Face Detection System

Using Python

Libraries

OBJECTIVE :

The objective of a face detection system is to identify and locate human faces within an image , video, Web cam. This technology is commonly used in various applications, such as security systems, photography, video analysis, and augmented reality. The primary goals include improving security, enhancing user experience, and automating processes that involve facial recognition.

DEFENITION :

a computer technology being used in a variety of applications that identifies human faces in digital images. Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene.

Python Libraries used :

1)OpenCV(computer Vision).

   Version -->1.21.1

• OpenCV is a Python library that allows you to perform image processing and computer vision tasks.

2)NumPy.(Numerical Python)

   Version -->4.5.3.56

• a library consisting of multidimensional array objects and a collection of routines for processing those arrays.

3)pip.( Preferred Installer Program)

   Version--->21.2.1

•  It is a command-line utility that installs, reinstalls, or uninstalls PyPI packages with one simple command: pip.

Uses :

• To identify Criminal in Faster investigation Times .

• To capture Images.

• To identify passenger in airport.

• In video Surveillance.

• In office(Bio metric).

• For FaceId (lock screen in mobile or any other device).

Advantages of face detection System:-

• Helps find missing people.

• Protects businesses against theft.

• Strengthens security measures.

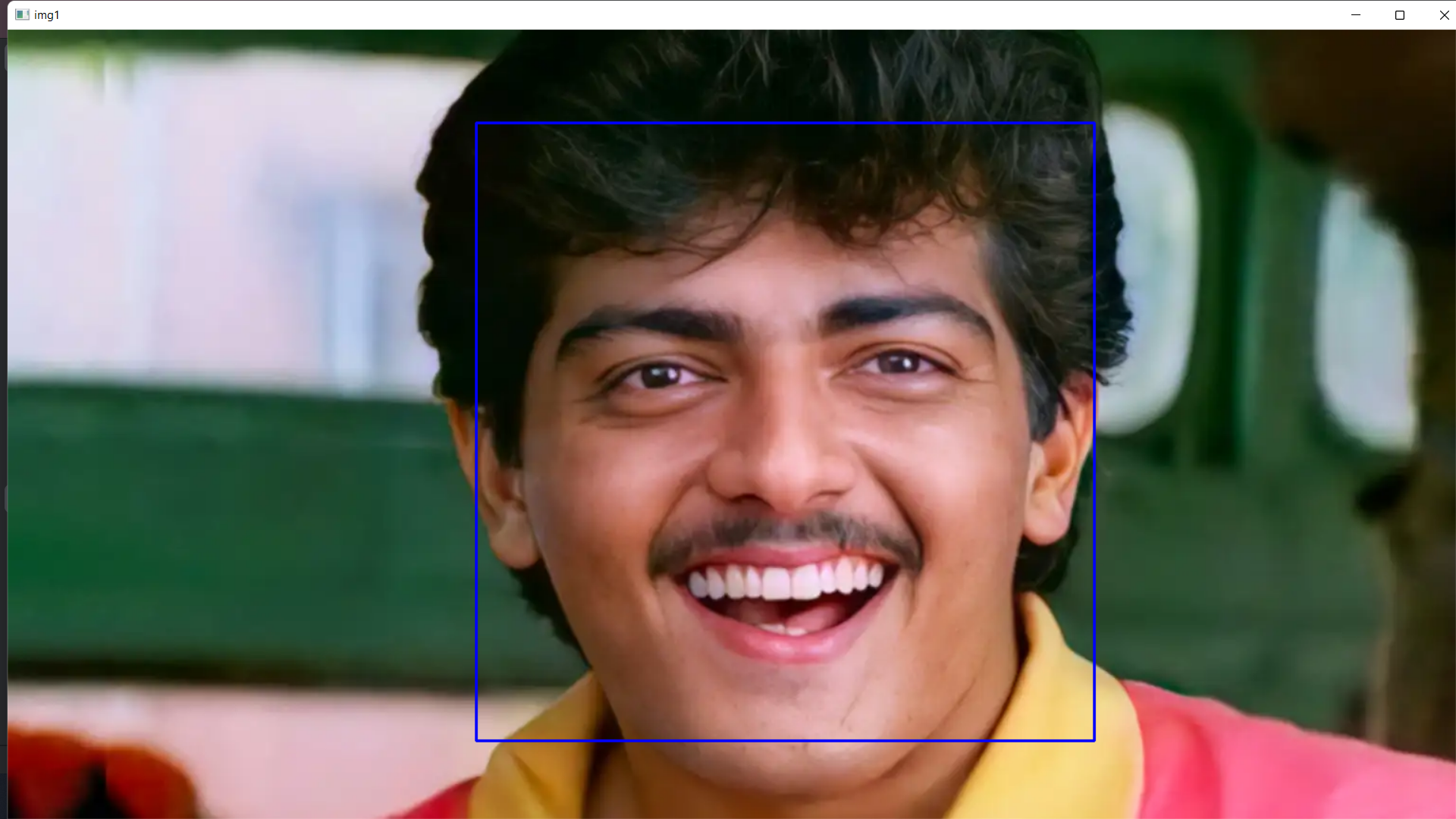
• Reduces the number of touchpoints.

• Improves photo organization.

Source code:

(i)Input to detect face in image:

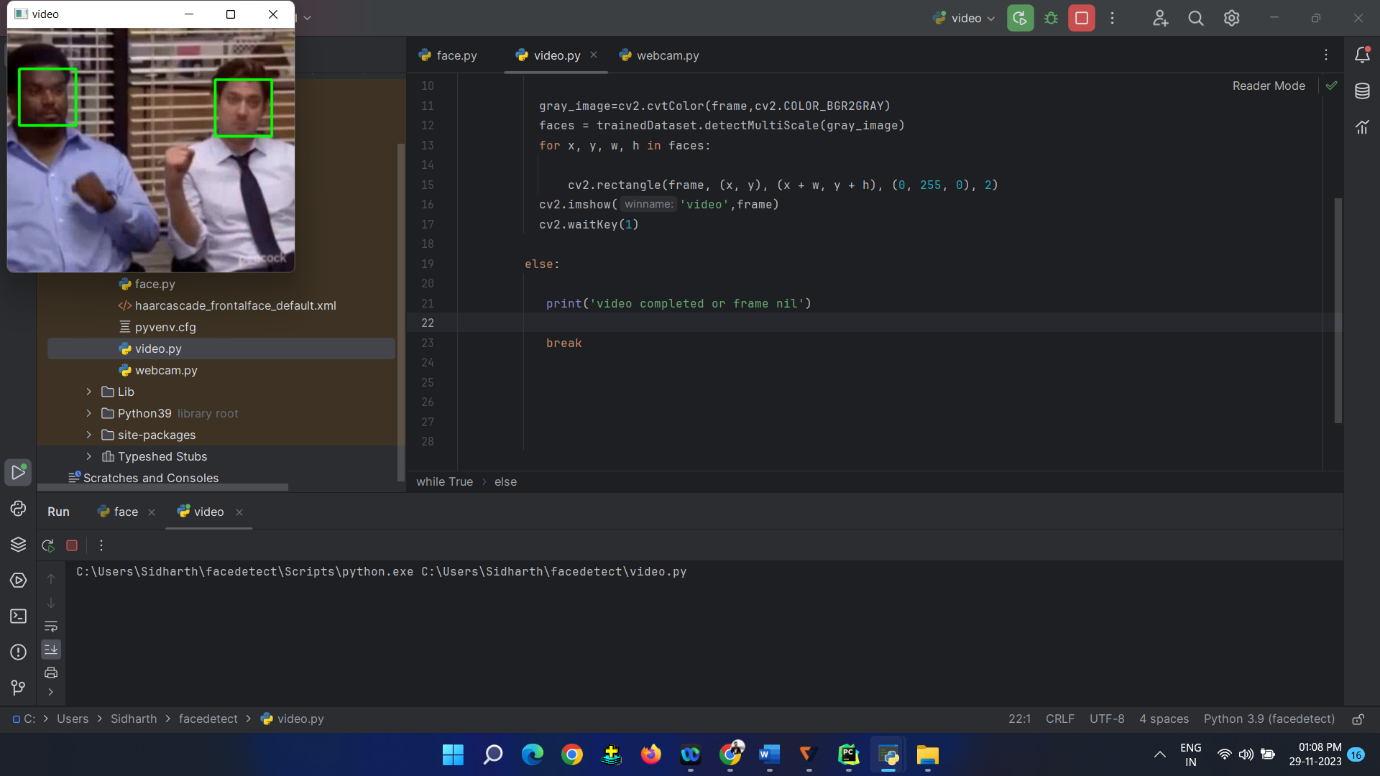
#Import openCV to perform image processing  
import cv2  
  
trainedDataset=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  
#haarcascade\_frontalface\_default it was designed by openCV.  
#it is used to detect front view of the face  
  
img=cv2.imread('images/img1.jpg')  
#imread() is an built in function.it is used to read the image.  
  
#convert the colored image into gray color.  
gray=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)  
  
#detecting the face.this function is suitable for detecting multiple or single image  
faces=trainedDataset.detectMultiScale(gray)  
print(faces)  
#printing co-ordinates of the image.  
  
for x,y,w,h in faces:  
  
 cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)  
  
#imshow() is the builtin function is used to show the image.  
cv2.imshow('img1',img)  
#waitkey() is the builtin function is used to display the result.  
cv2.waitKey()

OUTPUT: 

(ii)Input to detect face in video:

mport cv2  
  
trainedDataset=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  
  
video=cv2.VideoCapture('videos/vid1.mp4')  
while True:  
  
 success,frame=video.read()  
 if success==True:  
  
 gray\_image=cv2.cvtColor(frame,cv2.COLOR\_BGR2GRAY)  
 faces = trainedDataset.detectMultiScale(gray\_image)  
 for x, y, w, h in faces:  
  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
 cv2.imshow('video',frame)  
 cv2.waitKey(1)  
   
 else:  
  
 print('video completed or frame nil')  
  
 break

OUTPUT:



(iii)Input to detect face in Webcam:

import cv2  
  
trainedDataset=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  
video=cv2.VideoCapture(0)  
  
while True:  
 success,frame=video.read()  
   
 if success==True:  
 gray\_image=cv2.cvtColor(frame,cv2.COLOR\_BGR2GRAY)  
 faces = trainedDataset.detectMultiScale(gray\_image)  
 for x, y, w, h in faces:  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
 cv2.imshow('video',frame)  
 cv2.waitKey(1)  
  
 else:  
 print('video completed or frame nil')  
  
 break

OUTPUT:

