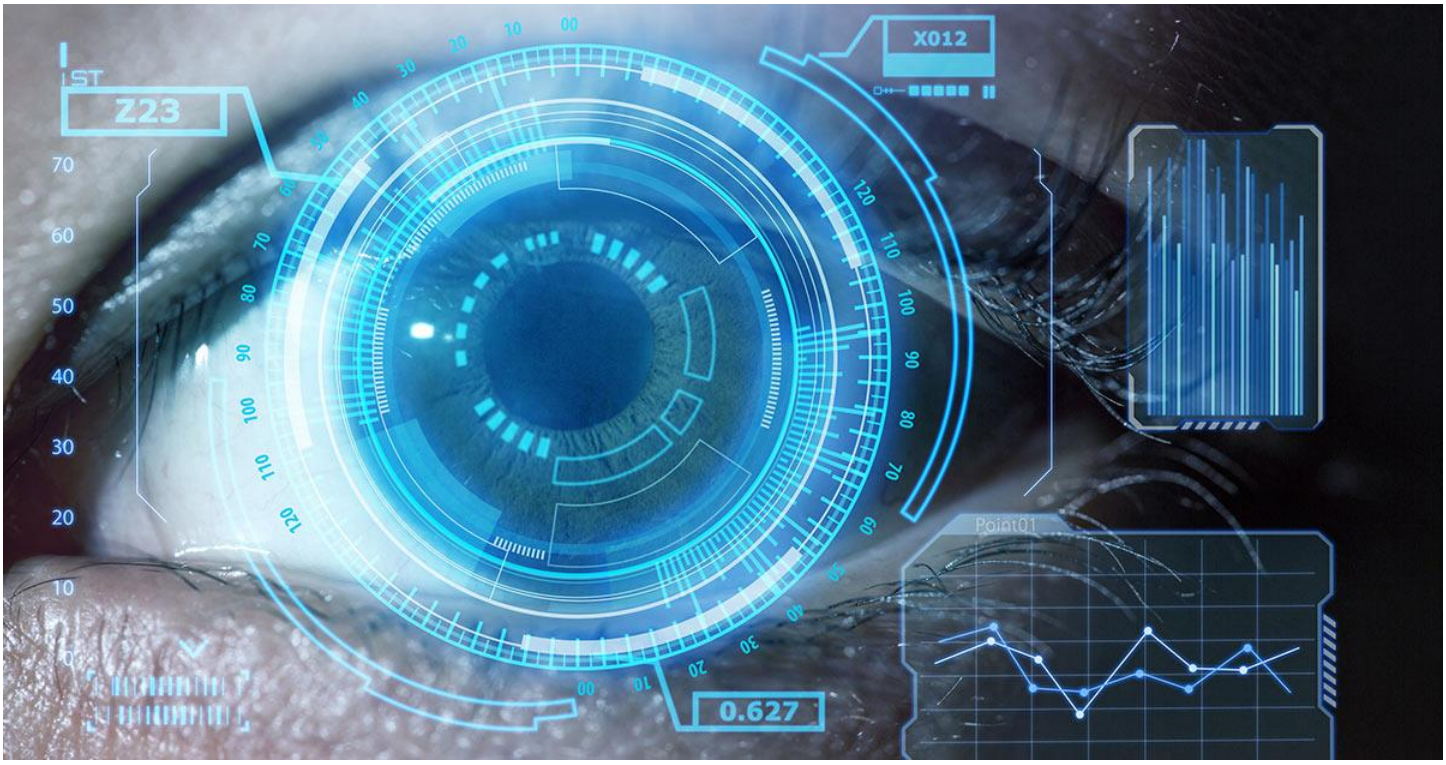
“COMPUTER VISION”

ENG: AHMED MUBARAK

01020451375



SESSION NO.“10”

- PROJECTS

1. Face, Smile and Eye Detection
2. Face dataset
3. Face training
4. Face recognition

ENG.AHMED MUBARAK

01020451375

1. Face detection from image

```
import cv2
from os import path

khan = cv2.imread('images\khan.jpg')
kids = cv2.imread('images\kids.jpg')


xml_classifier = path.join(path.dirname(cv2.__file__),
                           "data", "haarcascade_frontalface_default.xml")


def detect_faces(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    face_calssifier = cv2.CascadeClassifier(xml_classifier)
    rects = face_calssifier.detectMultiScale(image=gray,
                                             scaleFactor=1.15,
                                             minNeighbors=5,
                                             minSize=(30, 30))

    return rects


def draw(image, rects, title=None):
    print("=" * 30)
    print("i found {} people.".format(len(rects)).title())
    print("=" * 30)

    for x, y, w, h in rects:
        cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)

    if title:
        cv2.imshow(title, image)
        cv2.waitKey(0)


cv2.imshow("Khan", khan)
cv2.waitKey(0)
draw(khan, detect_faces(khan), "Khan")

cv2.imshow("Kids", kids)
cv2.waitKey(0)
draw(kids, detect_faces(kids), "Kids")
```

2. Face, smile and eye detection “real time”

```
import numpy as np
import cv2

# multiple cascades: https://github.com/Itseez/opencv/tree/master/data/haarcascades
faceCascade = cv2.CascadeClassifier('Cascades/haarcascade_frontalface_default.xml')
eyeCascade = cv2.CascadeClassifier('Cascades/haarcascade_eye.xml')
smileCascade = cv2.CascadeClassifier('Cascades/haarcascade_smile.xml')

cap = cv2.VideoCapture(0)
cap.set(3, 640)  # set Width
cap.set(4, 480)  # set Height

while True:
    ret, img = cap.read()
   # img = cv2.flip(img, -1)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.3,
        minNeighbors=5,
        minSize=(30, 30)
    )

    for (x, y, w, h) in faces:
        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]

        eyes = eyeCascade.detectMultiScale(
            roi_gray,
            scaleFactor=1.5,
            minNeighbors=5,
            minSize=(5, 5),
        )

        for (ex, ey, ew, eh) in eyes:
            cv2.rectangle(roi_color, (ex, ey),
                          (ex + ew, ey + eh), (0, 255, 0), 2)

        smile = smileCascade.detectMultiScale(
            roi_gray,
            scaleFactor=1.5,
            minNeighbors=15,
            minSize=(25, 25),
        )

        for (xx, yy, ww, hh) in smile:
```

```
            cv2.rectangle(roi_color, (xx, yy),
                          (xx + ww, yy + hh), (0, 255, 0), 2)

        cv2.imshow('video', img)

    k = cv2.waitKey(30) & 0xff
    if k == 27:  # press 'ESC' to quit
        break

cap.release()
cv2.destroyAllWindows()
```

3. Face dataset

```
import cv2
import os

cam = cv2.VideoCapture(1)
cam.set(3, 640) # set video width
cam.set(4, 480) # set video height

face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# For each person, enter one numeric face id
face_id = input('\n enter user id end press <return> ==>  ')

print("\n [INFO] Initializing face capture. Look the camera and wait ...")
# Initialize individual sampling face count
count = 0

while(True):

    ret, img = cam.read()
  #  img = cv2.flip(img, -1) # flip video image vertically
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_detector.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:

        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
        count += 1

        # Save the captured image into the datasets folder
        cv2.imwrite("dataset/User." + str(face_id) + '.' + str(count) + ".jpg", gray[y:y+h
,x:x+w])

        cv2.imshow('image', img)

    k = cv2.waitKey(100) & 0xff # Press 'ESC' for exiting video
```

```
    if k == 27:
        break
    elif count >= 30: # Take 30 face sample and stop video
         break

# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
cam.release()
cv2.destroyAllWindows()
```

4. Face training

```
import cv2
import numpy as np
from PIL import Image
import os

# Path for face image database
path = "dataset"

recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");

# function to get the images and label data
def getImagesAndLabels(path):

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []

    for imagePath in imagePaths:

        PIL_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')

        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)

        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)

    return faceSamples,ids

print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))
```

```
# Save the model into trainer/trainer.yml
recognizer.write('trainer/trainer.yml') # recognizer.save() worked on Mac, but not on Pi

# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained. Exiting Program".format(len(np.unique(ids))))
```

5. Face recognition

```
import cv2
import numpy as np
import os

recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainer/trainer.yml')
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath)

font = cv2.FONT_HERSHEY_SIMPLEX

# iniciate id counter
id = 0

# names related to ids: example ==> Marcelo: id=1,  etc
names = ['None', 'AhMed Mubarak', 'Paula', 'Ilza', 'Z', 'W']

# Initialize and start realtime video capture
cam = cv2.VideoCapture(1)
cam.set(3, 640)  # set video widht
cam.set(4, 480)  # set video height

# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)

while True:

    ret, img = cam.read()
   # img = cv2.flip(img, -1)  # Flip vertically

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.2,
        minNeighbors=5,
        minSize=(int(minW), int(minH)),
    )
```

```
    for(x, y, w, h) in faces:

        cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)

        id, confidence = recognizer.predict(gray[y:y+h, x:x+w])

        # Check if confidence is less them 100 ==> "0" is perfect match
        if (confidence < 100):
            id = names[id]
            confidence = "  {0}%".format(round(100 - confidence))
        else:
            id = "unknown"
            confidence = "  {0}%".format(round(100 - confidence))

        cv2.putText(img, str(id), (x+5, y-5), font, 1, (255, 255, 255), 2)
        cv2.putText(img, str(confidence), (x+5, y+h-5),
                    font, 1, (255, 255, 0), 1)

    cv2.imshow('camera', img)

    k = cv2.waitKey(10) & 0xff  # Press 'ESC' for exiting video
    if k == 27:
        break

# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
cam.release()
cv2.destroyAllWindows()
```

WITH MY BEST WISHES

ENG/AHMED MUBARAK 😉