

REAL TIME FACE DETECTION

Deepali Jadhav , Vaibhav Salve

Student, Department of Computer Engineering, Rajiv Gandhi College of Engineering, Ahmednagar^{1,2}

Guide Name :- Chaudhari N. J

Assistant Professor, Computer Engineering, Rajiv Gandhi College of Engineering, Ahmednagar³

Abstract- In order to recognize a face, we would first need to detect a face from an image. During the last few years, Local Binary Patterns (LBP) has aroused increasing interest in image processing and computer vision. LBP was originally proposed for texture analysis, and has proved a simple yet powerful approach to describe local structures. It has been extensively exploited in many applications, for instance, face image analysis, image and video retrieval, environment modelling, visual inspection, motion analysis, biomedical and aerial image analysis, remote sensing. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier.

Keywords- LBPH, Face Recognition

I INTRODUCTION

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. Over the last few decade lots of work is been done in face detection and recognition as it's a best way for person identification because it doesn't require human cooperation .So that it became a hot topic in biometrics. Since lots of methods are introduced for detection and recognition which considered as a milestone.

Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene. Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can use in

many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc.

This project uses LBPH (Local Binary Patterns Histograms) Algorithm to detect faces in real-time from the given input with the accurate results. All these approaches involves extracting the facial characteristics and then comparing them with the model to recognize whether it is a face or not.

II LITERATURE REVIEW

Sirovich and Kriby were able to show that feature analysis on a collection of facial images could form a set of basic features. In 1991, Turk and Pentland expanded upon the Eigenface approach by discovering how to detect faces within images. This led to the first instances of automatic face recognition.

Face detection is the process of automatically locating human faces in visual media (digital images or video). A face that is detected is reported at a position with an associated size and orientation. Once a face is detected, it can be searched for landmarks such as the eyes and nose. Here are some of the terms that we use in discussing face detection and the various functionalities of the Mobile Vision API.

Face recognition automatically determines if two faces are likely to correspond to the same person. Note that at this time, the Google Face API only provides functionality for face detection and **not** face recognition.

Face tracking extends face detection to video sequences. Any face appearing in a video for any length of time can be tracked. That is, faces that are detected in consecutive video frames can be identified as being the same person. Note that this is not a form of face recognition; this mechanism just makes inferences based

on the position and motion of the face(s) in a video sequence.

A **landmark** is a point of interest within a face. The left eye, right eye, and nose base are all examples of landmarks. The Face API provides the ability to find landmarks on a detected face.

Classification is determining whether a certain facial characteristic is present. For example, a face can be classified with regards to whether its eyes are open or closed. Another example is whether the face is smiling or not.

Face Orientation

The face API detects faces at a range of different angles, as illustrated below:

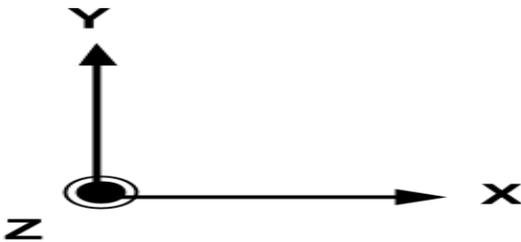


Figure 1: The coordinate system with the image in the XY plane and the Z axis coming out of the figure.

The **Euler X**, **Euler Y**, and **Euler Z** angles characterize a face’s orientation as shown in Fig. 1. The Face API provides measurement of Euler Y and Euler Z (but not Euler X) for detected faces. The Euler Z angle of the face is always reported. The Euler Y angle is available only when using the “accurate” mode setting of the face detector (as opposed to the “fast” mode setting, which takes some shortcuts to make detection faster). The Euler X angle is currently not supported.

Landmarks

A landmark is a point of interest within a face. The left eye, right eye, and nose base are all examples of landmarks. The figure below shows some examples of landmarks:



Figure 2: Landmarks of detection

Rather than first detecting landmarks and using the landmarks as a basis of detecting the whole face, the Face API detects the whole face independently of detailed landmark information. For this reason, landmark detection is an optional step that could be done after the face is detected. Landmark detection is not done by default, since it takes additional time to run. You can optionally specify that landmark detection should be done.

The following table summarizes all of the landmarks that can be detected, for an associated face Euler Y angle:

Euler Y-angle	Detectable Landmarks
< -36 degrees	left eye, left mouth, left ear, nose base, left cheek
-36 degrees to -12 degrees	left mouth, nose base, bottom mouth, right eye, left eye, left cheek, left ear tip
-12 degrees to 12 degrees	right eye, left eye, nose base, left cheek, right cheek, left mouth, right mouth, bottom mouth
12 degrees to 36 degrees	right mouth, nose base, bottom mouth, left eye, right eye, right cheek, right ear tip
> 36 degrees	right eye, right mouth, right ear, nose base, right cheek

Each detected landmark includes its associated position in the image.

Classification

Classification determines whether a certain facial characteristic is present. The Android Face API currently supports two classifications: **eyes open** and **smiling**. The iOS Face API currently supports the **smiling** classification. Classification is expressed as a certainty value, indicating the confidence that the facial characteristic is present. For example, a value of 0.7 or more for the smiling classification indicates that it is likely that a person is smiling.

Both of these classifications rely upon landmark detection.

Also note that “eyes open” and “smiling” classification only works for frontal faces, that is, faces with a small Euler Y angle (at most about +/- 18 degrees).

Most detection systems carry out the task by extracting certain properties (e.g., local features or holistic intensity patterns) of a set of training images acquired at a fixed pose (e.g., upright frontal pose) in an off-line setting. To reduce the effects of illumination change, these images are processed with histogram equalization [3, 1] or standardization (i.e., zero mean unit variance) [2]. Based on the extracted properties, these systems typically scan through the entire image at every possible location and scale in order to locate faces. The extracted properties can be either manually coded (with human knowledge) or learned from a set of data as adopted in the recent systems that have demonstrated impressive results [3, 1, 4, 5, 2]. In order to detect faces at different scale, the detection process is usually repeated to a pyramid of images whose resolution are reduced by a certain factor (e.g., 1.2) from the original one [3, 1]. Such procedures may be expedited when other visual cues can be accurately incorporated (e.g., color and motion) as pre-processing steps to reduce the search space [5]. As faces are often detected across scale, the raw detected faces are usually further processed to combine overlapped results and remove false positives with heuristics (e.g., faces typically do not overlap in images) [1] or further processing (e.g., edge detection and intensity variance). Numerous representations have

been proposed for face detection, including pixel-based [3, 1, 5], parts-based [6, 4, 7], local edge features [8, 9], Haar wavelets [10, 4] and Haar-like features [2, 11]. While earlier holistic representation schemes are able to detect faces [3, 1, 5], the recent systems with Haar-like features [2, 12, 13] have demonstrated impressive empirical results in detecting faces under occlusion. A large and representative training set of face images is essential for the success of learning-based face detectors. From the set of collected data, more positive examples can be synthetically generated by perturbing, mirroring, rotating and scaling the original face images [3, 1]. On the other hand, it is relatively easier to collect negative examples by randomly sampling images without face images [3, 1]. As face detection can be mainly formulated as a pattern recognition problem, numerous algorithms have been proposed to learn their generic templates (e.g., eigenface and statistical distribution) or discriminant classifiers (e.g., neural networks, Fisher linear discriminant, sparse network of Winnows, decision tree, Bayes classifiers, support vector machines, and AdaBoost). Typically, a good face detection system needs to be trained with several iterations. One common method to further improve the system is to bootstrap a trained face detector with test sets, and re-train the system with the false positive as well as negatives [1].

This process is repeated several times in order to further improve the performance of a face detector. A survey on these topics can be found in [5], and the most recent advances are discussed in the next section.

III SYSTEM ARCHITECTURE

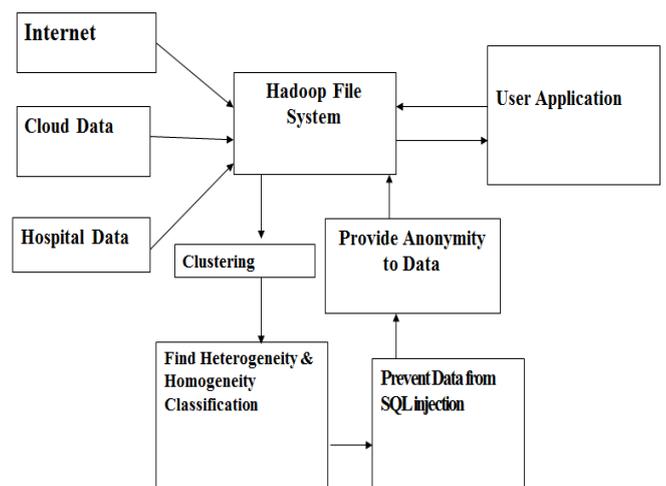


Figure 3: System Architecture

IV SOFTWARE DETAILS

OpenCV is a Library which is used to carry out image processing using programming languages like python. This project utilizes OpenCV Library to make a Real-Time Face Detection using your webcam as a primary camera.

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one. Its full details are given here: [Cascade Classifier Training](#).

Here we will deal with detection. OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. Those XML files are stored in `opencv/data/haarcascades/` folder. Let's create face and eye detector with OpenCV.

OpenCV was started at Intel in the year 1999 by **Gary Bradsky**. The first release came a little later in the year 2000. OpenCV essentially stands for **Open Source Computer Vision Library**. Although it is written in optimized C/C++, it has interfaces for Python and Java along with C++. OpenCV boasts of an active user base all over the world with its use increasing day by day due to the surge in computer vision applications.

OpenCV-Python is the python API for OpenCV. You can think of it as a python wrapper around the C++ implementation of OpenCV. OpenCV-Python is not only fast (since the background consists of code written in C/C++) but is also easy to code and deploy (due to the Python wrapper in foreground). This makes it a great choice to perform computationally intensive programs.

V APPROACH/ALGORITHMS USED:

1. This project uses LBPH (Local Binary Patterns Histograms) Algorithm to detect faces. It labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

2. LBPH uses 4 parameters :

- (i) Radius: the radius is used to build the circular local binary pattern and represents the radius around the central pixel.
- (ii) Neighbors : the number of sample points to build the circular local binary pattern.
- (iii) Grid X : the number of cells in the horizontal

direction.

(iv) Grid Y : the number of cells in the vertical direction.

3. The model built is trained with the faces with tag given to them, and later on, the machine is given a test data and machine decides the correct label for it.

How to use :

1. Create a directory in your pc and name it (say project)
2. Create two python files named `create_data.py` and `face_recognize.py`, copy the first source code and second source code in it respectively.
3. Copy `haarcascade_frontalface_default.xml` to the project directory, you can get it in `opencv`.
4. You are ready to now run the following codes.

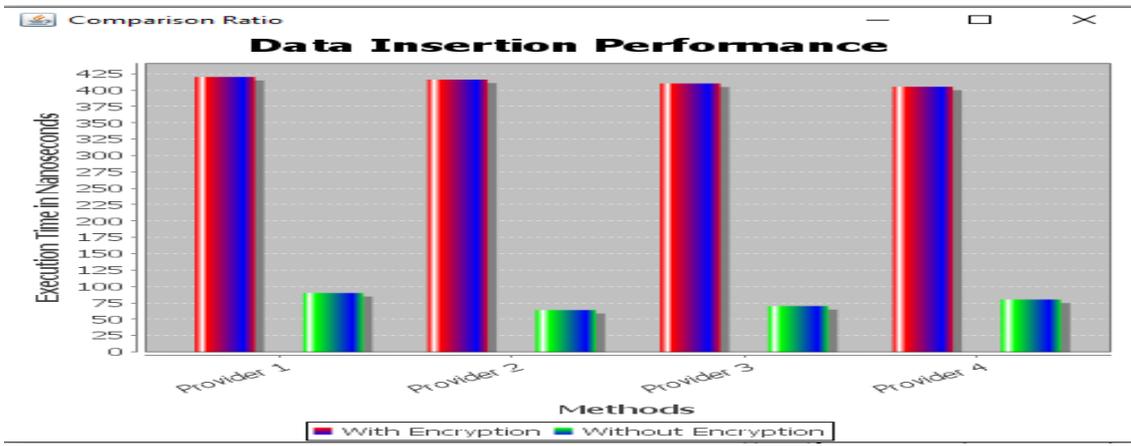
VI RESULT ANALYSIS & CODE

Code

```
import cv2
face_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.
xml')
eye_cascade =
cv2.CascadeClassifier('haarcascade_eye.xml')
cap = cv2.VideoCapture(0)
while 1:
    ret, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x,y,w,h) in faces:
        cv2.rectangle(img,(x,y),(x+w,y+h),(255,255,0),2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = img[y:y+h, x:x+w]
    cv2.imshow('img',img)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break
    cap.release()
cv2.destroyAllWindows()
```



Figure 4: Face Detection System result



NAME	AGE	GENDER	ZIP	DESEASE A...	PROVIDER
****	29	male	*****	achondropla...	p1
****	79	female	*****	cough_rest	p3
****	89	female	*****	cough_rest	p4
****	74	male	*****	cancer_medi...	p1
****	70	female	*****	abuse_excer...	p1
****	89	female	*****	fever_take a ...	p4
****	99	female	*****	tumor_no ne...	p3

NAME	AGE	GENDER	ZIP	DESEASE ...	PROVIDER
****	46	male	*****	dementia_b...	p1
****	25	male	*****	cough_rest	p1
****	29	female	*****	tumor_no n...	p1
****	26	female	*****	abuse_exce...	p3
****	73	female	*****	tumor_no n...	p3
****	23	female	*****	fever_take a...	p3

NAME	AGE	GENDER	ZIP	DESEASE A...	PROVIDER
****	29	female	*****	abuse_excer...	p1
****	25	male	*****	tumor_no ne...	p1
****	29	male	*****	cancer_medi...	p1
****	41	female	*****	backpain_tak...	p3
****	65	female	*****	achondropla...	p4
****	78	female	*****	cancer_medi...	p4

NAME	AGE	GENDER	ZIP	DESEASE ...	PROVIDER
****	90	male	*****	abuse_exce...	p1
****	94	female	*****	dementia_b...	p1
****	25	female	*****	cough_rest	p4
****	51	female	*****	achondropl...	p4
****	31	male	*****	fever_take a...	p1
****	43	female	*****	dementia_b...	p1

NAME	AGE	GENDER	ZIP	DESEASE A...	PROVIDER
****	44	female	*****	cough_rest	p4
****	49	male	*****	fever_take a ...	p1
****	35	female	*****	abuse_excer...	p4
****	25	male	*****	backpain_tak...	p3
****	47	female	*****	abuse_excer...	p3
****	84	male	*****	cancer_medi...	p4

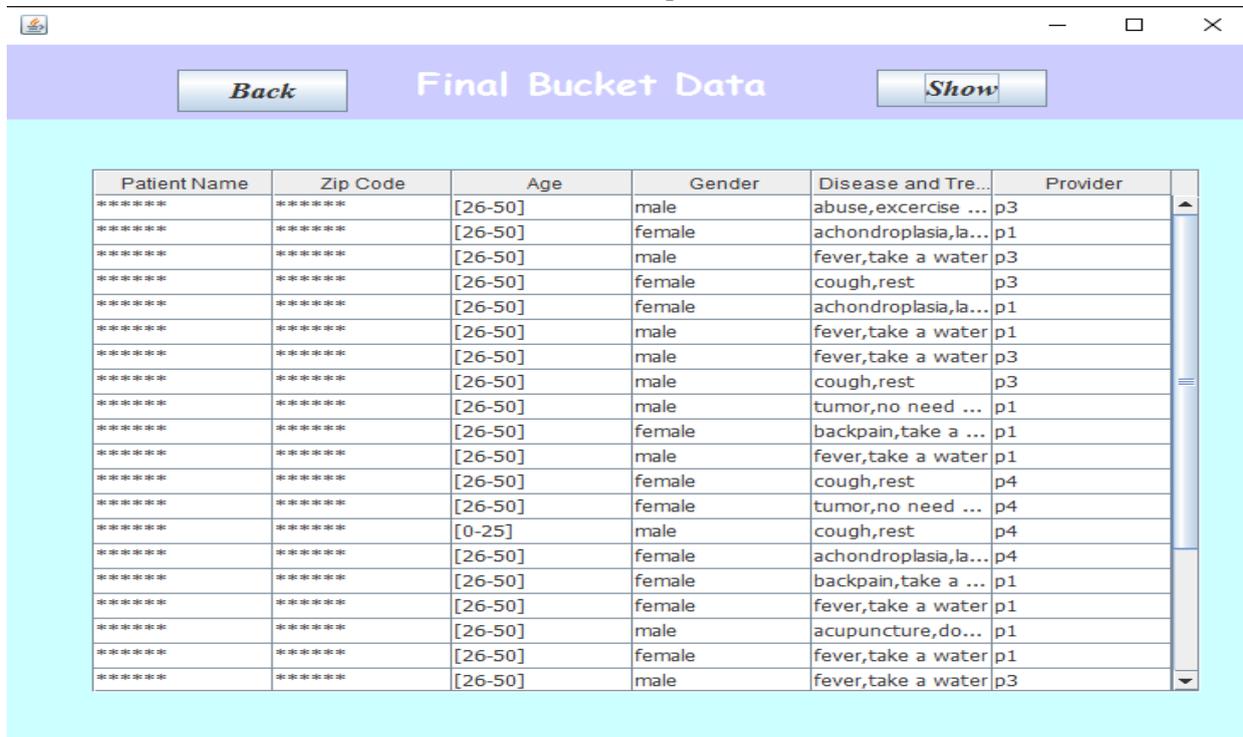
NAME	AGE	GENDER	ZIP	DESEASE ...	PROVIDER
****	60	male	*****	cough_rest	p1
****	71	male	*****	cancer_me...	p4
****	77	male	*****	cancer_me...	p1
****	46	female	*****	dementia_b...	p3
****	74	male	*****	backpain.ta...	p3
****	99	female	*****	achondropl...	p3

Final

Exit

Home

Final Output



Patient Name	Zip Code	Age	Gender	Disease and Tre...	Provider
*****	*****	[26-50]	male	abuse,exercise ...	p3
*****	*****	[26-50]	female	achondroplasia,la...	p1
*****	*****	[26-50]	male	fever,take a water	p3
*****	*****	[26-50]	female	cough,rest	p3
*****	*****	[26-50]	female	achondroplasia,la...	p1
*****	*****	[26-50]	male	fever,take a water	p1
*****	*****	[26-50]	male	fever,take a water	p3
*****	*****	[26-50]	male	cough,rest	p3
*****	*****	[26-50]	male	tumor,no need ...	p1
*****	*****	[26-50]	female	backpain,take a ...	p1
*****	*****	[26-50]	male	fever,take a water	p1
*****	*****	[26-50]	female	cough,rest	p4
*****	*****	[26-50]	female	tumor,no need ...	p4
*****	*****	[0-25]	male	cough,rest	p4
*****	*****	[26-50]	female	achondroplasia,la...	p4
*****	*****	[26-50]	female	backpain,take a ...	p1
*****	*****	[26-50]	female	fever,take a water	p1
*****	*****	[26-50]	male	acupuncture,do...	p1
*****	*****	[26-50]	female	fever,take a water	p1
*****	*****	[26-50]	male	fever,take a water	p3

VII Application

1. Payments

It doesn't take a genius to work out why businesses want payments to be easy. Online shopping and contactless cards are just two examples that demonstrate the seamlessness of postmodern purchases. With FaceTech, however, customers wouldn't even need their cards. In 2016, MasterCard launched a new selfie pay app called MasterCard Identity Check. Customers open the app to confirm a payment using their camera, and that's that. Facial recognition is already used in store and at ATMs, but the next step is to do the same for online payments. Chinese ecommerce firm Alibaba and affiliate payment software Alipay are planning to apply the software to purchases made over the Internet.

2. Access and security

As well as verifying a payment, facial biometrics can be integrated with physical devices and objects. Instead of using passcodes, mobile phones and other consumer electronics will be accessed via owners' facial features. Apple, Samsung and Xiaomi Corp. have all installed FaceTech in their phones. This is only a small scale example, though. In future, it looks like consumers will

be able to get into their cars, houses, and other secure physical locations simply by looking at them. Jaguar is already working on walking gait ID – a potential parallel to facial recognition technology. Other corporations are likely to take advantage of this, too. Innovative facial security could be especially useful for a company or organisation that handles sensitive data and needs to keep tight controls on who enters their facilities.

3. Criminal identification

If FaceTech can be used to keep unauthorised people out of facilities, surely it can be used to help put them firmly inside them. This is exactly what the US Federal Bureau of Investigation is attempting to do by using a machine learning algorithm to identify suspects from their driver's licences. The FBI currently have a database which includes half of the national population's faces. This is as useful as it is creepy, giving law enforcers another way of tracking criminals across the country. AI equipped cameras have also been trialled in the UK to identify those smuggling contraband into prisons.

4. Advertising

The ability to collect and collate masses of personal data has given marketers and advertisers the chance to get

closer than ever to their target markets. FaceTech could do much the same, by allowing companies to recognise certain demographics – for instance, if the customer is a male between the ages of 12 and 21, the screen might show an ad for the latest FIFA game. Grocery giant Tesco plans to install OptimEyes screens at 450 petrol stations in the UK to deliver targeted ads to customers. According to company CEO Simon Sugar, the cameras could change the face of British retail. Perhaps he's right – but only if the cameras can correctly identify customers. Being classified as the wrong age or gender is far less amusing than having your name spelt wrong on a Starbucks cup.

5. Healthcare

Instead of recognising an individual via FaceTech, medical professionals could identify illnesses by looking at a patient's features. This would alleviate the ongoing strain on medical centres by slashing waiting lists and streamlining the appointment process. The question is, would you really want to find out you had a serious illness from a screen? If it's a choice between a virtual consultation or a month long wait for an appointment, then maybe so. Another application of facial biometrics within healthcare is to secure patient data by using a unique patient photo instead of passwords and usernames.

VIII CONCLUSION

- This system is designed for Mining data in Health Care Application.
- It is based on HACE theorem.
- Our system will offer a good platform to extract information with the help of Hadoop File System.
- Our system will generate anonymous data, and prevent data from SQL injection & provide anonymity to data.

REFERENCE

- [1] Adelson, E. H., and Bertgen, J. R. (1986) The Extraction of Spatio-Temporal Energy in Human and Machine Vision, Proceedings of Workshop on Motion: Representation and •

- Analysis (pp. 151-155) Charleston, SC; May 7-9
- [2] AAFPRS(1997). A newsletter from the American Academy of Facial Plastic and Reconstructive Surgery. Third Quarter 1997, Vol. 11, No. 3. Page 3.
- [3] Baron, R. J. (1981). Mechanisms of human facial recognition. *International Journal of Man Machine Studies*, 15:137-178 • Beymer, D. and Poggio, T. (1995) Face Recognition From One Example View, A.I. Memo No. 1536, C.B.C.L. Paper No. 121. MIT
- [4] Bichsel, M. (1991). Strategies of Robust Objects Recognition for Automatic Identification of Human Faces. PhD thesis, Eidgenossischen Technischen Hochschule, Zurich. Brennan, S. E. (1982) The caricature generator. M.S. Thesis. MIT.
- [5] Brunelli, R. and Poggio, T. (1993), Face Recognition: Features versus Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042-1052
- [6] Craw, I., Ellis, H., and Lishman, J.R. (1987). Automatic extraction of face features. *Pattern Recognition Letters*, 5:183-187, February.
- [7] Deffenbacher K.A., Johanson J., and O'Toole A.J. (1998) Facial ageing, attractiveness, and distinctiveness. *Perception*. 27(10):1233-1243