SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

FACULTY OF ENGINEERING & TECHNOLOGY

(Formerly SRM University, Under section 3 of UGC Act, 1956)

S.R.M. NAGAR, KATTANKULATHUR - 603 203,
KANCHEEPURAM DISTRICT

SCHOOL OF COMPUTING
DEPARTMENT OF NETWORKING AND COMMUNICATIONS

Course Code: 18CSE305J

Course Name: Artificial Intelligence

Course Project

Title: LIVENESS DETECTION

Team Members:

1. RA1911030010001 – Aditya Semwal
2. RA1911030010017 – Sayak Roy
3. RA1911030010033 – Anurag Kumar
4. RA1911030010027 – Mayank Singh

Date: 11/04/2022

Title: LIVENESS DETECTION

Problem Statement:

How do I spot real versus fake faces?

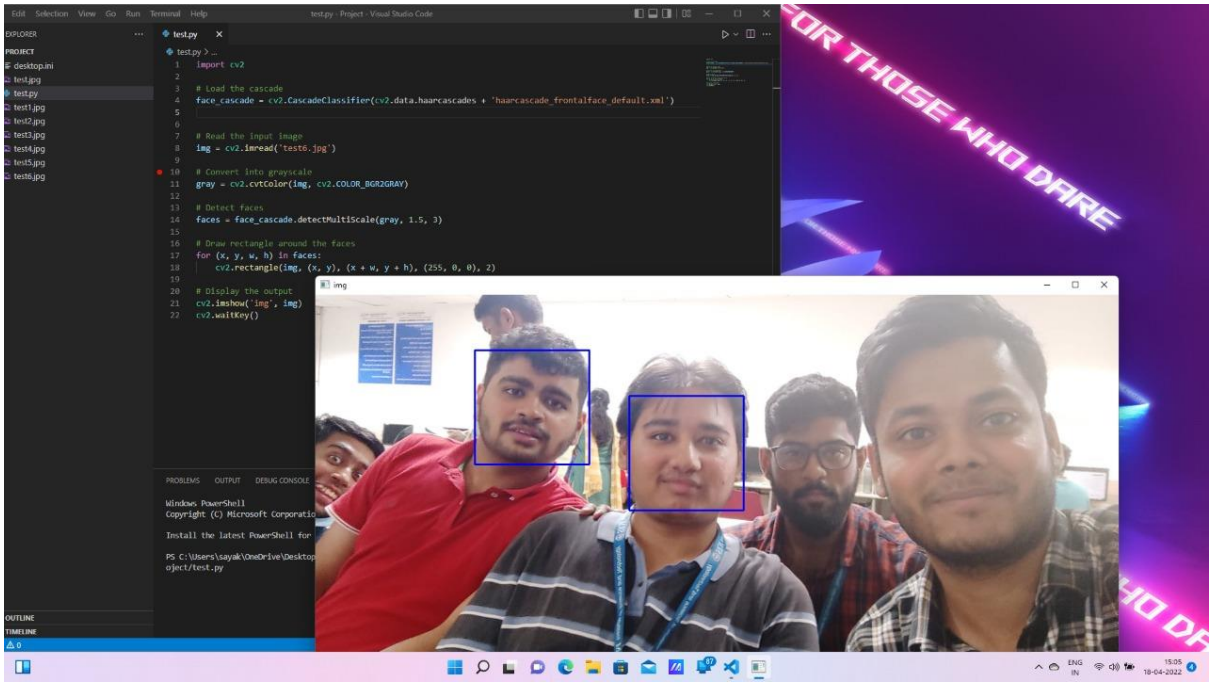
Consider what would happen if a nefarious user tried to *purposely circumvent your face recognition system*.

Such a user could try to **hold up a photo of another person.** Maybe they even **have a photo or video on their smartphone that they could hold up to the camera** responsible for performing face recognition (such as in the image at the top of this post).

In those situations it's entirely possible for the face held up to the camera to be correctly recognized…but ultimately leading to an unauthorized user bypassing your face recognition system!

How would you go about spotting these "fake" versus "real/legitimate" faces? How could you apply anti-face spoofing algorithms into your facial recognition applications?

The answer is to apply ***liveness detection*** with OpenCV which is exactly what we have covered.


test.py - Project - Visual Studio Code
import cv2
# Load the cascade
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
# Read the input image
img = cv2.imread('test6.jpg')
# Convert into grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Detect faces
faces = face_cascade.detectMultiScale(gray, 1.5, 3)
# Draw rectangle around the faces
for (x, y, w, h) in faces:
cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
# Display the output
cv2.imshow('img', img)
cv2.waitKey()
img
OR THOSE WHO DARE