Recognizing Driver in Cars

Contents

Introduction

I love the idea of smart cities. The thought of automated smart energy systems, electrical grids, one-touch access ports – it's an enthralling concept! Honestly, it's a dream for a data scientist and I'm delighted that a lot of cities around the world are moving towards becoming smarter.

One of the core components of a smart city is automated recognizing drivers in cars. And that got me thinking about it.

- Think about it - if you can integrate a driver recognition system into a car, you can easily perform many operations and protect the car from being stolen.

Face recognition is finding its way into the new generations of cars in an attempt to increase safety and convenience. From car ignition to theft prevention – there are countless possibilities of using facial recognition in cars.

Methodology

Introduction

This project was done with "Open Source Computer Vision Library", the Open CV. Open CV hasplenty of power to begin doing computer vision with Python. Open CV is written in C ++ and hasPython binding. Using Python is the most popular and perhaps simple way to detect faces using theOpen CV package. This project is being done on just detecting faces, face recognition which is actually assigning a name to a face that means recognizing a person with their name. Here, we needto do training our data and refining our dataset to make this kind of code work properly. The trainingportion is the really time consuming part. There is still a growing number of applications to recognize the face in real time; recent examples are security and smart room deployment. If the level of performance performed by other biometric features can be increased, facial recognition will be available in the most preferred methods due to the lack of necessary infrastructure and simplicity.

Driver error is one of the most common cause of traffic accidents. The number of accidents increasing from day to day, it has become important to take care of human errors and help the humanity. All of this could end with self-driving cars that just need to know the destination and then leave the passengers continue their work. This will not only prevent accidents, but also bringself-relief for day-to-day minors driving activities for small objects. This project's aim is to build an autonomous car with face recognition by using AI. An HD camera with an ultrasonic sensor is used to provide needed data from the real world to the car. The car is able to reach the given destination safely and intelligently, thus avoiding the risk of human error. A lot of existing algorithms such as lane detection, obstacle detection are combined to provide the necessary controlto the car.

Data/Image acquisition

To complete this project on Face Recognition, we work on two phases:

- Data Gathering and Face detection
- Face Identification/Recognition

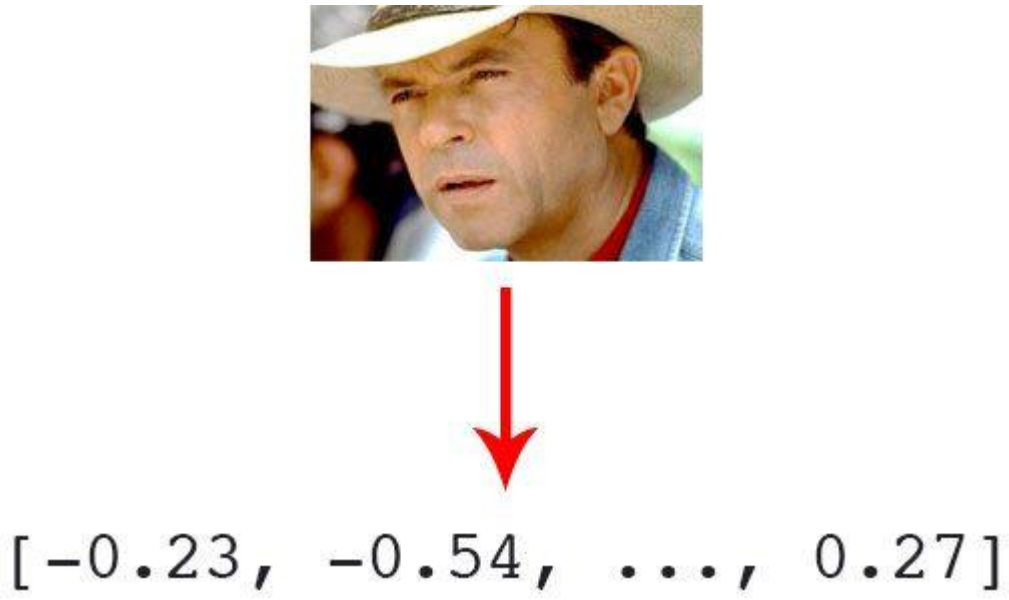

Figure 2.1: Facial recognition via deep learning and Python using the `face_recognition` module method generates a 128-d real-valued number feature vector per face.

Before we can recognize faces in images and videos, we first need to quantify the faces in our training set. Keep in mind that we are not actually training a network here — **the network has *already been trained* to create 128-d embeddings** on a dataset of ~3 million images.

We certainly *could* train a network from scratch or even fine-tune the weights of an existing model but that is more than likely overkill for many projects. Furthermore, you would need a *lot* of images to train the network from scratch.

Instead, it's easier to use the pre-trained network and then use it to construct 128-d embeddings for each of the 218 faces in our dataset.

Then, during classification, we can use a simple k-NN model + votes to make the final face classification. Other traditional machine learning models can be used here as well.

Recognizing faces in images

Figure 2.2: John Hammond's face is recognized using Adam Geitgey's deep learning `face_recognition` Python module.

Now that we have created our 128-d face embeddings for each image in our dataset, we are now ready to recognize faces in image using OpenCV, Python, and deep learning.

Preprocessed

Opening webcam

We can easily open the webcam using python language. Often, we have to capture live streams with cameras. Open CV provides a very easy interface to this. To capture a video from the camera(my laptop's webcam), convert it to video while

displaying it and display it. Just a simple way to get started the project. To capture a video need to create a video capture object. This logic could bethe device index or the name of a video file. The device index is the number to specify the cameraonly. Usually a camera will be connected. So simply pass 0. After that, we can capture frame-by- frame. But at the end, release the capture. We use the waitKey() after imshow() function to pause each frame in the video.

Face Detection

There are basically two primary ways to find faces using OpenCV:

- Haarcascade Classifier
- LBP Cascade Classifier

For face detection important module importing and the module work as:

- cv2: This is the OpenCV module and contains the functions for face detection and identification/recognition.

Using Haarcascade classifier because it is more accurate, but it is slower than LBP. OpenCV package has all the data that need to use Harrcascade classifier. OpenCV contains many pre-trainedclassifiers for face, eyes, smile etc. Basically, we need an XML file with correct face information.Here will use built-in classifier. Can find it in OpenCV library installed on it. Than we create a function that takes the path of an image file. Next we convert the images to grayscale. The computer's vision often works better than the gray color, or at least in the case of OpenCV we cansee it. There are some function that detects the actual face and is the key part of detecting face:

- The detectMultiScale function is a general function that detects objects. Since we are calling
  it on the face cascade, that’s what it detects.
- Gray is the input grayscale image.
- The second is the scaleFactor. Since some faces may be closer to the camera, they would appear bigger than the faces in the back. The scale factor

compensates for this.

- The detection algorithm uses a moving window to detect objects. minNeighbors defines how many objects are detected near the current one before it declares the face found. minSize, meanwhile, gives the size of each window. The function returns a list of rectangles in which it believes it found a face. Next, we will loop over where it thinks it found something. This function returns 4 values: the x and y location of the rectangle, and the rectangle's width and height (w , h). We use these values to draw a rectangle using the built-in rectangle() function. In the end, we display the image and wait for the user to press a key.

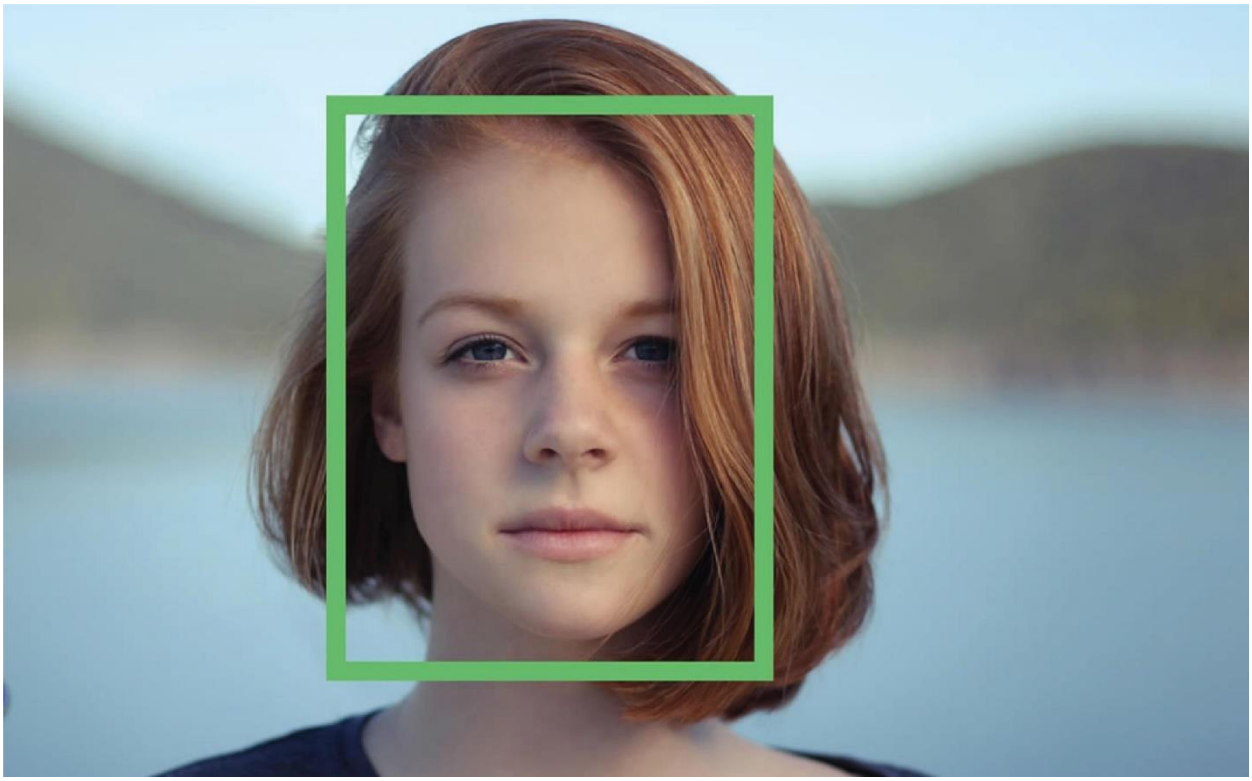

Figure 2.3: Example of face detecting

Face Identification/Recognition

After completing training, it can be applied to a region of interest (of the same size

as used duringthe training) in an input image. Here, we capture a face on our camera and if this person had his/herface captured before and trained before, our recognizer will make a prediction returning its id andname.

Figure 2.4: Example of face Identification

Result & Discussion

Face Detection, Identification/Recognition and Automation car

Introduction

Machine learning, artificial intelligence, face identification are the main topics of the day. Face recognition is an exciting and interesting field in real-time applications. In recent years, many facerecognition algorithms have been developed. The most common and easiest way to detect and recognize faces using Python is to use the OpenCV package. Face detection and identification aretwo different things. A real-time face recognition system can identify or confirm a person from a video image. In order to recognize a face in a frame, it is first necessary to judge whether or not a face exists in a frame. If it exists, mark it as a region of interest (ROI), extract it, and process it forface recognition. Face recognition is an important part of the human perception system's ability andhuman daily work while building a similar computer face recognition model. The computer modelcontributes not only to theoretical knowledge but also to many practical applications such as automatic crowd surveillance, access control, computer interface design, image database management by contents, criminal identification etc

The automotive automation industry is growing. The valuations of automakers and private investments in these companies are exploding. Technologies such as Lexus, BMW and Mercedes have already developed autonomous automotive technology. Automation cars are an unusual product from the perspective of safety-related outcomes. Driverless technology is still in advanced testing, but a partially automated technology has been in recent years. Renault / Nissan is hoping for a new partnership with Microsoft to advance the efforts of the autonomous car. Renault / Nissanplans to commercialize 10 types of passenger cars by 2020. Carlos Ghosn, Chief Executive Officer, told TechCrunch: "We know that autonomy is a major concern for consumers - it's about first highway in brick, then you will have several lanes I will have a highway and you will drive a city.All these steps must be

completed by 2020. 2020 is an autonomous urban car, maybe 2025 withoutdriver a car. In this project, we make a device that calculates the distance and triggers an alarm when objects are too close to the sensor. The buzzer will sound and an LED light will be triggered.

Face Registration

When user run the code **main.py** first time then it will start the registration process, it will ask to register the driver name.

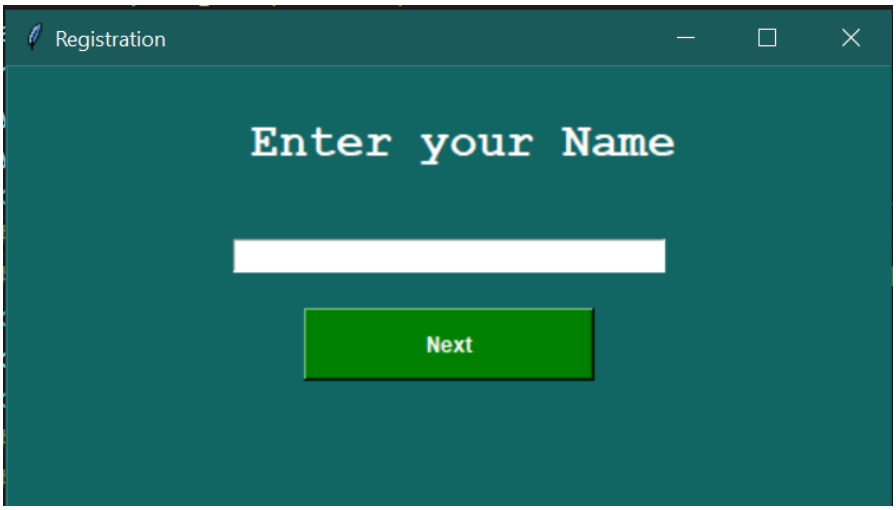

Figure3.1: For Register Name Of User

If the username field is left blank it will show a warning message with the name **'Invalid Name'** asking the user to enter a correct name.

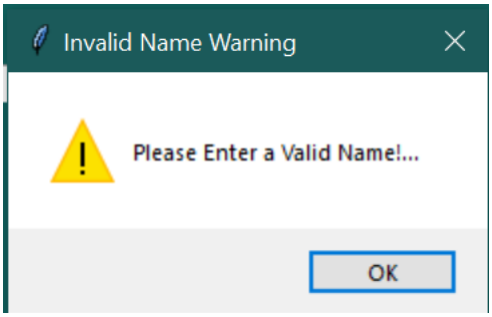

Figure 3.2:Warning when name field is empty.

After entering the correct name it will open the camera which detects the face. Here the user has two buttons, one to register the face and the other to skip the registration process.

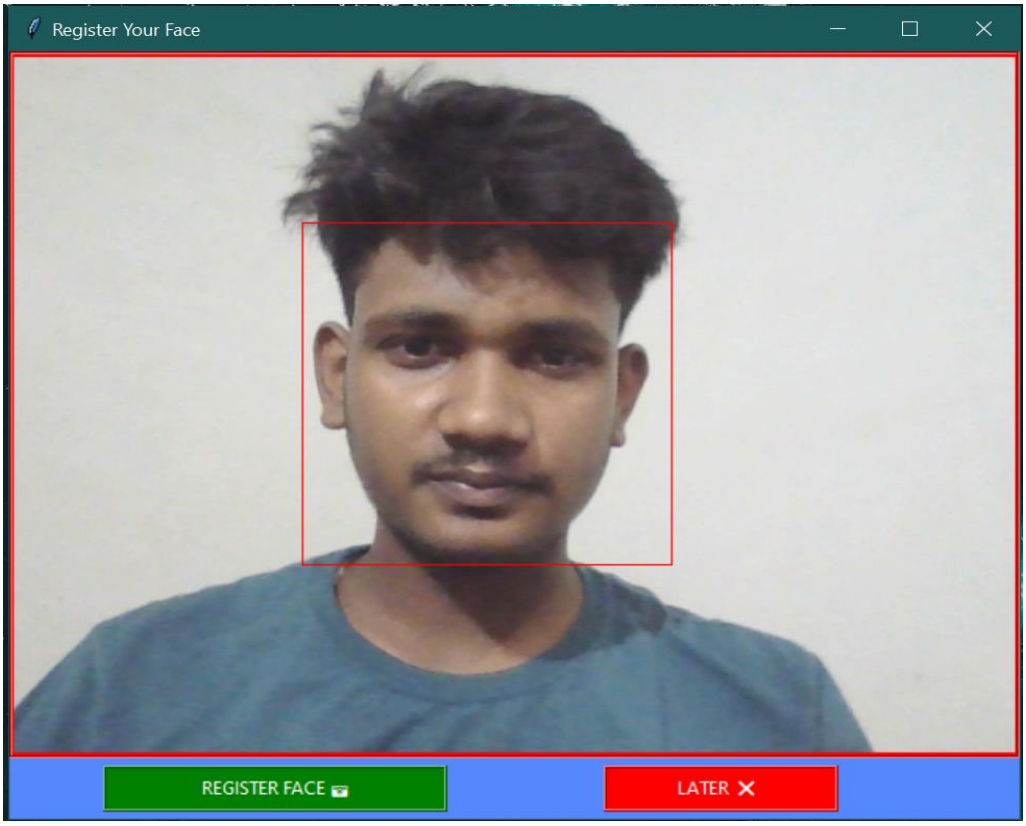

Figure3.3: Detecting the face in frame.

After clicking on the **'Register Face'** button, the face data will be saved in the locale system. And after all the process the user gets a confirmation message.

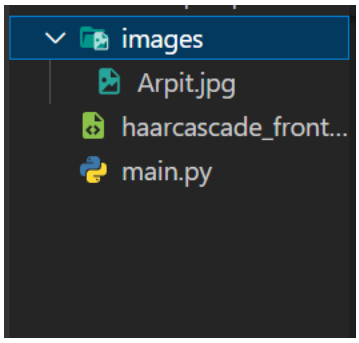

Figure 3.4:Save face data with given name in system.

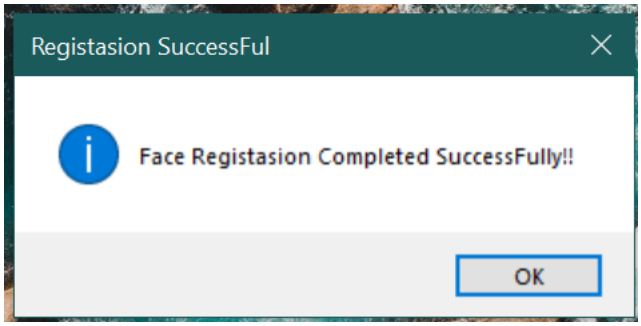

Figure 3.5:After successfully completing the local registration process.

Face Recognition And Identification

When the user comes after registration and re-runs the code, again the camera will open and try to detect the face.

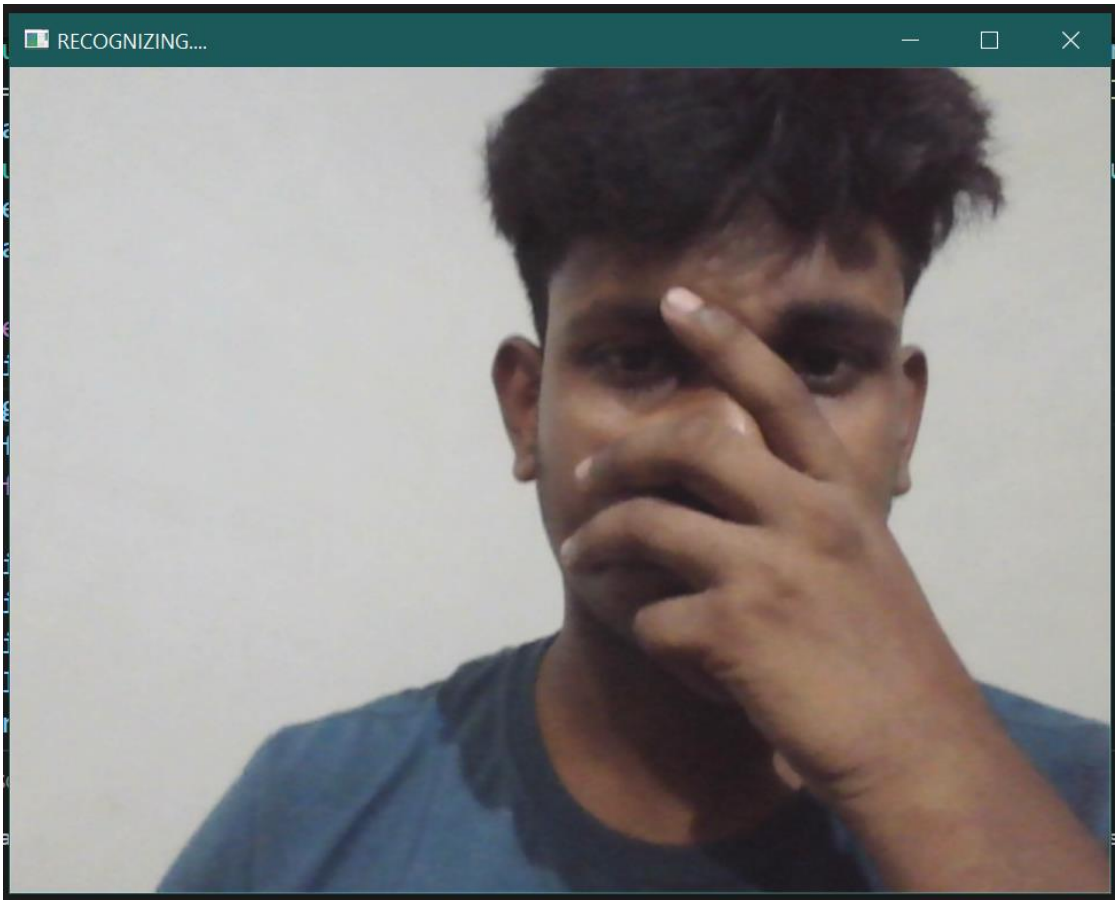

Figure3.6: Tring to Recognizing the face.

After identification of the user it shows the username and '**Access Granted'** message along with the welcome message.

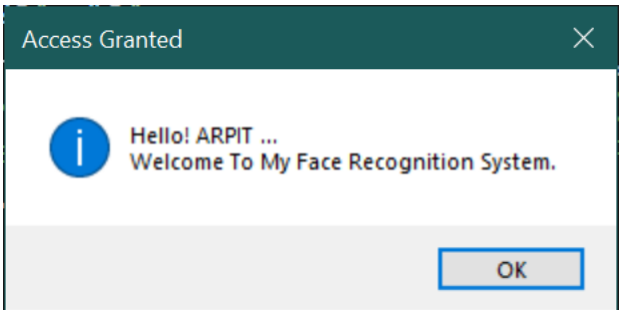

Figure3.7: After successful Identification of user.

Conclusion

Outcomes

In this project, there are face detection and identification of human faces and automation car are proposed and their performances are examined in the previous chapters. There are some unique outcomes of these results through this project. In this section, these aftermaths are mentioned gradually which will show the total project work at a glance. The outcomes are following below:

- The webcam can be opened using python and opencv.
- There can be opened multiple camera at a time.
- Face detection can now done in real time.
- Opencv uses machine learning algorithms to search for faces within a picture.
- Opencv comes with a number of built-in cascades for detecting everything fromfaces to eyes to hands to legs.
- The detectMultiScale function is a general function that detects objects. Here, weare calling it on the face cascade, so that's what it detects.
- The frame is in grayscale because cascade works in this way.
- Some faces may be closer to the camera, they would appear bigger

than the facesin the back. The scale factor compensates for this.

- Draw rectangles in real time camera around human faces.
- LBPH find its local structure by comparing each pixel to the neighboring pixels.
- Read training images for each person along with their labels, detect faces from eachimage and assign each detected face an integer label of the person it belongs.
- Train Opencv's LBPH recognizer by feeding it the database of images and finallydraw name of the predicted individual above face rectangle.
- Automation cars are an unusual product from the perspective of safety-relatedoutcomes.
- It gives signal if there is a obstacle comes near the safe distance of the car. So,reduced accident rates.

Future Scope

Face recognition can be used mostly for security purposes. Because of the wide range of commercial acceptance and law enforcement cases, face detection is increasing as a major researcharea. To remove the identification from something you know (such as a password) or something you have (such as a laptop or safety badge) and you actually increase it with it. So, people do not have to remember the passwords that they sometimes get pressured and sometimes they forget the password. There is no risk if someone forgets a password. Innovators are among the other ways of implementing facial recognition in the subcontinent and other transport outlets. They are looking to leverage this technology to use face like credit cards to pay transportation fees. Instead of going to buy a ticket, face-to-face detection will be run on a system and will charge the account owner.

Automation will be a crime using car and face detection. If an offense is committed by using face detection, the person will be recognized and will be punished for offenses with evidence. If all vehicles are unmanageable, then fewer accidents will happen, but it will be interesting to migrate to autonomous vehicles by sharing the road with humans and machines. Automatic vehicle meansvehicle without driver. Meaning of share car means less infrastructure and necessary infrastructure for transportation. Less infrastructures cost less money and driverless car means free driver from driving. The meaning of free drivers from driving means more time for the job and the cost is beneficial for both economies.

Source code for this

Face detection and identification code

Main.py

```
import cv2
from tkinter import *
from tkinter import messagebox
from PIL import Image, ImageTk
import numpy as np
import face_recognition
import os

path='C:/Users/Arpit Maurya/Desktop/Programs/car/1st/images'
```

```
images=[]
presonName=[]
myList=os.listdir(path)

for cu_Img in myList:
    currentImg=cv2.imread(f'{path}/{cu_Img}')
    images.append(currentImg)
    presonName.append(os.path.splitext(cu_Img)[0])

# ------------------------------------Face Registration Section---------
-----------------------------
if len(images)==0:

    root= Tk()

    root.geometry("500x250")
    root.title('Registration')
    root.configure(bg="#116562")

    def openingCamera():
        global entry
        name= entry.get()
        if(len(name)==0):
            messagebox.showwarning("Invalid Name Warning", "Please Enter
a Valid Name!... ")
        else:
            entry.pack_forget()
            label.pack_forget()
            B.pack_forget()
            root.title('Register Your Face')
            root.minsize(646,530)
            root.maxsize(646,530)
            root.configure(bg='#58F')
            face_cascade=cv2.CascadeClassifier('haarcascade_frontalface_d
efault.xml')
            cap= cv2.VideoCapture(0)
            if (cap.isOpened() == False):
```

```
                print("Unable to read camera feed")

            def captureImage():
                image=Image.fromarray(img1)
                personName=str(name)+'.jpg'
                image.save(f'C:/Users/Arpit
Maurya/Desktop/Programs/car/1st/images/{personName}')
                cap.release()
                cv2.destroyAllWindows()
                root.destroy()
                root.quit()
                messagebox.showinfo('Registasion SuccessFul', 'Face
Registasion Completed SuccessFully!!')

            def exitWindow():
                cap.release()
                cv2.destroyAllWindows()
                root.destroy()
                root.quit()

            f1=LabelFrame(root,bg='red')
            f1.pack()
            l1=Label(f1,bg='red')
            l1.pack()

            b1=Button(root,bg='green',fg='white',activebackground='white'
,activeforeground='green',text='REGISTER FACE
📷',relief=RIDGE,height=200,width=30,command=captureImage)
            b1.pack(side=LEFT,padx=60,pady=5)
            b2=Button(root,fg='white',bg='red',activebackground='white',a
ctiveforeground='red',text='LATER ❌
',relief=RIDGE,height=200,width=20,command=exitWindow)
            b2.pack(side=LEFT,padx=40,pady=5)

            while True:
                img=cap.read()[1]
                gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
```

```
                faces=face_cascade.detectMultiScale(gray,1.1,4,minSize=(6
0,60))
                for (x,y,w,h) in faces:
                    cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),1)
                img=cv2.flip(img,1)
                img1=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
                img=ImageTk.PhotoImage(Image.fromarray(img1))
                l1['image']=img
                root.update()
            cap.release()
    label=Label(root, text=" Enter your Name", font=("Courier 20
bold"),fg="white",bg='#116562',height=2)
    label.pack(pady=10,fill='x')
    entry= Entry(root, width= 40)
    entry.focus_set()
    entry.pack(pady=10)
    B=Button(root, text= "Next",width=
22,bg='green',fg='white',height=2,activebackground='white',activeforegrou
nd='red',font=("none 9 bold"), command= openingCamera)
    B.pack(padx=32,pady=10)
    root.mainloop()

    # ------------------------------ Face Recognition and Identification-
---------------------------------------
else:
    def faceEncodings(images):
        encodeList=[]
        for img in images:
            img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
            encode=face_recognition.face_encodings(img)[0]
            encodeList.append(encode)
        return encodeList
    encodeListKnown=(faceEncodings(images))
    print('All encoding complete!!! ')
    cap=cv2.VideoCapture(0)
    while True:
        ret, frame= cap.read()
```

```
        faces=cv2.resize(frame,(0,0), None,0.25,0.25)
        faces=cv2.cvtColor(faces,cv2.COLOR_BGR2RGB)
        facesCurrentframe=face_recognition.face_locations(faces)
        encodeCurrentFrame=face_recognition.face_encodings(faces,facesCur
rentframe)
        for encodeFace, faceLoc in
(zip(encodeCurrentFrame,facesCurrentframe)):
            matches=face_recognition.compare_faces(encodeListKnown,encode
Face)
            faceDis=face_recognition.face_distance(encodeListKnown,encode
Face)
            matchIndex=np.argmin(faceDis)
            if matches[matchIndex]:
                name=presonName[matchIndex].upper()
                y1,x2,y2,x1=faceLoc
                y1,x2,y2,x1=y1*4,x2*4,y2*4,x1*4
                cv2.rectangle(frame,(x1-10,y1-
10),(x2+5,y2+5),  (160,233,126),1)
                # cv2.rectangle(frame,(x1,y2-
30),(x2,y2+10),(255,255,255),cv2.FILLED)
                # cv2.putText(frame,name,(x1+6,y2-
6),cv2.FONT_HERSHEY_DUPLEX,1,(87,18,255),1)
                cv2.imshow('RECOGNIZING....',frame)
                cap.release()
                cv2.destroyAllWindows()
                # if cv2.waitKey(1) & 0xFF==ord('q'):
                #     break
                messagebox.showinfo('Access Granted', 'Hello! '+name+'
...\nWelcome To My Face Recognition System.')
        cv2.imshow('RECOGNIZING....',frame)
        if cv2.waitKey(1) & 0xFF==ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()
```