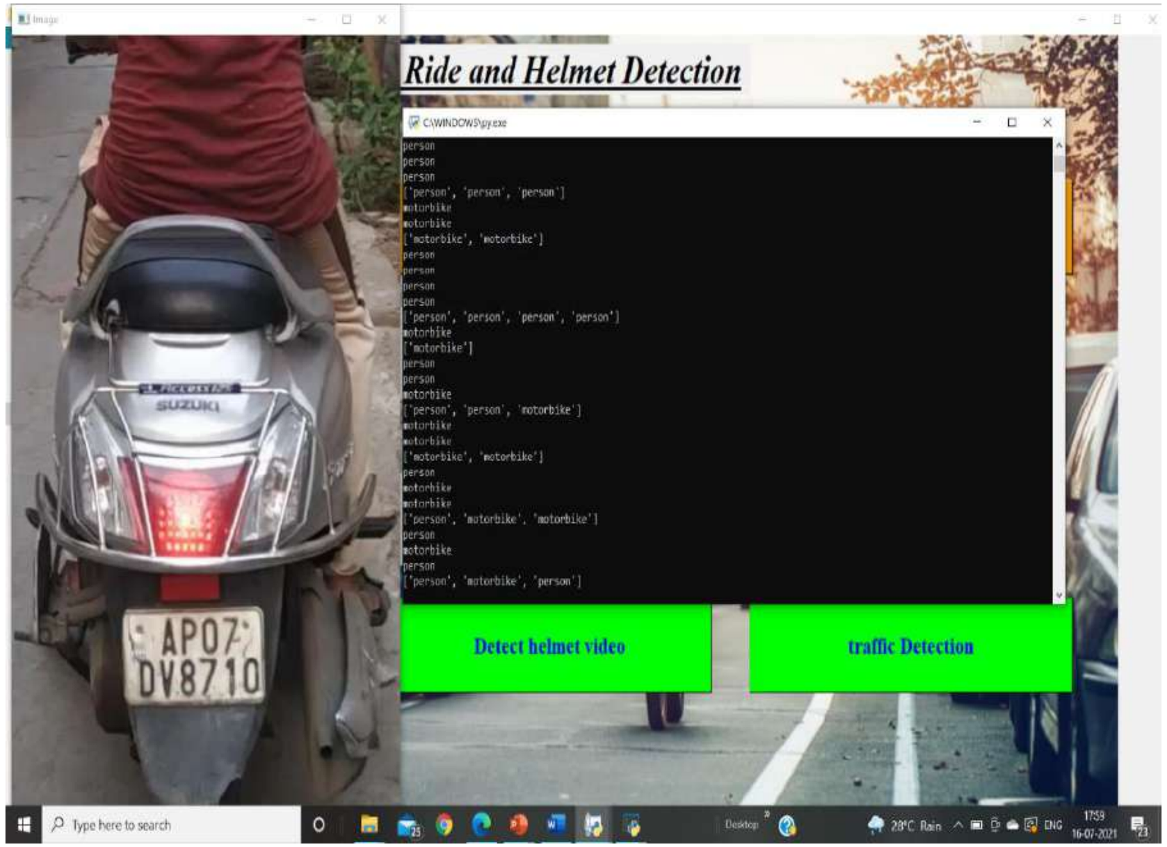
```
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

TripleRide input Video



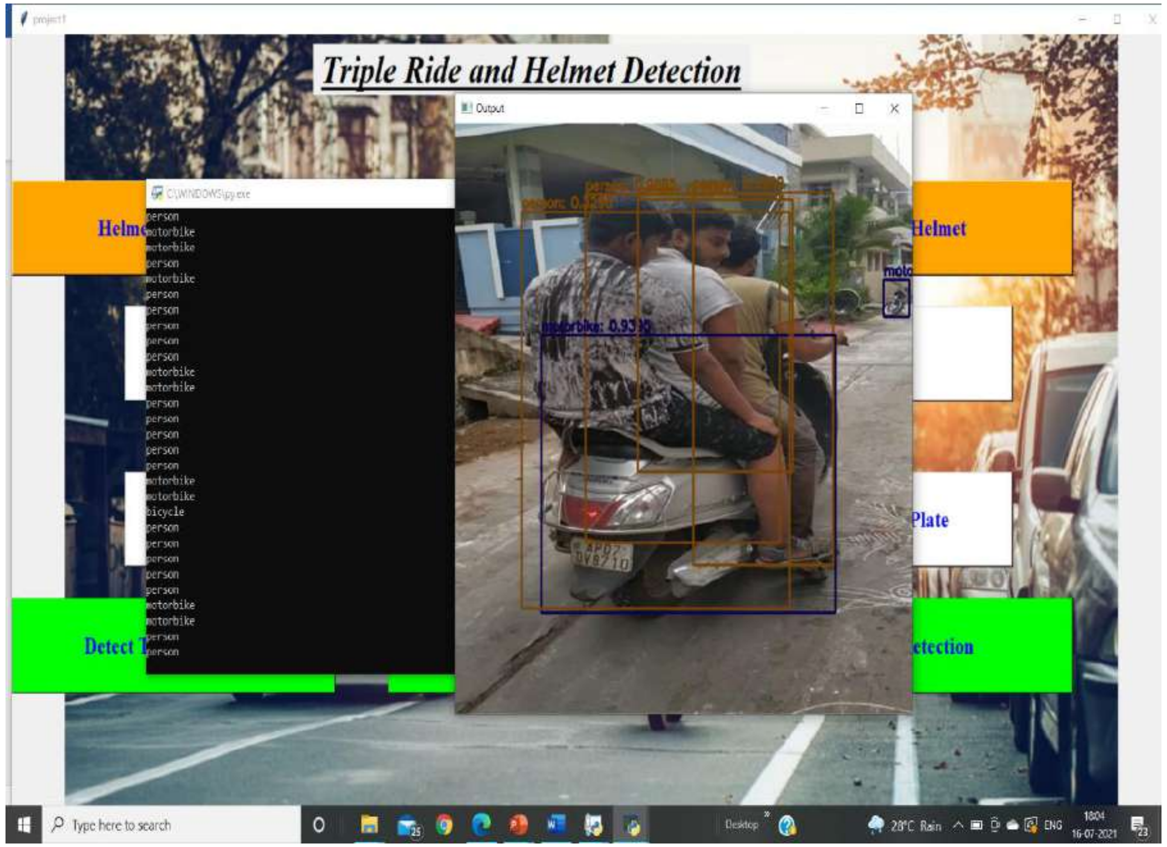
```
motorbike
person
person
motorbike
motorbike
person
bicycle
person
motorbike
motorbike
person
person
motorbike
motorbike
person
person
motorbike
motorbike
bicycle
person
motorbike
motorbike
person
motorbike
person
motorbike
motorbike
person
person
```

Detect Persons



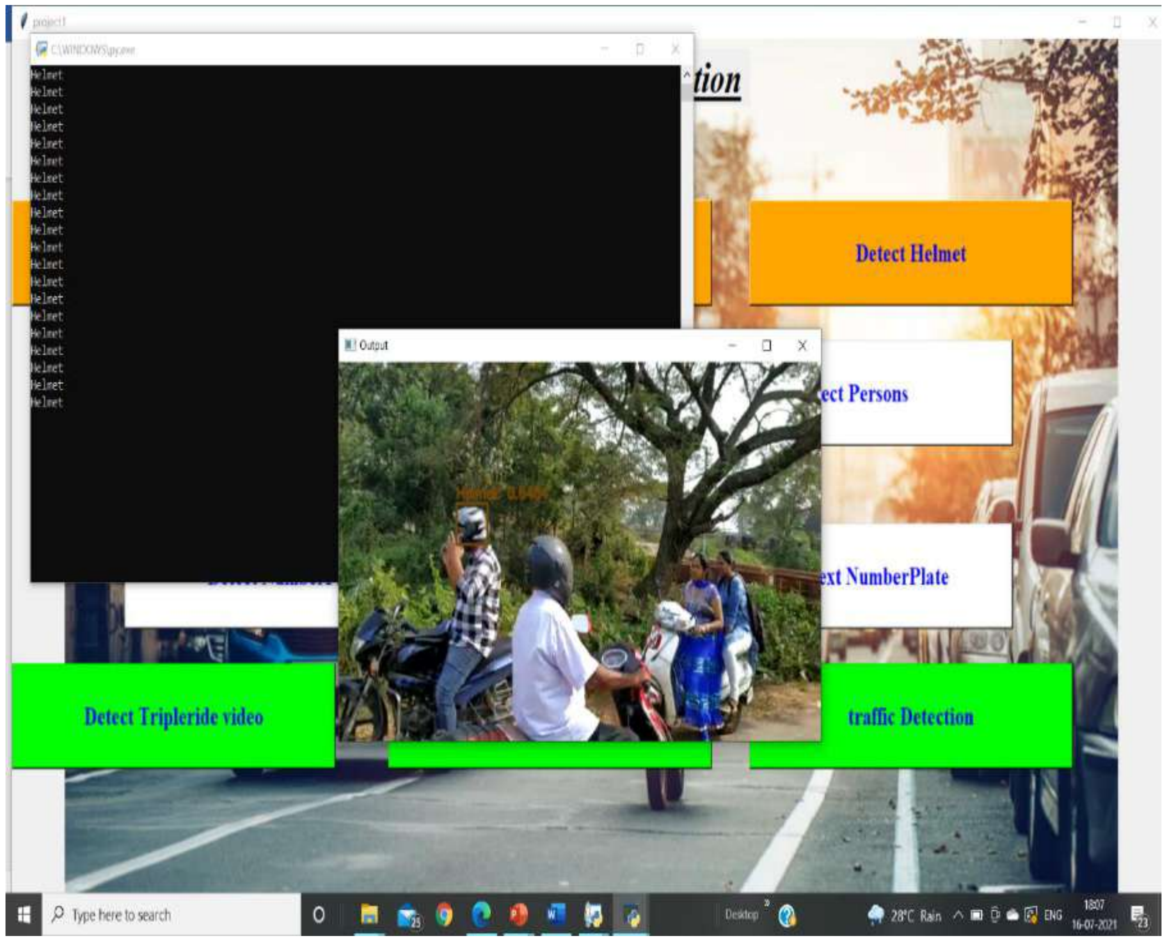
```
person
person
person
['person', 'person', 'person']
motorbike
motorbike
['motorbike', 'motorbike']
person
person
person
person
['person', 'person', 'person', 'person']
motorbike
['motorbike']
person
person
motorbike
['person', 'person', 'motorbike']
motorbike
motorbike
['motorbike', 'motorbike']
person
motorbike
motorbike
['person', 'motorbike', 'motorbike']
person
motorbike
person
['person', 'motorbike', 'person']
```

Detect Tripleride video



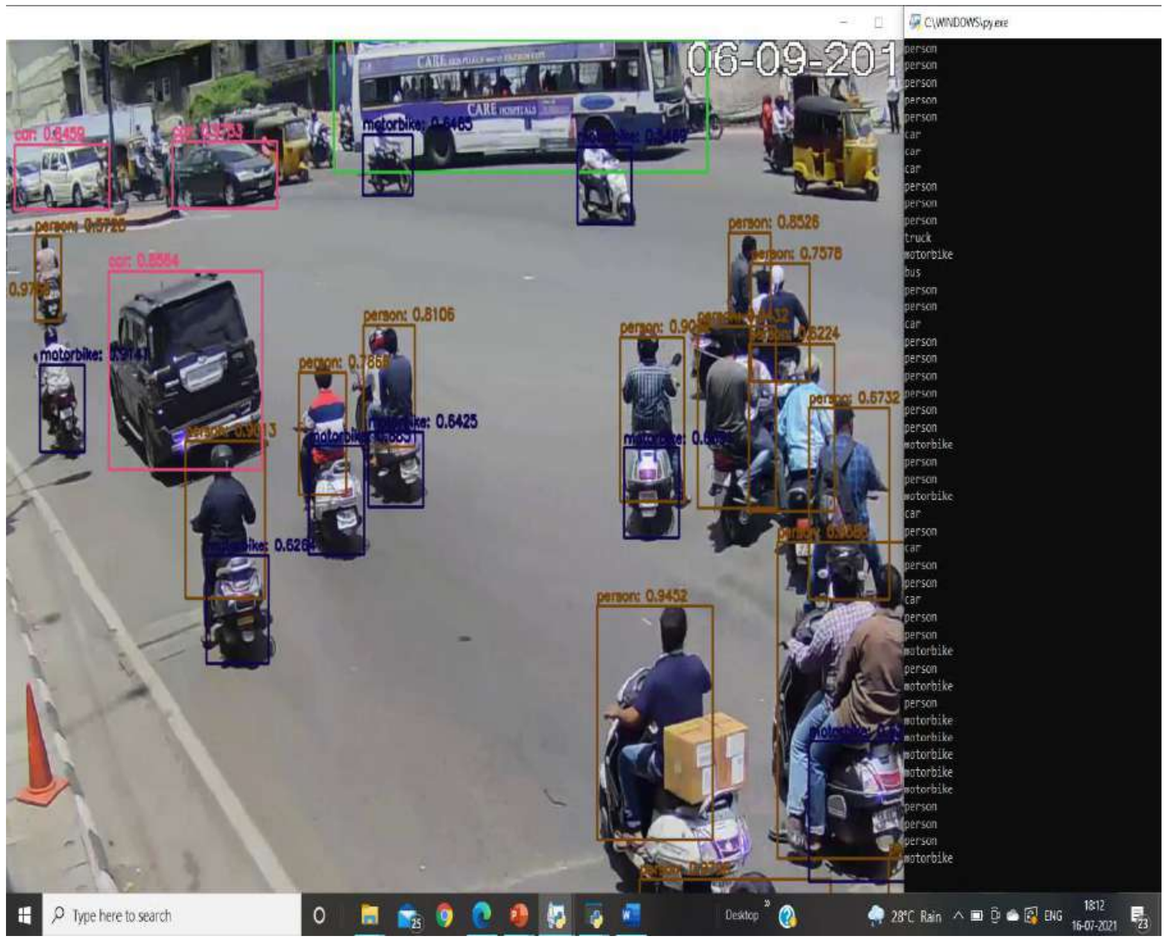
```
person
motorbike
motorbike
person
motorbike
person
person
person
person
person
motorbike
motorbike
person
person
person
person
person
motorbike
motorbike
bicycle
person
person
person
person
person
motorbike
motorbike
person
person
```

Detect Helmet video



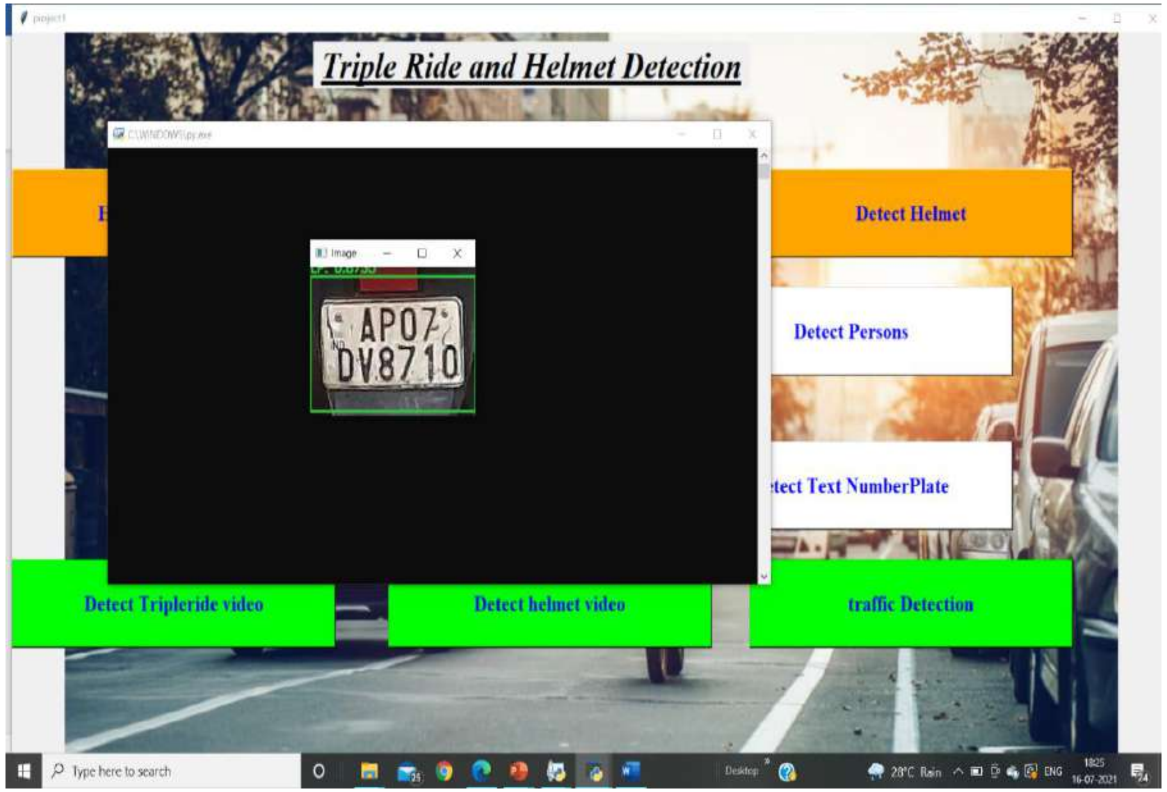
```
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
Helmet
```

Traffic Detection

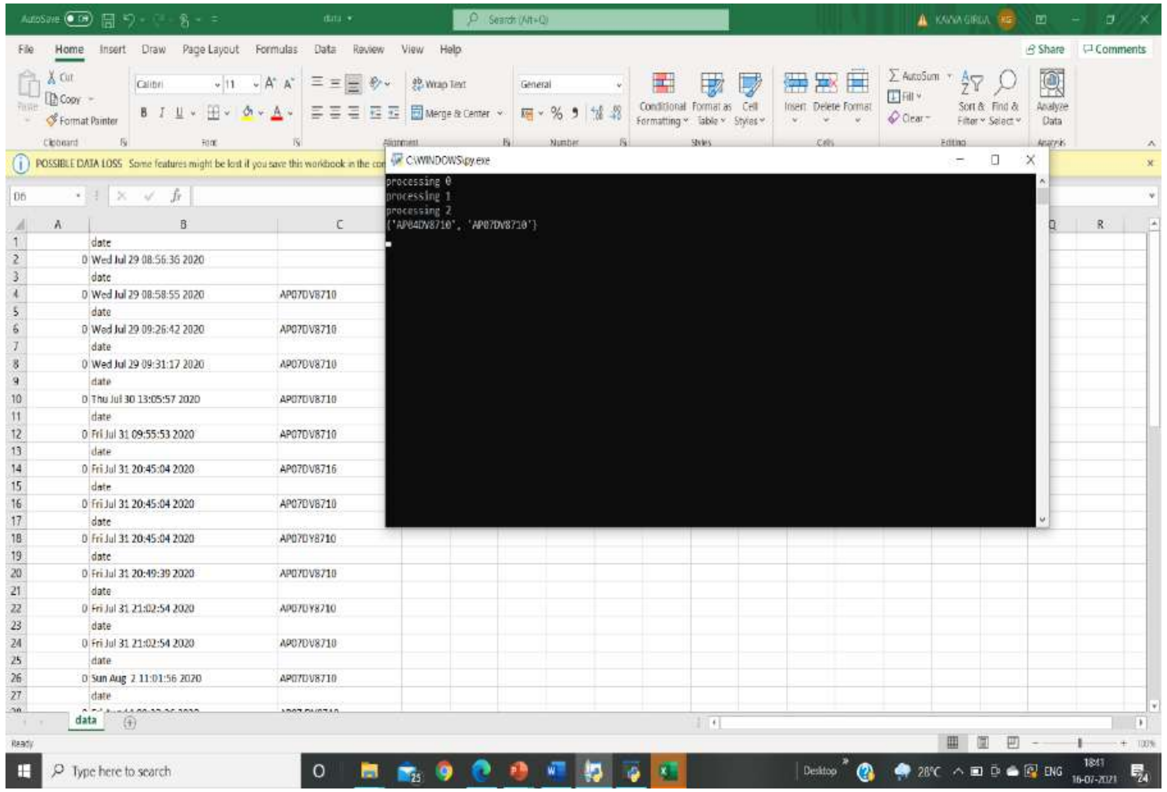


```
person
person
person
person
person
car
car
car
person
person
person
truck
motorbike
bus
person
person
car
person
person
person
person
person
person
motorbike
person
person
motorbike
car
person
car
person
person
car
person
person
motorbike
person
motorbike
person
motorbike
motorbike
motorbike
motorbike
motorbike
person
person
person
motorbike
```

Detect NumberPlate



Detect Text NumberPlate


POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the co

C:\WINDOWS\py.exe

```
processing 0
processing 1
processing 2
{'AP04DV8710', 'AP07DV8710'}
```

	A	B	C
1		date	
2	0	Wed Jul 29 08:56:36 2020	
3		date	
4	0	Wed Jul 29 08:58:55 2020	AP07DV8710
5		date	
6	0	Wed Jul 29 09:26:42 2020	AP07DV8710
7		date	
8	0	Wed Jul 29 09:31:17 2020	AP07DV8710
9		date	
10	0	Thu Jul 30 13:05:57 2020	AP07DV8710
11		date	
12	0	Fri Jul 31 09:55:53 2020	AP07DV8710
13		date	
14	0	Fri Jul 31 20:45:04 2020	AP07DV8716
15		date	
16	0	Fri Jul 31 20:45:04 2020	AP07DV8710
17		date	
18	0	Fri Jul 31 20:45:04 2020	AP07DY8710
19		date	
20	0	Fri Jul 31 20:49:39 2020	AP07DV8710
21		date	
22	0	Fri Jul 31 21:02:54 2020	AP07DY8710
23		date	
24	0	Fri Jul 31 21:02:54 2020	AP07DV8710
25		date	
26	0	Sun Aug 2 11:01:56 2020	AP07DV8710
27		date	

Conclusion

- As when compared to other algorithm YOLO is found be more advantageous and has higher efficiency and accuracy. Hence, we use the same approach to identify triple riding.

- Since we are implementing our project using YOLO algorithm which is a CNN based approach, results are obtained at fastest speed .

Future scope

- In future we can implement the traffic rules violation system in a high profile quality. So that we can improve the traffic rules violation system.

- In future we can use camera and other sensor identify the vehicle and identify the traffic rules violation and can automatically send the images to the administrator.

- When the fine is applied the people can easily understand and they will came to know the fine for their violation.

Advantages

- Automation of this procedure is exceptionally attractive for vigorous observing of these infringements and additionally it likewise altogether lessens the measure of human resource required.
- It does continuous monitoring of traffic rules 24x7.
- No need of human intervention.
- An automated system, if employed to detect and penalize motorcycle riders without a helmet, will naturally motivate the two-wheeler riders to wear helmets.
- Reduce road accidents.

Applications

- Can be used in all traffic signals.
- Can be used in any type of streets.

References :

1] Automatic Motorcyclist Helmet Rule Violation Detection using Tensorflow & Keras in OpenCV - IEEE Conference Publication : https://ieeexplore.ieee.org/document/9087168

[2] https://www.sciencedirect.com/science/article/pii/S026322411830900X ,2019

[3] Pattern Recognition and Machine learning by Christopher Bishop in 2006

[4] Digital image processing by William K Pratt

THANK YOU