Smart Gate

Batch No: A-04

R.Hemalatha (164G1A0530)
K.Likith kumar (164G1A0549)
K.Keerthi Reddy (164G1A0537)
A.Ganesh (164G1A0526)

Project Guide:

Dr.T.Hithendra sarma, MTech., Ph.D.
Principal

Srinivasa Ramanujan Institute of Technology
Department of Computer Science & Engineering

Abstract

- **Smart Gate Technology** is based on Internet Of Things and Deep Learning. It involves face recognition and sensor kit. Smart gate is used to allow only valid persons.

Literature Survey:

Existing Systems:

Normal gate:

- It opens the gate without any authorization

Automatic gate:

- Just as sensor without validation

Limitations:

- No security
- Time Tacking Process

Proposed System

- Sensor with validation check
  - Checks the image with existing dataset
  - If it is exist then it opens the gate otherwise it does not open the gate

Face recognition:

A **facial recognition system** is a technology capable of identifying or verifying a person from a digital image as a source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database.

Cntd…

- Face recognition is used to check that a person has permission to enter the organization or not. The output of facial recognition is send to sensor kit as binary value. if it is true then it opens the gate otherwise it won’t open the gate.

Requirements

Hardware Requirements:

RAM : 4GB

Processor : intel core i3

Face recognition sensor

Software requirements:

Python IDE with cmd /Anaconda platform(Jupyter)

Opencv

Open CV

- **OpenCV** (*Open source computer vision*) is a library of programming functions mainly aimed at real-time computer vision.

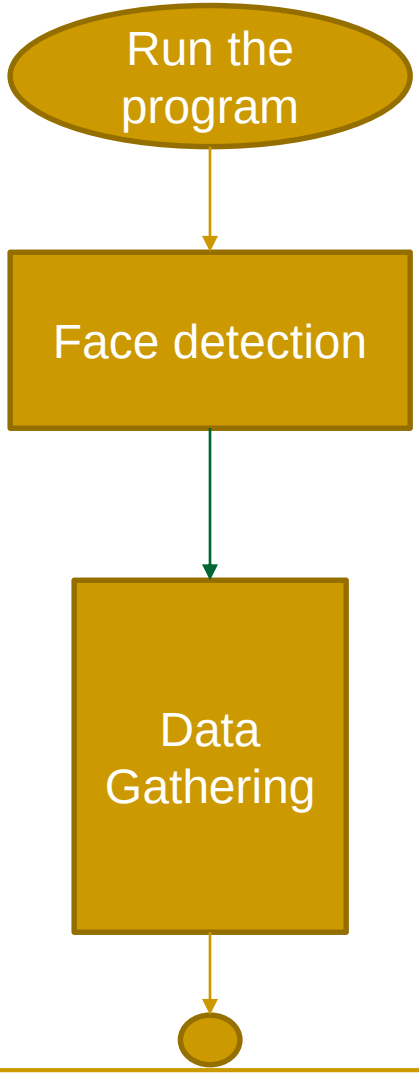
- Computer vision is a field of study which encompasses on how computer see and understand digital images and videos.

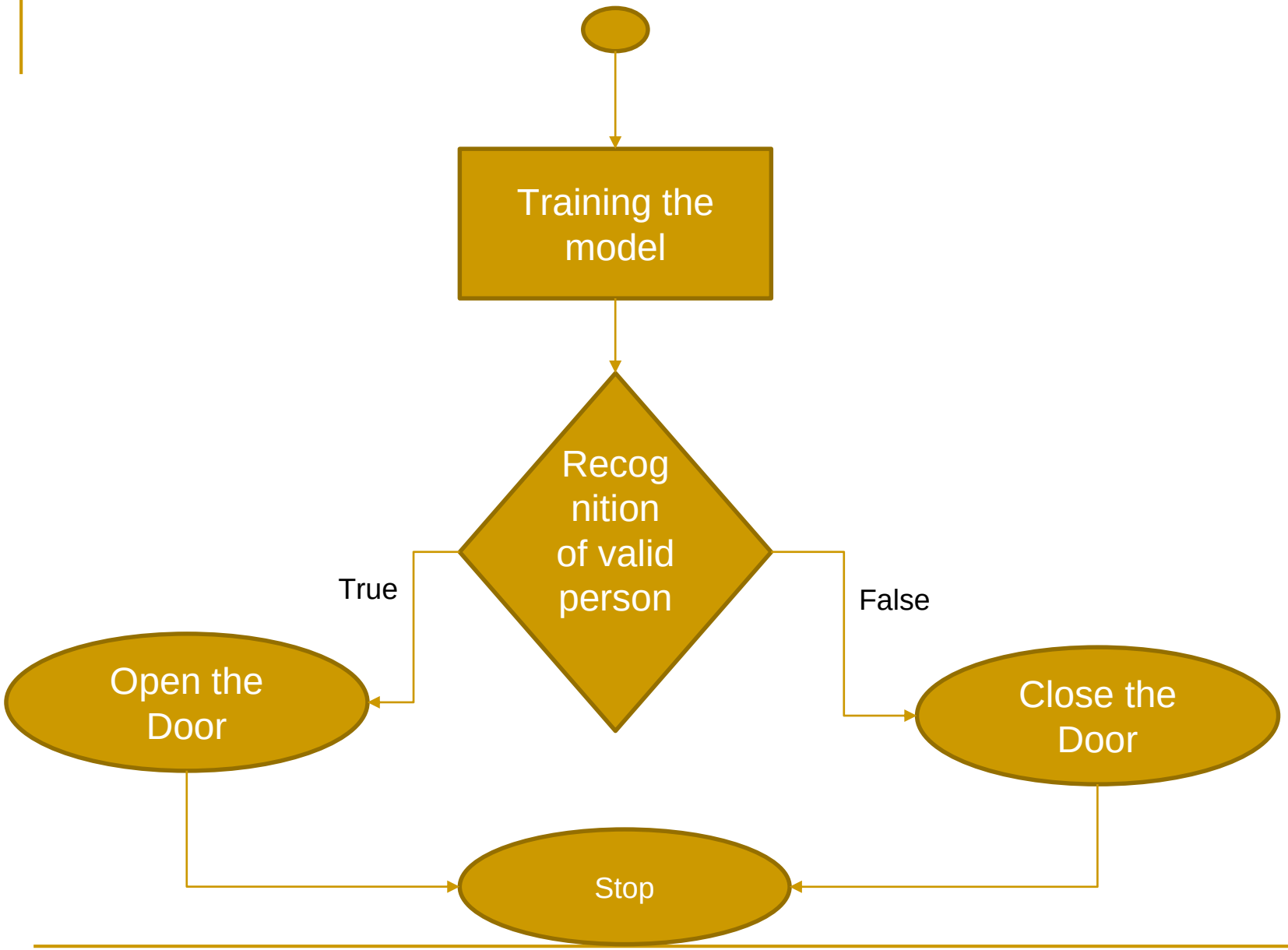
Install opencv from prebuilt Libraries

- Install python-2.7.x

- install Numpy, which can be done with pip, the Python package manager, by sending the following command on the command line:
  - pip install numpy

Planning:




Training the model
Recog nition of valid person
True
False
Open the Door
Close the Door
Stop

Face Detection

- we need to import the **cv2** module, which will make available the functionalities needed to read the original image and to convert it to gray scale.
  - import cv2

- Syntax for xml file:
  - faceCascade = cv2.CascadeClassifier('Cascades/haarcascade_frontalface_default.xml')

Cntd...

- Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images.
- The most common way to detect a face by using the "Haar Cascade classifier".
- **Haar Cascade** is a machine learning object detection **algorithm used** to identify objects in an image or video.

Cntd...

- To read the original image, We call the **imread** function of the **cv2** module, passing as input the path to the image, as a string.
- Image=cv2.imread(' C:\Users\R hemalatha/164G1A0530.jpg')
- To read the image through video,we call videocapture function.
- cap = cv2.VideoCapture(0)

Image colour

- To convert our original image from the **BGR** color space to **gray**, we use the code **COLOR_BGR2GRAY**.
  - gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)

Detect Multi Scale:

- we must call our classifier function, passing it some very important parameters, as scale factor, number of neighbors and minimum size of the detected face.

```
faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.2,
        minNeighbors=5,
        minSize=(20, 20)
    )
```

where facecascade is the xml file,

- **gray** is the input grayscale image

Cntd...

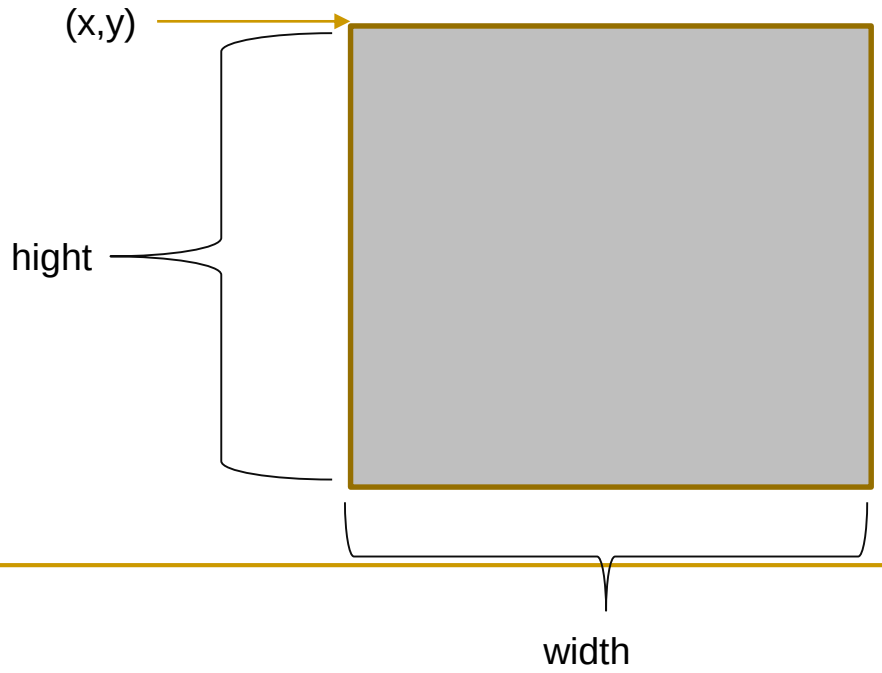
- **scaleFactor** is the parameter specifying how much the image size is reduced at each image scale. It is used to create the scale pyramid.

- **minNeighbors** is a parameter specifying how many neighbors each candidate rectangle should have, to retain it. A higher number gives lower false positives.

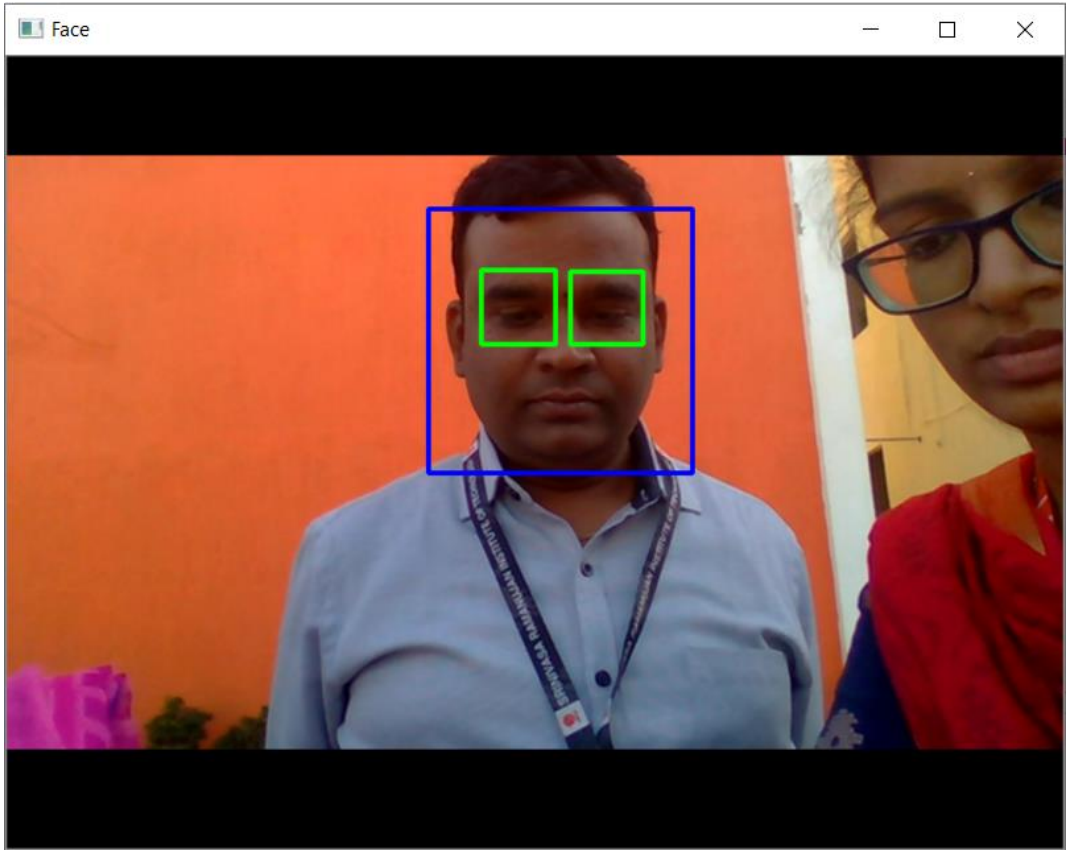
- **minSize** is the minimum rectangle size to be considered a face.

Image Shape

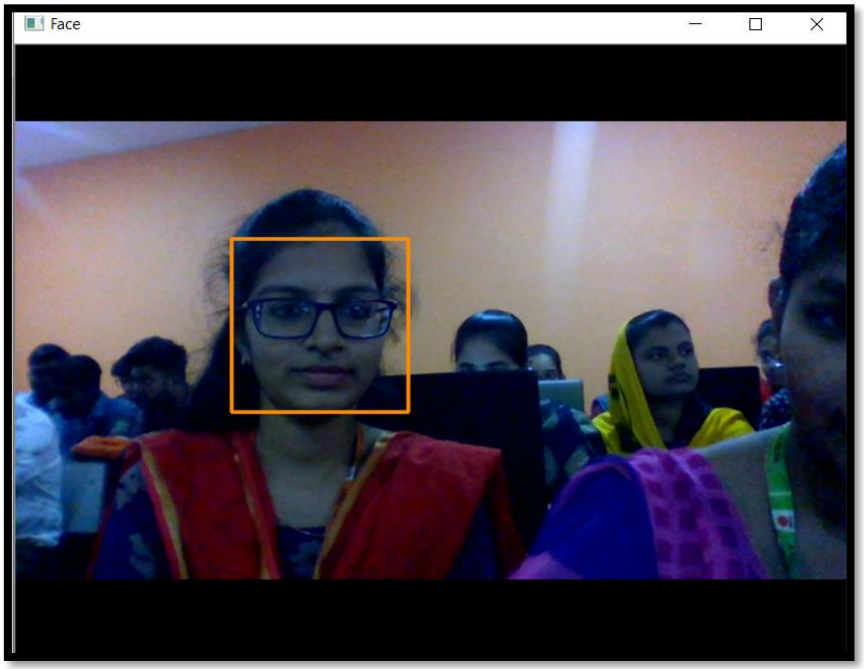
- To mark the faces in the image, using, for example, a blue rectangle.
  - cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]



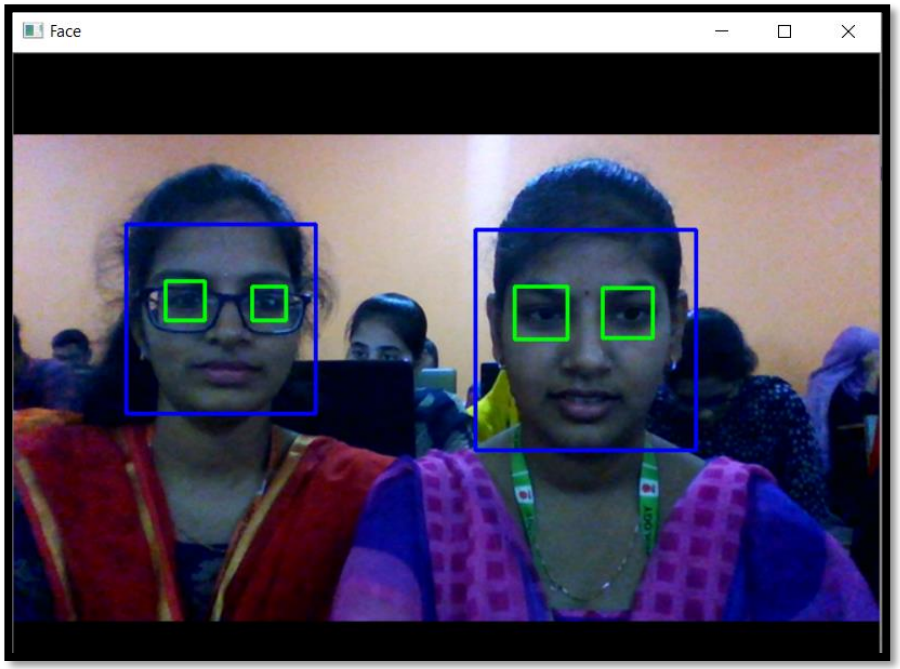
Output face with eye capture



Outputs:

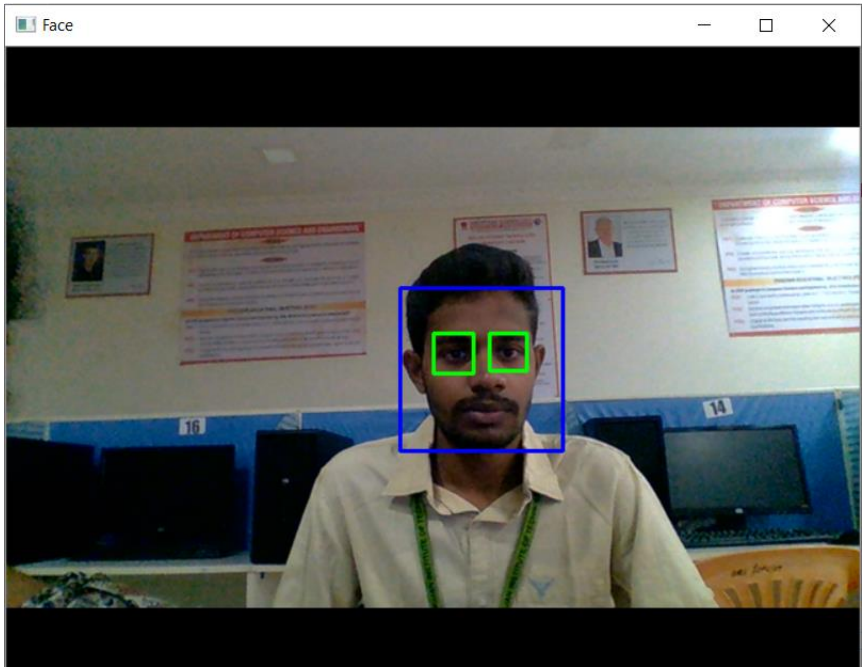


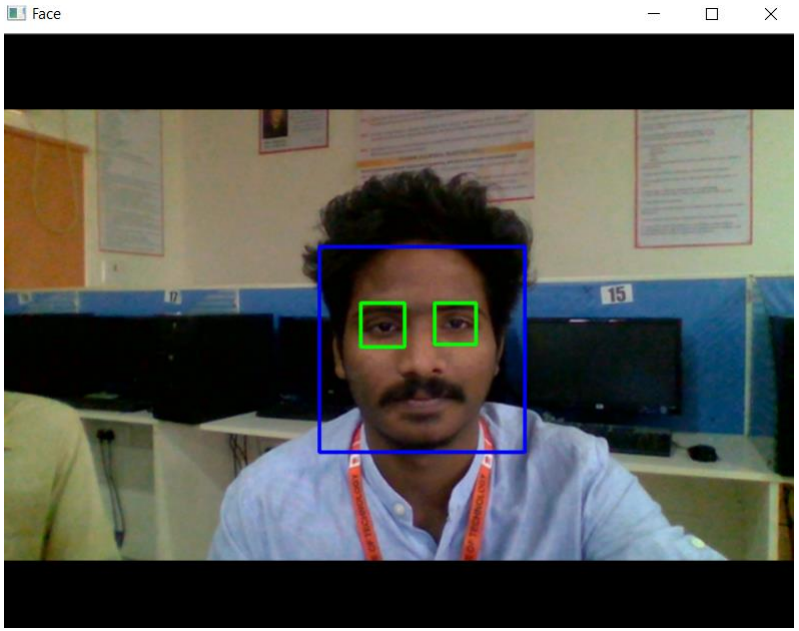
Face detect



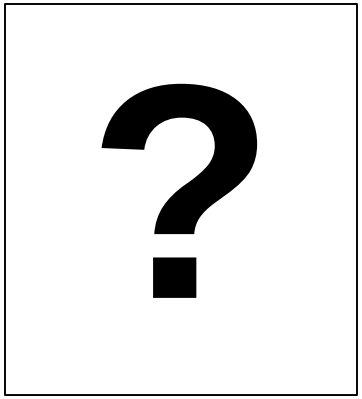
Multiple faces

Cntd..





Queries



THANK YOU