IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.301

Volume 3, Issue 6, April 2023

Android App for Face Detection

Kamble Narendra Sanjay[1], Guldagad Rutuja Babasaheb[2], Umap Pallavi Satish[3], Gambhire Shivani Mahadev[4], Avinash Balasaheb Anap[5]

Students, Department of Computer Technology[1,2,3,4]

Guide, Department of Computer Technology[5]

Pravara Polytechnic, Loni, Maharashtra, India

Abstract: *This work describes the development of a face detection and recognition application developed into Raspberry Pi and Android. The application connects with the Raspberry Pi by Bluetooth protocols. The object detection is based on boosted cascade while the face recognition is based on Eigenfaces. The developed system may be especially useful for visually impaired users since it can contribute to facilitate their autonomous behavior during their everyday life. The developed device shows great potential for extrapolation to other areas as education of visually impaired users*.

Keywords: Object Detection; Face Recognition; Boodting Cascade; Simple Features; Eigenfaces

I. INTRODUCTION

Human survival depends on the main five perception senses of the human sensorial mechanism defined by Aristotle as smell, hearing, touch, vision and taste [2]. Loss of one of the main senses does not mean the death of the person, but it can significantly complicate its daily activity. It also can create discomfort and insecurity as well as requirement of company, especially in the case of vision and hearing deficiencies [1]. The inability to recognize or identify people during the stay at home or during meetings becomes a frustration (inconvenient) for blind people; it is especially unpleasant for visually impaired people when the near people are quite and do not make noises since they may not note their presence. New technologies based on speech, object and face recognition have become complementary system for disabled people. Usually, they convert human environment into speech or tactile information. Blind people or people with low vision may perceive persons from the environment, familiars, friends or colleagues at work by face detection and recognition systems. Real-time object detection [3] face recognition [4], text recognition [5] and currency bills identification [6, 7] are some of the large amount of developed applications. Ref. [3] describes the object detection and uses cascade structure and AdaBoost classifiers based on Haar basis functions; [4] uses Eigenfaces based on the Turk and Pentland models; [8, 9], [10] implement the Principle Component Analysis or Eigenfaces for face recognition in order to perceive facial expressions, emotions and gesture; [7] employs PCA for dollar recognition and [6] uses PCA-SHIFT in order to ach improvements over the object recognition approach. The present work is focused on developing face detec and face recognition algorithms to be used by visually impa people. Face detection uses the algorithms proposed by V and Jones [4]. Also the Haar Cascades functions and principal Component Analysis of the Eigenfacesalgorit were used in order to achieve the pursued objective. system is implemented on Raspberry Pi hardware and interface for Android smartphones. Raspberry Pi cams Pi N have a free open code and are able to run under Open libraries and C++ bindings for Python. The Raspberry Pi N cameras have the 5 megapixel and Bluetooth connection. The paper is structured as follows: Section II provide overall description of the developed system; Section explains the face detection algorithm and its foundati Section IV presents the conclusions of the work.**Advantages**

II. FACE DETECTION

Databases

In order to train the face detector application, the Yale Face Database [11] with the size 6.4MB, that contains 165 grayscale images in GIF format of 15 individuals was used. The database contains 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, and sleepy, surprised, and wink. The collection includes 60 non-facial standard test negative images for the system, because they do not contain faces (such as cameraman, coins...).

Face Detection

The Boosted Cascade technique was used for face detection. The technique is based on fast feature evaluation by implementing Haar functions, the discrete AdaBoost algorithms. The aim of these algorithms is to select a small number of features calculating the haar-likes features. The effective method for selecting a small number of correct features that does not have a significant variation is AdaBoost algorithm. Due to that, the learning algorithm selects the features that best classify the positive and negative samples Fig. 1.

II. MATERIALS AND METHODS

TABLE I :SCALE FACTOR VERSUS ACCURACY

minNeighbors	0	1	2	3	4	5	6	7	8	9	10
Hit rate	0.05	0.75	0.90	0.97	0.982	0.984	0.982	0.98	0.97	0.96	0.95

The classifier was based on the composition of several weak classifiers through boosting. Each one of the classifiers is discriminated in function of haar-like features.

The haar features are like Kernel convolution. Each feature represents a single value obtained from the extraction of the sum of pixels of white rectangle from sum of pixels under black rectangle. database, the algorithm was tested and the 10 relevant characteristics for classification were detected.

f s = i=1 αihi(s)   (1)

Where   represent the "weak" or basis classifier and H(x)=sing(f(x)) represent the "strong" or final classifier.

g *fj* s ; p, 8 = 0 ojtherwise (2)

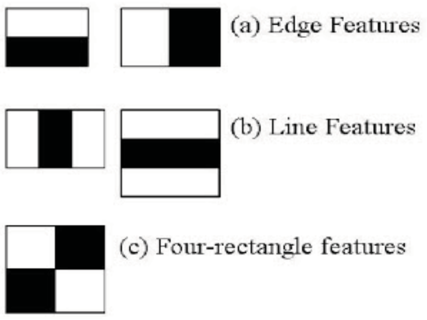


Fig.1. Haar features

In order to test the face detection algorithm, the Yale Face Database with the 165 negative images, gray scale and GIF format of 15 individuals was used. The pre-trained Haar cascade classifier created by Rainer Lienhart [11] was used.

Where fj(x) is a set of future responses extracted by applying the feature fj to each training image xi and associated labels {y1,…yn}; where yiϵ{0,1} and a set of non-negative θ=(p,θ) and is the threshold value, p ϵ{-1,1} and a parity value, ϵ is the value of the error associated with this classifier applied to the training data.

After that, the weighted mean of the positive examples and negative examples are computed.

The database contains 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, and sleepy, surprised, and wink. The collection of images used for the experiments will be completed with 400 non-facial images from the Background image dataset [12]. These images are considered as negative images for the system, because they do not contain faces.

μp = ni—1niα—1ifαji(syii)yi , μN = (3)

A combination of The Yale Face Database and the 400 negative images were used for testing the face detection algorithm. For the positive images, we used the negative images and we have passed on them faces pertaining to the

III. RESULTS AND DISCUSSION

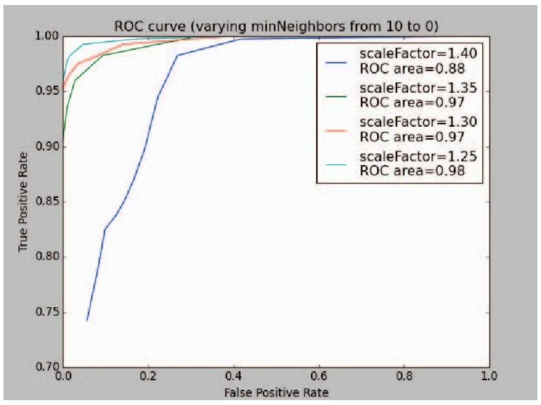


Fig.2. ROC curve

As observed, the best result is obtained with scaleFactor=1.25. Hence, we will consider now the hit rate for each value of minNeighbors with the selected scaleFactor=1.25 (see Fig.2).

Haar Feature-based Cascade Classifier was implemented into OpenCV computer vision library written in C and C++. The library is a free open-source for academic and commercial use.

OpenCV function also detects the objects, in our case, the face of different sizes in the input image and returns them as a list of rectangles by using the detectMultiScale function. And receives the parameter minNeighbors, which is the minimum number (n-1) of neighbor rectangles that makes up an object.

The Haar cascade is designed for a window of 24x24 that means that an image of this dimension has 160000 features. The considered images are bigger than 24x24; due to this fact, we need to resize these images. Also, the detector needs to be scanned across an image at multiple scales in addition to multiple locations creating small windows. The parameter scaleFactor determines the factor used to rescale, e.g 1.1 means increasing window by 10%.

Accuracy results were obtained for different scale factors (see Table 1). In our case the best result is 0.984, with minNeighbors value set on 5.

The values of its confusion matrix are the following:

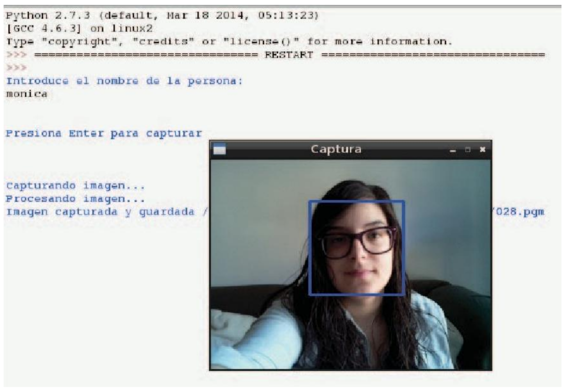
Total: 800

True Positives: 391 True Negatives:399 False Positives:4 False Negatives:9

Once the algorithm was trained, the new faces were introduced through capture_faces.py Script. A small application was created with detectMultiScale method.

The application is capable to take a photo through RPI camera to process the image, to detect the face, to resize the image with the detected face, to convert the images in grey scale, to resize the image for 92x112 pixels in order to coincide with the images of the Yale Faces database and to store the images into the file with names.

For each subject, more than 10 images were taken (images at different angles, with different light condition and different expressions, see Fig.3 and Fig.4).



```
Python 2.7.3 (default, Mar 18 2014, 05:13:23)
[GCC 4.6.3] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Introduce el nombre de la persona:
monica


Presiona Enter para capturar



Capturando imagen...
Procesando imagen...
Imagen capturada y guardada /                                      028.pgm
```

Fig.3. Example of a new face database.

Face Recognition

Once the face detector parameters set had been obtained, it was necessary to evaluate the developed face recognition system, implemented using the FaceRecognizer class of OpenCV library [12].

For this experiment, 81.81% of the images of the database were used (9 images of each subject) for training the eigenvector model, while the remaining 18.18% (2 images of each subject) were employed for testing the results. For the testing, the images were selected randomly so they do not influence in the results. The experiment was performed 3 times, changing the training and testing images each time.

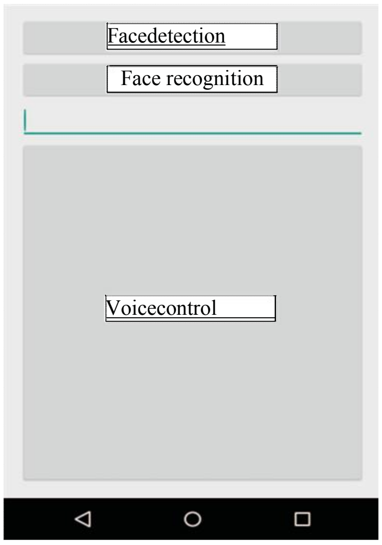
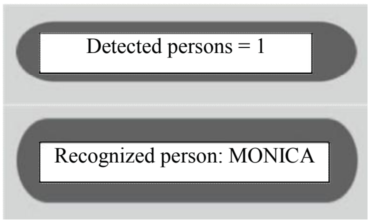


Figure 4: Output Image

We have to mention that the training data preparation step requires two types of samples: negative and positive. The negative samples correspond to non-face images and the positive images correspond to images with the corresponding face. The negative samples are taken from arbitrary images that do not correspond to the person incase.



Detected persons = 1

Recognized person: MONICA

Fig.5. Android interface. Results of the application Detected persons 1, The recognized person is MONICA

The positive samples are the ten images of the subject face. They may be taken with the camera or processed by applying blur and insignificant distortions. After training the model, several tests by varying the number of components (eigenfaces) kept for the PCA were conducted. As it can be seen, the system reaches its maximum hit rate with 150 eigenfaces and then stabilizes. Therefore, the number of components is set to 150, and out final system performs with an 84.4% of success.

Once all patterns were created and stored, the model for face recognition was built by using createEigenFace Recognizer. Through the train.py script, the file with all the images was read. As soon as the training part took off, the file was stored into training.xml.

With the rec_faces.py script which contains the initialization, detection and recognizer methods, the face recognition was carried out. Through RFCOMM protocol, the Raspberry Pi is connected to the Android application for face recognition through voice control platform.

IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.301

Volume 3, Issue 6, April 2023

IV. SYSTEM DESCRIPTION

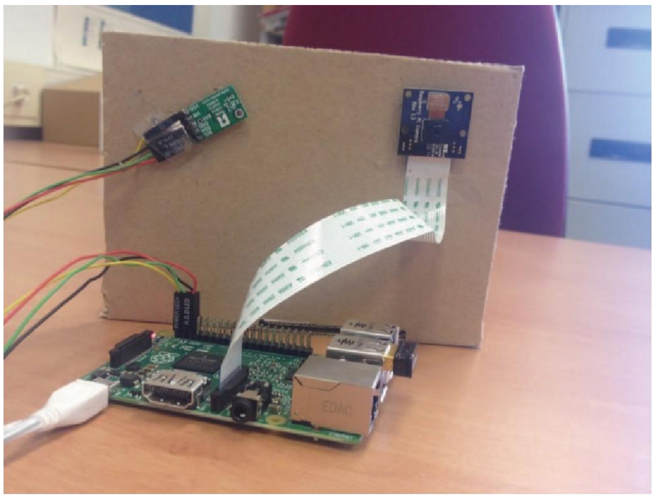


Fig.6. System equipments

A model B Raspberry Pi computer module unit of a credit-card size and raspberry Pi camera compose the system. The Model B Raspberry Pi has 3,5W, an ARM1176JZF-S 700MHz processor, 512 Megabytes of RAM, VideoCore IV GPU and multiple Input/Output interfaces. It supports the whole artificial vision process.
The Raspberry Pi computer runs under Python. It has access to OpenCV libraries. The Raspberry Pi is connected through Bluetooth with the mobile phone running under Android Fig 5 and Fig. 6.

V. CONCLUSION

The paper deals with the development of a face detection and recognition system for blind people in real time by using Raspberry Pi and Android app. Two main aspects on machine learning compose the work: the object detection based on boosted cascade and the face recognition based on Eigenfaces. The system is developed on Raspberry hardware. From the results, we can conclude that the average hit rate of the recognition system is of 84,4%. The developed system is especially helpful for visually impaired users, since it can help them to recognize near people. It can be also utilized in educational applications of people with problems in their vision.

REFERENCES

[1]. Neil Smyth, “Android Studio Development Essentials”, Android 6 edition.
[2]. Ryan Hodson , “Android Programming Succinctly”, edition 3.
[3]. Wei-MengLee , “Beginning Android 4 Application Development”, Willey India Pvt., Ltd., 2012.
[4]. J.F. DiMarzio, “Android :A Programmers Guide”, First Edition, Tata Mcgraw-hill,2010.
[5]. Jason Morris, “Android User Interface Development”, Packet publication, 2011., “SQLite” Sams.
[6]. Jay A Kreibich, “Using SQLite”, O„Reily Media, 2010.
[7].http://www.android.com/
[8].http://developer.android.com/guide/topics/ui/
[9].http://www.life360.com/
[10].http://tehula.com/
[11].http://www.google.com/mobile/latitude/
[12].https://sqlite.org
[13].http://www.mysql.com/about/
[14].http://www.androiddeveloper.com/
[15].http://www.tutorialspoint.com/