Artificial Intelligence

Final Project Report

Sno.	Stud. ID	Names	Course Name/CID
01	9517	Javeria Hassan	Artificial Intelligence (103778)
02	9442	Jaweria Asif	
03	8904	Saba Hussain	
04	8718	Areeba Siddiqui	

Class ID: _103778_

Course Teacher: Dr. Umema Hani

Index

1. INTRODUCTION

We are going to show how to create an unbeatable AI agent that the classic **VEHICLE & PEDESTRIAN DETECTION**. You will learn the concept of that is widely and successfully used across the fields like **Artificial Intelligence**. We are proposing an artificial intelligence based system to detect motion of pedestrians on road, cars, bikes and buses. This project use machine learning technique, trained from XML files on how the vehicles look like and how to identify them.

2. PROJECT DETAILS

Our Project will detect and recognize vehicles and pedestrians motion in a video. Video should be in avi or mp4 format. We are also working on counting how many vehicles and pedestrians passed from the road.

- **Detect Vehicles**
  Using open-cv, the system detects vehicles bikes, cars and buses.
- **Detect People on pedestrian**
  Detect people while moving.
- **Counting the Number of Vehicle and Pedestrian (In Progress).**
  This feature is under progress. The system will detect how many vehicles and people passed during specified time.

2.1 PEAS

PEAS	Detail
P	Detect the vehicles and pedestrian movement
E	Car, Bus ,Bike, Pedestrian, Road
A	Screen/GUI
S	Algorithm sense with a object move

2.2 ENVIRONMENT TYPE

Deterministic / Stochastic	Fully / Partially Observable	Episodic / Sequential	Static / Dynamic /Semi Dynamic	Discrete / Continuous	Single / Multi agent	Known / Un Known
Deterministic	Fully observable	Sequential	Static Dynamic	Continuous	Multi Agent	Unknown

2.3 AGENT TYPE

Applicable Type	Reason
Simple	Knowledge of recognize vehicles and pedestrians detection.
Reflex	Detect people while moving.
Goal	The goal is to detect the moving vehicles and humans in a video.
Utility	Not a utility based.
Learning	Trained from data file (XML) on how the vehicles look like? Performance depends on data.

2.4 AGENT ORGANIZATION

Applicable Organization Type	Reason
Atomic	**Not Atomic.**
Structures	**Not Structured.**
Factored	Lie in factored type because it is basically a *VEHICLE & PEDESTRIAN_MOTION DETECTION*

3. CODE DETAILS

3.1. Demo on YouTube URL

https://www.youtube.com/watch?v=Psl5oLzuAv8

3.2. Major Steps in your Code and Libraries used

Open CV code has been used to detect motion of objects like Humans, cars, bike and buses. Tkinter used to create GUI of project.

Its major steps include following along with the information of libraries used:

Step 1: Reading, Writing and Displaying Images. ... OpenCV

- OpenCV reads a given image in the BGR format by default. So, you'll need to change the color space of your image from BGR to RGB when reading images using OpenCV
- By default, the *imread* function reads images in the BGR (Blue-Green-Red) format. We can read images in different formats using extra flags in the *imread* function:

- **cv2.IMREAD_COLOR:** Default flag for loading a color image
- **cv2.IMREAD_GRAYSCALE:** Loads images in grayscale format

Step 2: Resizing Images…OpenCV

- Machine learning models work with a fixed sized input. The same idea applies to computer vision models as well. The images we use for training our model must be of the same size.

Step 3: Basic segmentation. ... OpenCV

- Thresholding is an image **segmentation** method. It compares pixel values with a threshold value and updates it accordingly. OpenCV supports multiple variations of thresholding. A simple thresholding function can be defined like this: if Image(x, y) > threshold, Image(x, y) = 1
- Otherwise, Image(x, y) = 0
- A simple application of image thresholding could be dividing the image into its foreground and background.

Step 4: Image Filtering. ... OpenCV

- Gaussian filtering is also used for image blurring that gives different weights to the neighboring pixels based on their distance from the pixel under consideration.

Step 5: Image Contours. ... OpenCV

- A contour is a closed curve of points or line segments that represents the boundaries of an object in the image. Contours are essentially the shapes of objects in an image.
- Unlike edges, contours are not part of an image. Instead, they are an abstract collection of points and line segments corresponding to the shapes of the object(s) in the image.
- We can use contours to count the number of objects in an image, categorize objects on the basis of their shapes, or select objects of particular shapes from the image.

Step 6: Image Dilation. ... OpenCV

- It is useful in joining broken parts of an object.

Step 7: Drawing Rectangles Around moving Objects…..OpenCV

- **OpenCV-Python** is a library of Python bindings designed to solve computer vision problems. cv2.rectangle () method is used to draw a rectangle on any image.

3.3. Code

```
from tkinter import *
root=Tk()
root.configure(background="#87CEEB")
```

CODE FOR PEDESTRIAN DETECTION

```
def people(): # method for pedestrian detection
    import cv2
    import numpy as np
    cap = cv2.VideoCapture('pedestrians.avi')
    frame_width = int( cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height =int( cap.get( cv2.CAP_PROP_FRAME_HEIGHT))

    #--------for avi file
    fourcc = cv2.VideoWriter_fourcc('X','V','I','D')
    out = cv2.VideoWriter("output.avi", fourcc, 5.0, (1280,720))

    #--------- for mp4 files
    #fourcc = cv2.VideoWriter_fourcc(*"X264")
    #out = cv2.VideoWriter("output.mp4", fourcc, 15.0, (1280, 360))
    ret, frame1 = cap.read()
    ret, frame2 = cap.read()
    print(frame1.shape)
    while cap.isOpened():
        diff = cv2.absdiff(frame1, frame2)
        gray = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
        blur = cv2.GaussianBlur(gray, (5,5), 0)
        _, thresh = cv2.threshold(blur, 20, 255, cv2.THRESH_BINARY)
        dilated = cv2.dilate(thresh, None, iterations=3)
        contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
        for contour in contours:
            (x, y, w, h) = cv2.boundingRect(contour)
            if cv2.contourArea(contour) < 900:
                continue
            cv2.rectangle(frame1, (x, y), (x +w, y +h), (0, 255, 0), 2)
            cv2.putText(frame1, "Status: {}".format('Movement'), (10, 20),
cv2.FONT_HERSHEY_SIMPLEX,1, (0, 0, 255), 3)
        #cv2.drawContours(frame1, contours, -1, (0, 255, 0), 2)
        image = cv2.resize(frame1, (1280,720))
        out.write(image)
        cv2.imshow("People", frame1)
        frame1 = frame2
        ret, frame2 = cap.read()
        if cv2.waitKey(44) == 27:
            break
    cv2.destroyAllWindows()
    cap.release()
    out.release()
```

CODE FOR BUS DETECTION

```
def bus():
    import cv2
    cascade_src = 'Bus_front.xml'
```

```
    video_src = 'bus1.mp4'
    cap = cv2.VideoCapture(video_src)
    fgbg=cv2.createBackgroundSubtractorMOG2()
    car_cascade = cv2.CascadeClassifier(cascade_src)
    while True:
        ret, img = cap.read()
        fgmask=fgbg.apply(img)
        if (type(img) == type(None)):
            break

        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        cars = car_cascade.detectMultiScale(gray, 1.16, 1)
        for (x ,y ,w ,h) in cars:
            cv2.rectangle(img,(x, y),(x+ w, y +h),(0,0,255),3)
        cv2.imshow('video', img)
        if cv2.waitKey(33) == 27:
            break

    cv2.destroyAllWindows()
```

CODE FOR BIKE DETECTION

```
def bike():

    import cv2
    cascade_src = 'two_wheeler.xml'
    video_src = 'two_wheeler2.mp4'
    cap = cv2.VideoCapture(video_src)
    fgbg = cv2.createBackgroundSubtractorMOG2()
    car_cascade = cv2.CascadeClassifier(cascade_src)

    while True:
        ret, img = cap.read()
        fgbg.apply(img)
        if (type(img) == type(None)):
            break
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        cars = car_cascade.detectMultiScale(gray,1.1, 2)
        for (x ,y ,w ,h) in cars:
            cv2.rectangle(img,(x, y),(x +w ,y +h),(0,255,255),2)
        cv2.imshow('video', img)
        if cv2.waitKey(33) == 27:
            break

    cv2.destroyAllWindows()
```

CODE FOR CAR DETECTION

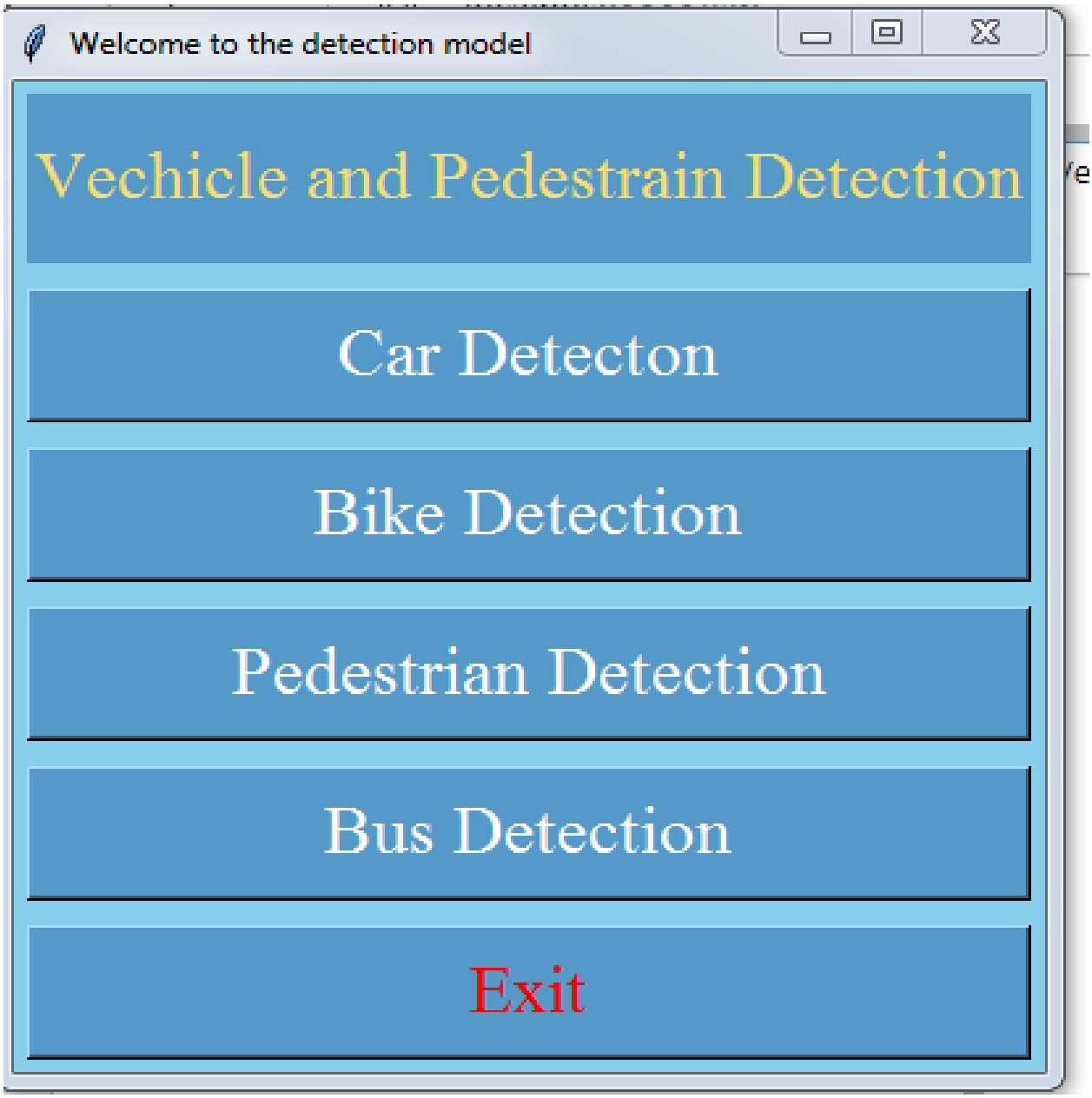
```
def cars():
    import cv2
    cascade_src = 'cars.xml'
    video_src = 'car1.avi'
    cap = cv2.VideoCapture(video_src)
    fgbg = cv2.createBackgroundSubtractorMOG2()
    car_cascade = cv2.CascadeClassifier(cascade_src)
    while True:
```

```
        ret, img = cap.read()
        fgbg.apply(img)
        if (type(img) == type(None)):
            break
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        cars = car_cascade.detectMultiScale(gray, 1.1, 2)
        for (x ,y ,w ,h) in cars:
            cv2.rectangle(img,(x ,y),(x+ w, y+ h),(0,255,255),2)
        cv2.imshow('video', img)
        if cv2.waitKey(33) == 27:
            break
    cv2.destroyAllWindows()

def exit():

    root.destroy()
root.title("Welcome to the detection model")
```

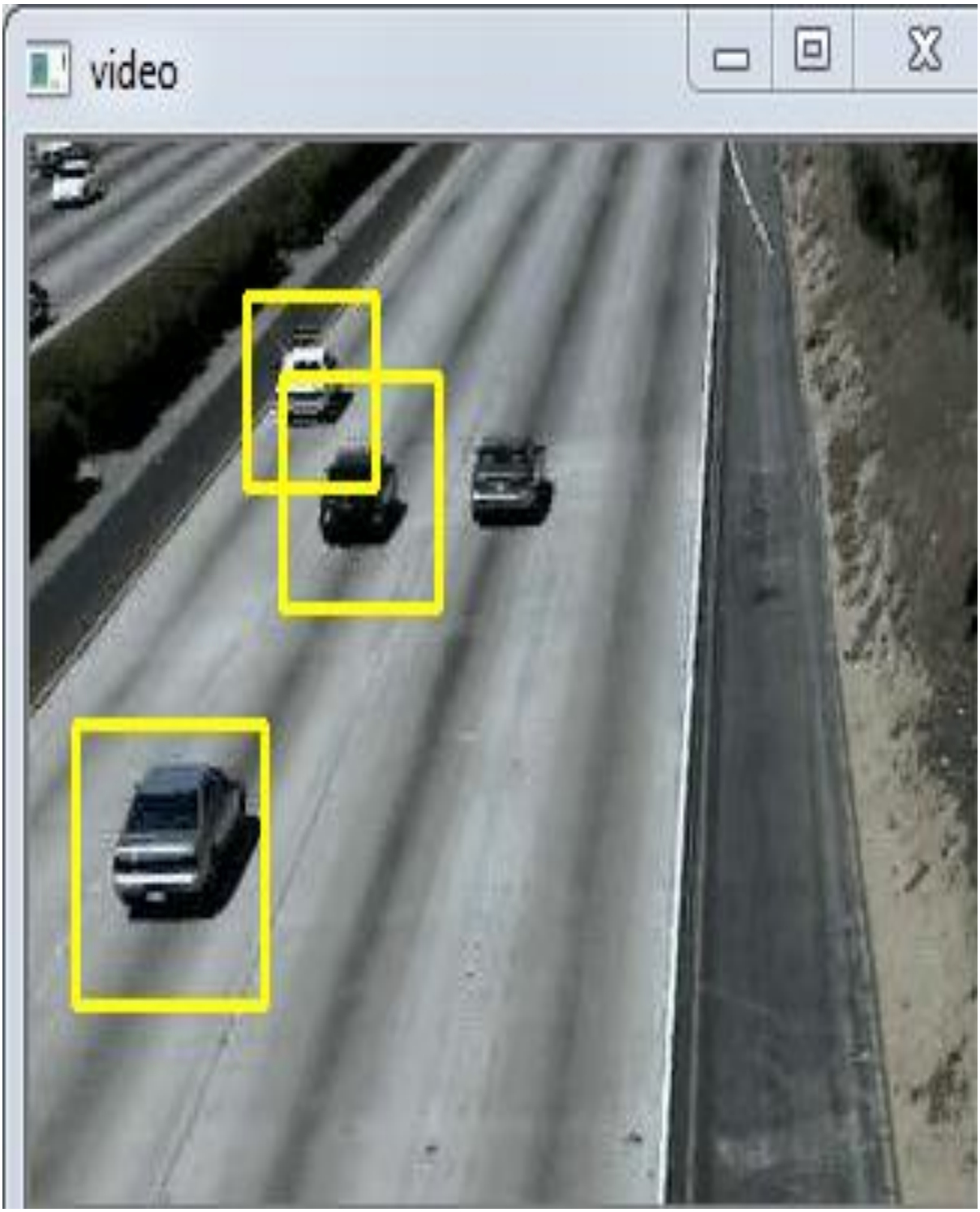
CODE FOR GUI

```
Label(root, text="Vehicle and Pedestrain Detection", font=("times new
roman",20),fg="#F7DC6F",bg="#5499C7",height=2).grid(row=0,rowspan=2,columnspan=2,sticky
=N+E+W+S,padx=5,pady=5)
Button(root, text="Car Detection",font=("times new
roman",20),bg="#5499C7",fg='white',command=cars).grid(row=4,columnspan=2,sticky=W+E+N+
S,padx=5,pady=5)
Button(root, text="Bike Detection", font=("times new
roman",20),bg="#5499C7",fg='white',command=bike).grid(row=5,columnspan=2,sticky=N+E+W+
S,padx=5,pady=5)
Button(root,text="Pedestrian Detection",font=('times new
roman',20),bg="#5499C7",fg="white",command=people).grid(row=6,columnspan=2,sticky=N+E+
W+S,padx=5,pady=5)

Button(root,text="Bus Detection",font=('times new
roman',20),bg="#5499C7",fg="white",command=bus).grid(row=7,columnspan=2,sticky=N+E+W+
S,padx=5,pady=5)

Button(root,text="Exit", font=('times new
roman',20),bg="#5499C7",fg="red",command=exit).grid(row=9,columnspan=2,sticky=N+E+W+S,
padx=5,pady=5)


root.mainloop()
```

3.4. Output images

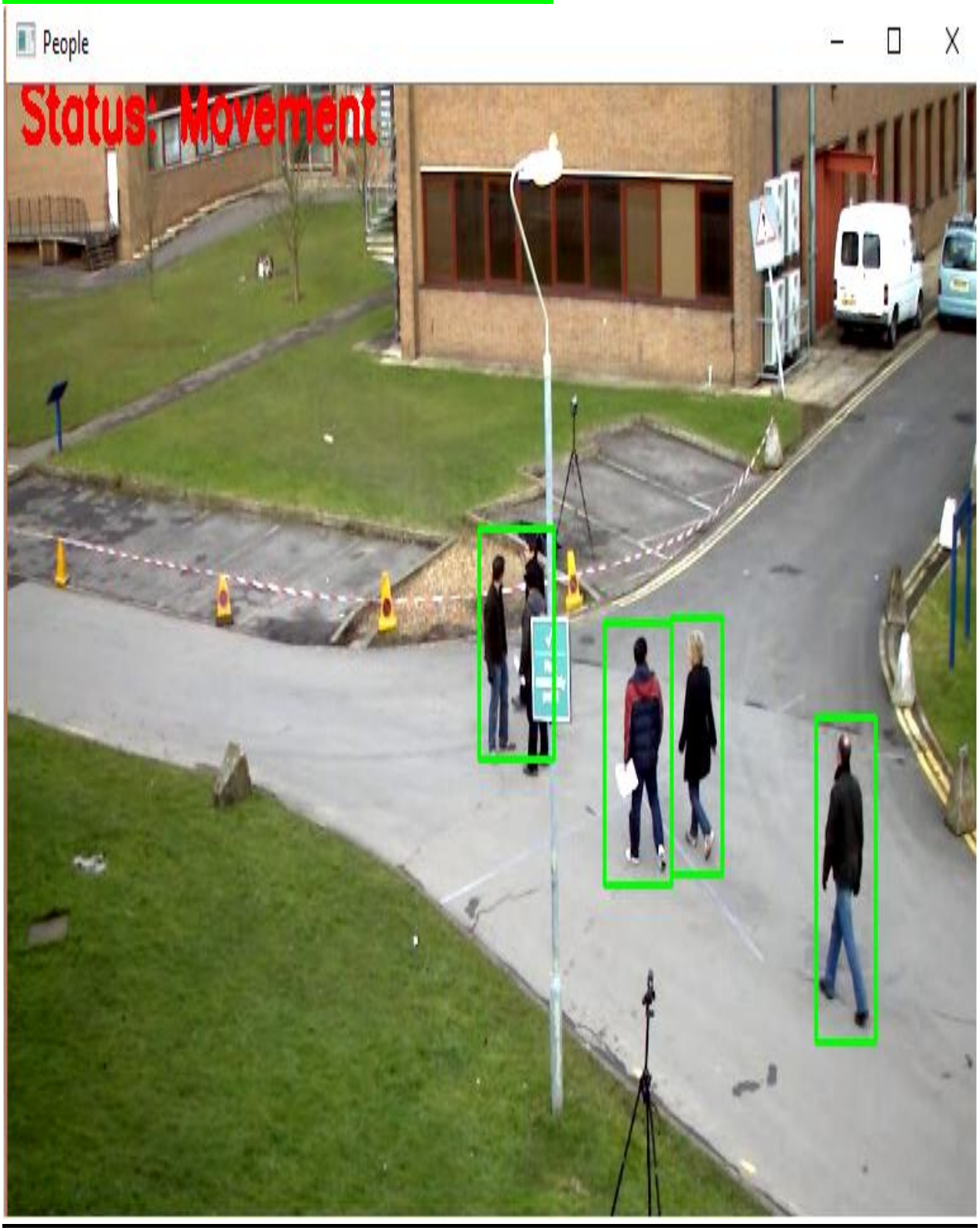


CAR DETECTION:



BIKE DETECTION:

PREDESTRIAN DETECTION:



BUS DETECTION:

4. PROJECT CONCLUSION

Application of AI concepts studied during course work, such that to attain high competency in market.

The area implemented is Computer Vision.

The Vehicle detection is a very important component in traffic surveillance and automatic driving. Vehicle detection is still an important challenge in computer vision. We are proposing an artificial intelligence based system to detect motion of pedestrians on road, cars, bikes and buses. This project use machine learning technique, trained from XML files on how the vehicles look like and how to identify them.