**IMAGE AND FACE COGNITION**

**Hardware requirements:**

· Camera

· Computer

**Software requirements:**

· Python 2.7

· OpenCV

· OS-Linux/Windows/Mac

**Steps for image and face recognition using python:**

Install Python 2.7 and OpenCV from SourceForge

<https://www.python.org/downloads/>

<https://sourceforge.net/projects/opencvlibrary/>

**Install numpy,scipy,matplotlib, ipython,jupyter,pandas,sympy and nose libraries .**

**Code:**

//To load the image//

import numpy as np

import cv2

//load a color image in grayscale//

img=cv2.imread(‘messi5.pg’,0)

//load a color image//

img=cv2.imread(‘messi5.pg’,1)

//to display or save an image//

cv2.imshow(‘image’,img)

cv2.waitKey(0)

cv2.destroy all windows()

cv2.imwrite(‘messigray.png’,img)

//to load ,display and save a file//

import numpy as np

import cv2

img=cv2.imread(‘messi5.pg’,0)

cv2.imshow(‘image’,img)

k=cv2.waitKey(0)

if k==27:

cv2.destroy all windows()

elif k==ord(‘s’):

cv2.imwrite(‘messigray.png’,img)

cv2.destroy all windows()

// to capture the video//

cap = cv2.VideoCapture(0)

//to Read video from file//

cap = cv2.VideoCapture('vtest.avi')

while(True):

//to Capture frame-by-frame//

ret, frame = cap.read()

//to Display the resulting frame//

cv2.imshow('frame',gray)

//Use the below commands to release video capture//

cap.release()

cv2.destroyAllWindows()

//to draw a line or rectangle//

import numpy as np

import cv2

//Create a black image//

img = np.zeros((512,512,3), np.uint8)

// Draw a diagonal blue line with thickness of 5 px//

img = cv2.line(img,(0,0),(511,511),(255,0,0),5)

//Draw a rectangle//

img = cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)

//to draw a circle//

img = cv2.circle(img,(447,63), 63, (0,0,255), -1)

//dropping test//

font = cv2.FONT\_HERSHEY\_SIMPLEX

cv2.putText(img,'OpenCV',(10,500), font, 4,(255,255,255),2,cv2.LINE\_AA)

//to read image as a tuple//

print img.shape

print img.size

//make blue pixels as zero//

b = img[:,:,0]

//make red pixels as zero //

img[:,:,2] = 0

//to blend images//

img1 = cv2.imread('ml.png')

img2 = cv2.imread('opencv\_logo.jpg')

dst = cv2.addWeighted(img1,0.7,img2,0.3,0)

cv2.imshow('dst',dst)

cv2.waitKey(0)

cv2.destroyAllWindows()

//to detect corner//

/\*OpenCV has the function cv2.cornerHarris() for this purpose. Its arguments are :

img - Input image, it should be grayscale and float32 type.

blockSize - It is the size of neighbourhood considered for corner detection

ksize - Aperture parameter of Sobel derivative used.

k - Harris detector free parameter in the equation.\*/

import cv2

import numpy as np

filename = 'chessboard.jpg'

img = cv2.imread(filename)

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

gray = np.float32(gray)

dst = cv2.cornerHarris(gray,2,3,0.04)

//result is dilated for marking the corners, not important//

dst = cv2.dilate(dst,None)

//Threshold for an optimal value, it may vary depending on the image//

img[dst>0.01\*dst.max()]=[0,0,255]

cv2.imshow('dst',img)

if cv2.waitKey(0) & 0xff == 27:

cv2.destroyAllWindows()

//edge recognition using fast algorithm//

import numpy as np

import cv2

from matplotlib import pyplot as plt

img = cv2.imread('simple.jpg',0)

//Initiate FAST object with default values//

fast = cv2.FastFeatureDetector()

// find and draw the keypoints//

kp = fast.detect(img,None)

img2 = cv2.drawKeypoints(img, kp, color=(255,0,0))

cv2.imwrite('fast\_true.png',img2)

//\*face recognition\*//

import cv2

cap = cv2.VideoCapture(0)

cap.set(3, 640) #WIDTH

cap.set(4, 480) #HEIGHT

face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

while(True):

//Capture frame-by-frame//

ret, frame = cap.read()

// Our operations on the frame come here//

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

print(len(faces))

// Display the resulting frame//

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

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

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = frame[y:y+h, x:x+w]

cv2.imshow('frame',frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

// When everything done, release the capture//

cap.release()

cv2.destroyAllWindows()

//face and eye detection//

import cv2

cap = cv2.VideoCapture(0)

cap.set(3, 640) #WIDTH

cap.set(4, 480) #HEIGHT

face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

eye\_cascade = cv2.CascadeClassifier('haarcascade\_eye.xml')

while(True):

//Capture frame-by-frame//

ret, frame = cap.read()

// Our operations on the frame come here//

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

print(len(faces))

//Display the resulting frame//

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

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

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = frame[y:y+h, x:x+w]

eyes = eye\_cascade.detectMultiScale(roi\_gray)

for (ex,ey,ew,eh) in eyes:

cv2.rectangle(roi\_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('frame',frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

//When everything is done, release the capture//

cap.release()

cv2.destroyAllWindows()

**dataSetGenerator:**

This file essentially identifies the faces using the camera image,crops the image to just the face,turns it into a grayscale image and stores the image with a unique id,this unique id is used to identify the person uniquely. This script captures multiple images of the same person for better recognition later.

import cv2  
cam = cv2.VideoCapture(0)  
detector=cv2.CascadeClassifier('Classifiers/face.xml')  
i=0  
offset=50  
name=raw\_input('enter your id')  
while True:  
 ret, im =cam.read()  
 gray=cv2.cvtColor(im,cv2.COLOR\_BGR2GRAY)  
 faces=detector.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100), flags=cv2.CASCADE\_SCALE\_IMAGE)  
 for(x,y,w,h) in faces:  
 i=i+1  
 cv2.imwrite("dataSet/face-"+name +'.'+ str(i) + ".jpg", gray[y-offset:y+h+offset,x-offset:x+w+offset])  
 cv2.rectangle(im,(x-50,y-50),(x+w+50,y+h+50),(225,0,0),2)  
 cv2.imshow('im',im[y-offset:y+h+offset,x-offset:x+w+offset])  
 cv2.waitKey(100)  
 if i>40:  
 cam.release()  
 cv2.destroyAllWindows()  
 break

**trainer.py:**

This file takes all the previously captured images,analyses them,calculates the ratios and converts the lines and curves into mathematical equations. This data is then stored in an XML file in a well structured format.

import cv2,os  
import numpy as np  
from PIL import Image   
  
recognizer = cv2.face.LBPHFaceRecognizer\_create()  
cascadePath = "Classifiers/face.xml"  
faceCascade = cv2.CascadeClassifier(cascadePath);  
path = 'dataSet'  
  
def get\_images\_and\_labels(path):  
 image\_paths = [os.path.join(path, f) for f in os.listdir(path)]  
 # images will contains face images  
 images = []  
 # labels will contains the label that is assigned to the image  
 labels = []  
 for image\_path in image\_paths:  
 # Read the image and convert to grayscale  
 image\_pil = Image.open(image\_path).convert('L')  
 # Convert the image format into numpy array  
 image = np.array(image\_pil, 'uint8')  
 # Get the label of the image  
 nbr = int(os.path.split(image\_path)[1].split(".")[0].replace("face-", ""))  
 #nbr=int(''.join(str(ord(c)) for c in nbr))  
 print nbr  
 # Detect the face in the image  
 faces = faceCascade.detectMultiScale(image)  
 # If face is detected, append the face to images and the label to labels  
 for (x, y, w, h) in faces:  
 images.append(image[y: y + h, x: x + w])  
 labels.append(nbr)  
 cv2.imshow("Adding faces to traning set...", image[y: y + h, x: x + w])  
 cv2.waitKey(10)  
 # return the images list and labels list  
 return images, labels  
  
  
images, labels = get\_images\_and\_labels(path)  
cv2.imshow('test',images[0])  
cv2.waitKey(1)  
  
recognizer.train(images, np.array(labels))  
recognizer.save('trainer/trainer.yml')  
cv2.destroyAllWindows()

**detector.py:**

The detector file reads the previously stored XML file and compares the features mentioned in the XML file to the faces that it detects using the camera. This data is used to uniquely identify the user.

import cv2,os  
import numpy as np  
from PIL import Image   
import pickle  
  
recognizer = cv2.face.LBPHFaceRecognizer\_create()  
recognizer.read('trainer/trainer.yml')  
cascadePath = "Classifiers/face.xml"  
faceCascade = cv2.CascadeClassifier(cascadePath);  
path = 'dataSet'  
  
cam = cv2.VideoCapture(0)  
  
while True:  
 ret, im =cam.read()  
 gray=cv2.cvtColor(im,cv2.COLOR\_BGR2GRAY)  
 faces=faceCascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100), flags=cv2.CASCADE\_SCALE\_IMAGE)  
 for(x,y,w,h) in faces:  
 nbr\_predicted, conf = recognizer.predict(gray[y:y+h,x:x+w])  
 cv2.rectangle(im,(x-50,y-50),(x+w+50,y+h+50),(225,0,0),2)  
 if(nbr\_predicted==1):  
 nbr\_predicted='Sanath'  
 elif(nbr\_predicted==9):  
 nbr\_predicted='Yash'  
 elif(nbr\_predicted==5):  
 nbr\_predicted='Ganesh'   
 print(nbr\_predicted)  
 cv2.imshow('im',im)  
 cv2.waitKey(10)