PYTHON MINI PROJECT

FACE AND EYE DETECTOR

ABSTRACT

Face and Eye detector is an optical recognition project built in Python and involves usage of Image Processing module like OpenCV and some Machine Learning concepts to implement it. It can be used in various field wherein Optical Sensors are required as discussed in upcoming sections.
Python is used as base programming language to code on because of it's ease to implement and run robust modules and is almost compatible in across most platforms.
The program Face detector can be used for many purposes such as for attendance in offices , schools etc.
It can be used as a password which is nowadays a very famous and is available in every new model of smartphones and now even in laptops.
Facebook uses face recognition technology to automatically recognize when Facebook members appear in photos. This makes it easier for people to find photos they are in and can suggest when particular people should be tagged in photos.
Eye tracking has also been introduced to the human-computer interaction and gaming industry which now enables for instance game designers to get a better understanding of the game experience so that it is somewhat possible to control the experience and create features that push the boundaries of reality even more.In the time to come, it will most likely even be possible to personalize the game's development in regard to pupil dilation of the player and the gamer will be able to control the game with eye movements.

SYSTEM REQUIREMENTS

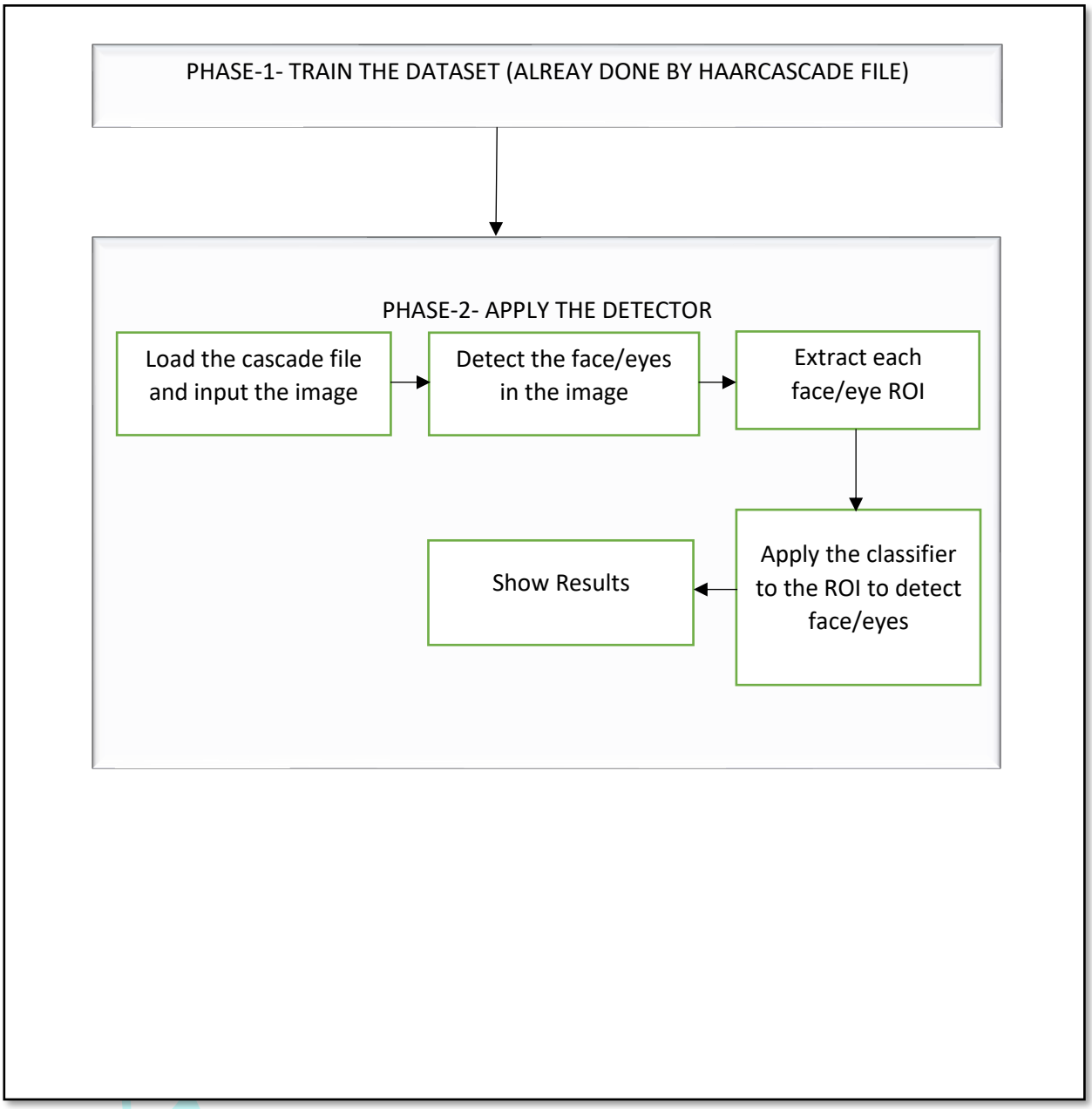
➢ HARDWARE REQUIREMENTS:

1. CPU With Atleast 35hz Refresh Rate
2. GPU(Optional)
3. Intel Core I3(Minimum)
4. Camera (Built In Or External)

➢ SOFTWARE/MODULE REQUIREMENTS:

1. Python 3.6(Minimum)
2. Tkinter (frontend)
3. Python Modules (used in HaarCascade file):
    a) Cv2 (image processing)
    b) Numpy (for calculations)
    c) Os (for OS access)
    d) Sys (for System details)
    e) Datetime (for date and time)
    f) Keras (for training data)
    g) Tensorflow (for training data)

ARCHITECTURE DIAGRAM



CODING

```
import cv2
import tkinter as tk


def facedetect():
    face_Cascade = cv2.CascadeClassifier("hc_ff.xml")
    cap = cv2.VideoCapture(0)
    cap.set(10, 100)
    while cap.isOpened():
        _ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_Cascade.detectMultiScale(gray, 1.1, 4)
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv2.putText(frame, "FACE", (x, y),
                        cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2,
2)
        cv2.imshow("Face Detect", frame)
        if cv2.waitKey(40) == 27:
            break
    cap.release()
    cv2.destroyAllWindows()


def eyedetect():
    eye_Cascade = cv2.CascadeClassifier("eye.xml")
    cap = cv2.VideoCapture(0)
    cap.set(10, 100)
    while cap.isOpened():
        _ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        eyes = eye_Cascade.detectMultiScale(gray, 1.1, 4)
        for (x, y, w, h) in eyes:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
            cv2.putText(frame, "EYE", (x, y),
                        cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2,
2)
        cv2.imshow("Eye Detect", frame)
        if cv2.waitKey(40) == 27:
            break
    cap.release()
    cv2.destroyAllWindows()
```

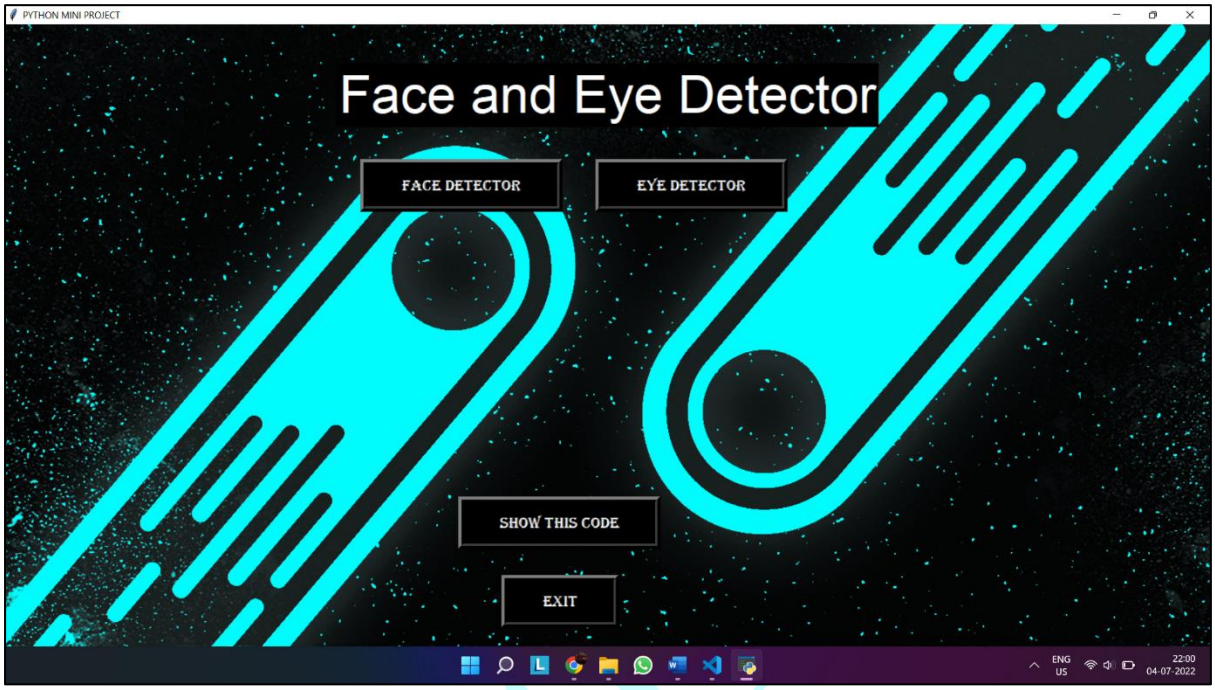
```
def show_this_code():
    root = tk.Tk()
    root.title("CODE")
    root.geometry("1920x1080")
    bg = tk.PhotoImage(file="imgg.png")
    my_label = tk.Label(root)
    my_label.place(x=0, y=0, relwidth=1, relheight=1)

    my_text = tk.Label(root, text="""import cv2
import tkinter as tk


def facedetect():
    face_Cascade = cv2.CascadeClassifier("hc_ff.xml")
    cap = cv2.VideoCapture(0)
    cap.set(10, 100)
    while cap.isOpened():
        _ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_Cascade.detectMultiScale(gray, 1.1, 4)
        for (x,y,w,h) in faces:
            cv2.rectangle(frame, (x,y), (x+w, y+h), (0,255,0), 2)
            cv2.putText(frame, "FACE", (x,y),
cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0),2, 2)
        cv2.imshow("Face Detect", frame)
        if cv2.waitKey(40) == 27:
            break
    cap.release()
    cv2.destroyAllWindows()


def eyedetect():
    eye_Cascade = cv2.CascadeClassifier("eye.xml")
    cap = cv2.VideoCapture(0)
    cap.set(10, 100)
    while cap.isOpened():
        _ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        eyes = eye_Cascade.detectMultiScale(gray, 1.1, 4)
        for (x,y,w,h) in eyes:
            cv2.rectangle(frame, (x,y), (x+w, y+h), (0,255,0), 2)
            cv2.putText(frame, "EYE", (x,y),
cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0),2, 2)
        cv2.imshow("Eye Detect", frame)
        if cv2.waitKey(40) == 27:
            break
```

```
    cap.release()
    cv2.destroyAllWindows()
def show_this_code():
    ...

root = tk.Tk()
root.title("PYTHON MINI PROJECT")
root.geometry("1920x1080")
bg = tk.PhotoImage(file="imgg.png")
my_label = tk.Label(root, image=bg)
my_label.place(x=0, y=0, relwidth=1, relheight=1)

my_text = tk.Label(root, text="Face and Eye Detector",
font=("Jocker", 50), fg="white",bg="black")
my_text.pack(pady=50)

my_button1 = tk.Button(my_label, text="Face Detector"
,command=facedetect ,fg="white", bg="black", font=("Algerian" ,15) ,
padx=40 , pady=10 , borderwidth=7)
my_button1.grid(row=0, column=1, padx=10 , pady=250)
my_button1.place(x=450,y=170)

my_button2 = tk.Button(my_label, text="Eye Detector"
,command=eyedetect ,fg="white", bg="black", font=("Algerian" ,15) ,
padx=40 , pady=10 , borderwidth=7)
my_button2.grid(row=0, column=1, padx=10 , pady=250)
my_button2.place(x=750,y=170)

my_button3 = tk.Button(my_label, text="Show This
Code",command=show_this_code,fg="white", bg="black",font=("Algerian"
,15), padx=40 , pady=10 , borderwidth=7)
my_button3.grid(row=0, column=3, padx=10 , pady=250 )
my_button3.place(x=575,y=600)

my_button4 = tk.Button(my_label,
text="Exit",command=root.quit,fg="white",
bg="black",font=("Algerian" ,15), padx=40 , pady=10 , borderwidth=7)
my_button4.grid(row=0, column=4, padx=10 , pady=250 )
my_button4.place(x=630,y=700)

root.mainloop()
exit()
    """, font=("Jocker", 10), fg="white", bg="black")
    my_text.pack(pady=10)
```

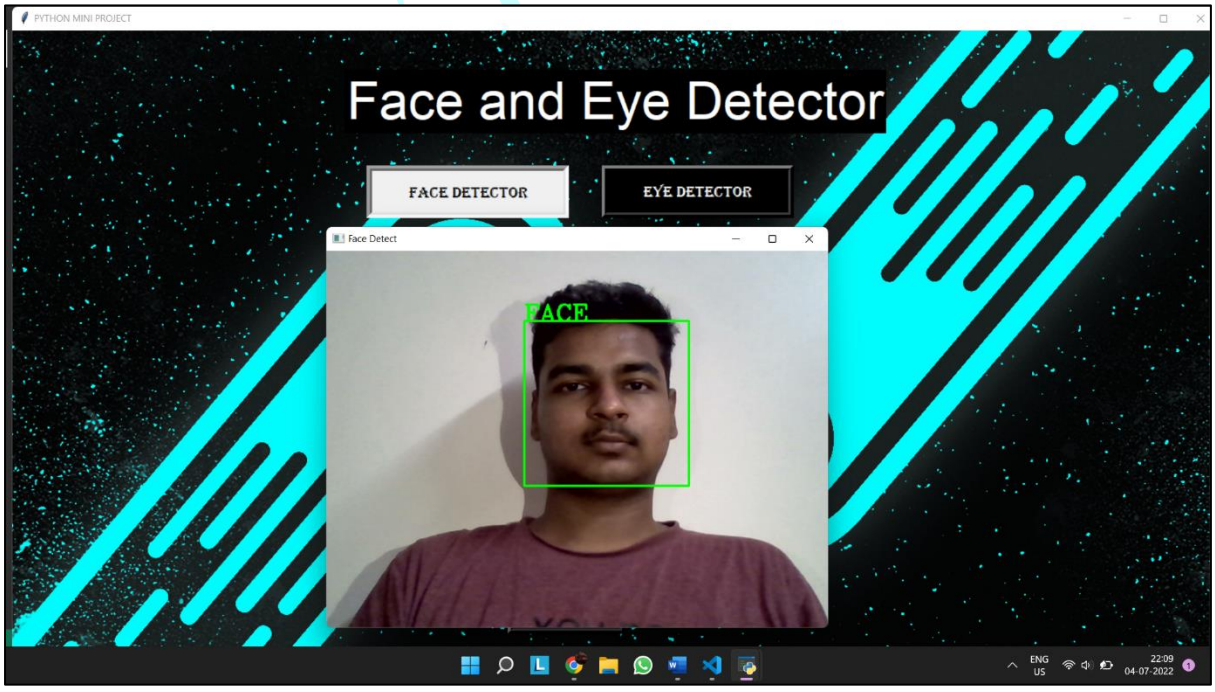
```
root = tk.Tk()
root.title("PYTHON MINI PROJECT")
root.geometry("1920x1080")
bg = tk.PhotoImage(file="imgg.png")
my_label = tk.Label(root, image=bg)
my_label.place(x=0, y=0, relwidth=1, relheight=1)

my_text = tk.Label(root, text="Face and Eye Detector",
                   font=("Jocker", 50), fg="white", bg="black")
my_text.pack(pady=50)

my_button1 = tk.Button(my_label, text="Face Detector",
command=facedetect,
                       fg="white", bg="black", font=("Algerian",
15), padx=40, pady=10, borderwidth=7)
my_button1.grid(row=0, column=1, padx=10, pady=250)
my_button1.place(x=450, y=170)

my_button2 = tk.Button(my_label, text="Eye Detector",
command=eyedetect, fg="white",
                       bg="black", font=("Algerian", 15), padx=40,
pady=10, borderwidth=7)
my_button2.grid(row=0, column=1, padx=10, pady=250)
my_button2.place(x=750, y=170)

my_button3 = tk.Button(my_label, text="Show This Code",
command=show_this_code,
                       fg="white", bg="black", font=("Algerian",
15), padx=40, pady=10, borderwidth=7)
my_button3.grid(row=0, column=3, padx=10, pady=250)
my_button3.place(x=575, y=600)

my_button4 = tk.Button(my_label, text="Exit", command=root.quit,
fg="white",
                       bg="black", font=("Algerian", 15), padx=40,
pady=10, borderwidth=7)
my_button4.grid(row=0, column=4, padx=10, pady=250)
my_button4.place(x=630, y=700)

root.mainloop()
exit()
```

OUTPUT

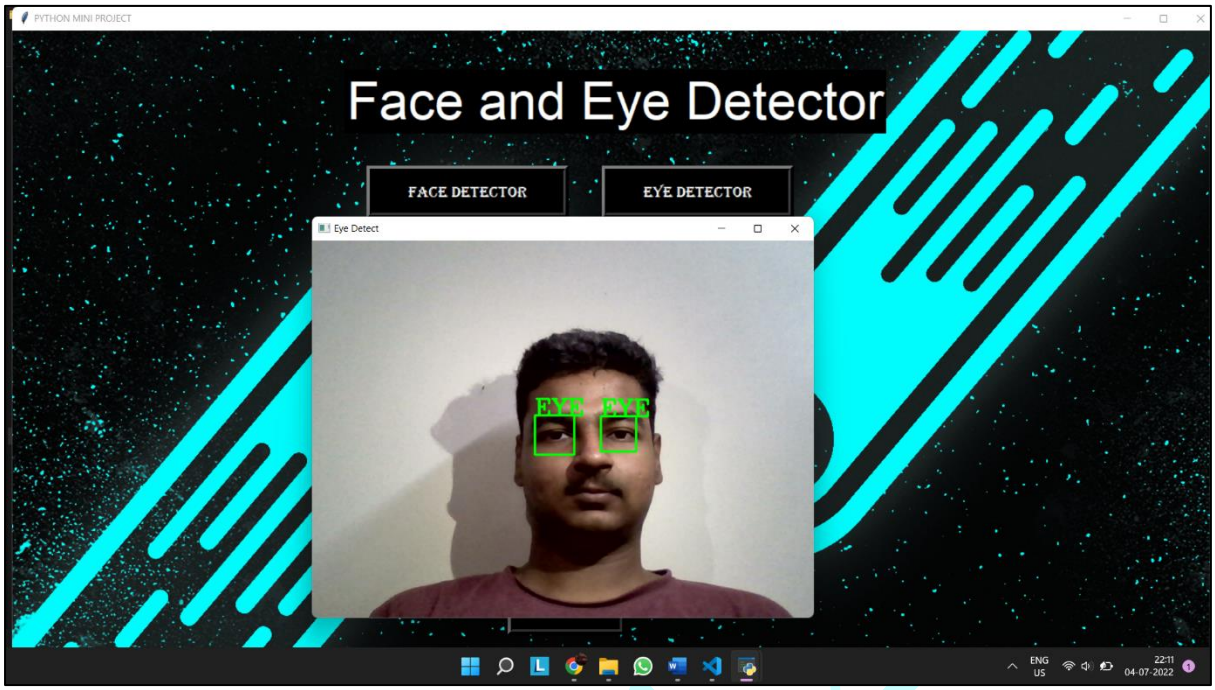
Frontend using Tkinter



Face Detection



Eye Detection



Conclusion and future developments

In this mini project, we can see that with the help of few libraries and a small python code, we are able to detect face and eyes. This model can be extended to various uses as given in the abstract.

It is also built on Open Source Python Packages, which are always subjected to updates, which makes the project vulnerable to be updated once in a while, which indeed is a good aspect for the user.

Further, in the future, this project can be implemented in various fields such as:

1. Smarter advertising

Face recognition has the ability to make advertising more targeted by making educated guesses at people's age and gender. Companies like Tesco are already planning on installing screens at gas stations with face recognition built in. It's only a matter of time before face-recognition becomes an omni-present advertising technology.

2. Diagnose diseases

Face recognition can be used to diagnose diseases that cause detectable changes in appearance. As an example, the National Human Genome Institute Research Institute, uses face recognition to detect a rare disease
called DiGeorge syndrome, in which there is a portion of the 22nd chromosome missing. Face recognition has helped diagnose the disease in 96% of cases. As algorithms get even more sophisticated, face recognition will become
an invaluable diagnostic tool for all sorts of conditions.

3. Validate identity at ATMs

It seems likely that face scans will eventually replace ATM cards completely. But in the meantime, face recognition can be used to make sure that individuals using ATMs cards are who they say they are. Face recognition is currently being used at ATMs in Macau to protect peoples' identities.

The scope of this project is limitless for the future use and it'll be really intriguing to see what the future holds for this project and what developments would be needed to fit in for the requirements at that point of time.