Software Development Framework for Real-Time Face Detection and Recognition in Mobile Devices

https://doi.org/10.3991/ijim.v14i04.12077

Laxmisha Rai (✉), Zhiyuan Wang,
Amila Rodrigo, Zhaopeng Deng, Haiqing Liu
Shandong University of Science and Technology, Qingdao, China
laxmisha@ieee.org

Abstract—With the rapid use of Android OS in mobile devices and related products, face recognition technology is an essential feature, so that mobile devices have a strong personal identity authentication. In this paper, we propose Android based software development framework for real-time face detection and recognition using OpenCV library, which is applicable in several mobile applications. Initially, the Gaussian smoothing and gray-scale transformation algorithm is applied to preprocess the source image. Then, the Haar-like feature matching method is used to describe the characteristics of the operator and obtain the face characteristic value. Finally, the normalization method is used to match the recognition of face database. To achieve the face recognition in the Android platform, JNI (Java Native Interface) is used to call the local Open CV. The proposed system is tested in real-time in two different brands of smart phones, and results shows average success rate in both devices for face detection and recognition is 95% and 80% respectively.

Keywords—Authentication, Image processing, Wearable, Framework, JNI, OpenCV, Personal identity, Smart phones.

1 Introduction

In the recent years, devices running on Android platform, and smart phones are becoming increasingly ubiquitous. However, as most of these devices carry personal, and sensitive information, they demand biometric authentication [1]. The application of biometric authentication systems is countless. In these kinds of biometrics, physiological or behavioral characteristic are analyzed and their applications can be extended to several applications starting from industry to education, medicine, marketing, immigration centers, playgrounds, theatres, government offices, transportation systems, airports, and border security control systems, etc [2]. The common biometric authentication methods used for confirming user identity are face and fingerprints [3]. Considering face detection and recognition, applications can be extended to services such as better customer satisfaction verification, where the advertisers can make decisions more accurately and favorably based on the features derived from actual human faces, rather than depending on anonymous searches, and unverified data collected

over the Internet. Face recognition technology is an emerging biometric technology, mainly used in intelligent robots, smart homes, and military security systems etc. The major challenges of face recognition system are the identification problem; where the human face has dynamic biological features, such as the similarities of face structures and the face variation caused by different observation angles [4]. In addition, the factors such as the masking and the degree of user cooperation while authentication also makes it difficult during face recognition. As a kind of human intrinsic attribute, face features have strong individual differences and easy to collect. The facial features are the basis of identity and authentication, which is safer and more reliable than traditional methods. Using face recognition and authentication process is useful is several applications related multimedia processing, where camera, and mobile devices are widely used. This can save valuable time and resources with better user experience especially during cumbersome registration process, and logging into mobile terminals remotely.

In several cases, a simple variation in the facial condition affect the accuracy of the systems through which the facial recognition is performed especially while using highly populated databases. This is why the facial recognition systems are not widely used in security systems as compared to other biometric systems such as fingerprint or iris recognition systems. The face recognition technology is based on physiological characteristics of identification method, where through the computer to extract facial features, and according to characteristics obtained authentication is determined. In the past several years, researchers have developed large applications for real-time object detection and face recognition, text recognition and currency bill identification [5,6]. In the earlier years, several researchers focused on popular methods available for of object and face detection. Chen, and Yuille [7] described the object detection and used cascade structure and AdaBoost classifiers based on Haar basis functions; Viola and Jones [8] uses Eigenfaces based on the Turk and Pentland models [9]. Other researchers implemented Principle Component Analysis (PCA) or Eigenfaces for face recognition in order to perceive facial expressions, emotions and gesture. In addition, researchers also focused on developing face detection and face recognition algorithms to be used by visually impaired people in the recent years [10,11]. Most of the face detection methods uses the algorithms proposed by Viola and Jones [8]. The Haar Cascades functions and the Principal Component Analysis of the Eigenfaces algorithms were used in order to achieve the detection and recognition objectives.

Recently, the works related to real-time face detection, and their role in mobile applications is getting widely popular among programmers, developers, and researchers. Authors in [12] proposed client-server-based framework, where face detection, and tracking application is designed for Android mobile devices. In [13] an algorithm designed to identify facial features on an android mobile platform. This algorithm is based on anthropometric face model and box-blur filtering. Similarly, other researchers also proposed the methods of emotion recognition in Android smart phones based on heart rates and the talk users obtained using built in camera, and microphones respectively [14]. An emotion recognition framework to analyze the facial expressions is presented in [15]. The main focus of this work is to identify emotions in complex environments such variation in lighting, and device movements. A related appli-

cation to recognize and analyze the user's audio and characteristics on smart phones is presented in [16]. This application is developed on Android platform for the primary goal of emotion recognition efficiently in real-time. Considering the development environment, since early 2010, several researchers developed applications on Android platform. An experience report of development environment for Android applications along with Eclipse IDE and open source tools can be found in [17]. A development of a face detection and recognition application developed into Raspberry Pi and Android is described in [18]. However, most of the earlier works related to Android, face recognition, and emotion recognition etc., are fail to generalize the approach towards developing Android based software framework for face recognition, which can be applied to several applications and devices for the privacy protection, user security, user authentication, and fraud detection. Considering these developments, in this paper, we explore the tools and methods necessary for implementing generalized Android based software development framework for real-time face detection and recognition in mobile applications. Initially, the face detection based on Adaboost face and the Haar feature is performed. Then the eigenface extraction algorithm of OpenCV is used to extract the features, and the extracted eigenface is compared with the saved eigenface. If the similarity exceeds the threshold, the face is identified as belong to the same person.

The paper is organized as follows. The Section 2 introduces the proposed system framework and detailed design. The image preprocessing, Gaussian smoothing, gray transformation, and binarization steps are described in this Section, along with Haar-like features and point graph. The details of specific face recognition are also presented here. The Section 3 presents the development environment with details of implementation, where details of building of Android development environment is presented, along with details OpenCV, JNI, and NDK. Section 4 presents the results obtained after face detection and recognition. Finally, the conclusion is presented in Section 5.

2 System Framework and Detailed Design

The face recognition system proposed in this paper include steps as shown in Figure 1 and Figure 2. Firstly, the image is captured by a camera, which either is capable of capturing videos or photographs. However, the camera which captures photographs is more suitable for accurate recognition. Secondly, as the face detection is a complex process in general, the system will try to standardize the image captured with the similar characteristics with the previously stored images in the gallery. This is required because most of the captured images have some random background or other images of other faces. Thirdly, feature extraction and mathematical representation named biometric reference is achieved. This step is the essential step which form the basis for face recognition. Final step is about process of comparison of models, where biometric reference is compared with the other models of familiar faces in the gallery. Declaring identity is establishing the close connection and affinity between two references, which is often carried out by the human factor. The flowchart of the overview of the proposed system is provided in Figure 1.

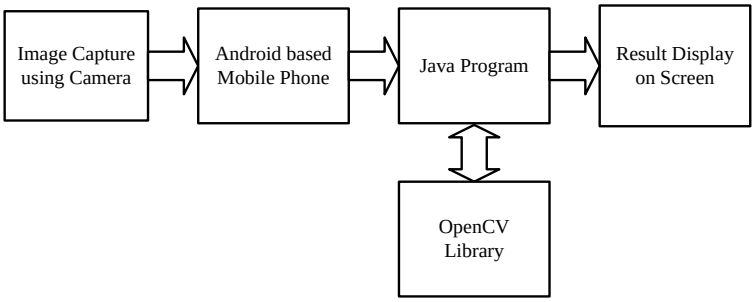


Fig. 1. Flowchart describing overview of system framework

The Figure 2 shows the flowchart for steps involving in face recognition and detection. The process of image detection and recognition includes several steps starting from capturing a dynamic image and making it as a static image using system camera. Afterwards, the system locates the face position to the obtained image based on the contour symmetry detection method. The image containing the effective face is filtered out, and several steps follows after this step. These steps include processing and adopting Haar-like feature matching method to extract the facial feature information [19], comparing the extracted feature data with the face database information, and then using the normalized square difference matching method in the OpenCV library to perform specific face recognition as shown in Figure 2.

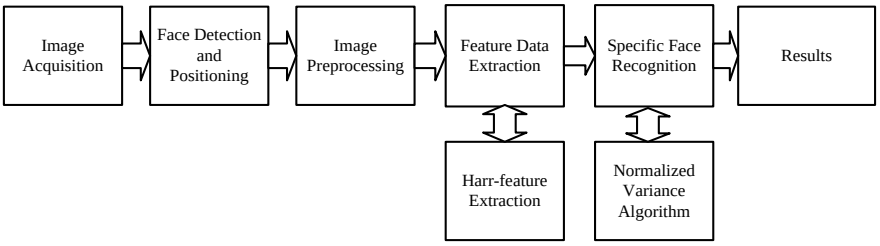


Fig. 2. Flowchart describing overview of system framework

2.1 Image preprocessing

In order to reduce the image noise, which may hinder the image extraction, detection, and recognition in the later stages, the image preprocessing method is selected as a combination of different algorithms to process the image step by step (Figure 3). The image preprocessing involves four steps, namely Gaussian smoothing, gray-scale transformation, contrast enhancement, and binarization. Using the Gaussian smoothing filtering and weighted average methods, the image is de-noised along with gray-scale transformation after the conversion by local mean and standard deviation algorithm to realize the contrast enhancement. Finally, the local adaptive binary method is used for binary processing after the processing of the grayscale image.

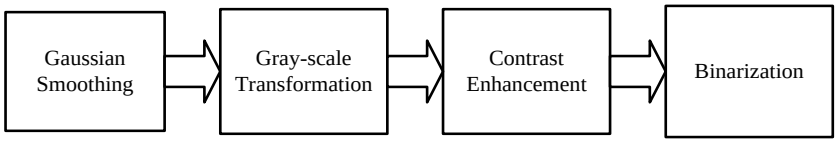


Fig. 3. Steps involved in image preprocessing

The reasons behind choosing Gaussian smoothing technique can be described as follows. Because of the interference of various external factors such as irregularity and noise which is highly inevitable during the process of acquiring video or image. This also leads to lose the image information and data corruption while preprocessing, and will affect the image quality during the subsequent steps such as image extraction, detection, and identification. Therefore, image noise filtering is highly essential. Compared with the smoothing method such as median filtering and adaptive filtering, the Gaussian smoothing filter is used to eliminate the noise in the spatial and frequency domain. The Gaussian smoothing image not only enhances the significant low frequency information, but also retains the image edge contour. The contrast enhancement method applied is as follows. According to the different brightness point level measurements and different levels of pixel statistics, the pixel information is compared at each point using clustering method. The method compares the differences between bright and dark points and improves the difference between the pixels, the contrast is enhanced. By using the selection of local mean and standard deviation algorithms to complete contrast enhancement, the excessive contrast in other high-frequency parts is avoided. Haar feature recognition algorithm, SIFT and SURF are based on grayscale, but generalized Hough transform is more suitable for the detection of the whole face. The Haar more inclined to face detection, SIFT and SURF are more complicated than Haar, thus the Haar feature algorithm is used for face recognition in this paper.

Gaussian smoothing: Image acquisition is easily distorted by various environmental factors resulting from irregular noise. So, this needs to be processed by a Gaussian smoothing filter. The smoothing process reduces the image noise using a bilateral filter, while keeping the edges sharp. The formula for Gaussian smoothing process for two-dimensional Gaussian function is shown as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2} \mathrm{e}^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{1}$$

Where x and y are defined as the pixel template coordinates, σ is defined as the smoothness parameter. The larger σ is, the better the smoothness is. The resulting kernel of the filter also exhibits Gaussian distribution characteristics, and the processed image enhances the low-frequency information.

Gray-scale transformation: The principle of Grayscale conversion method is that, the color image is converted to grayscale images for easier processing of images and face detection. The principle is to convert the R (red), G (green), and B (blue) values

in the picture to gray values. The system conversion formula uses the weighted average method as shown in below:

$$\text{Gray}=0.229*R + 0.587*G + 0.114*B \tag{2}$$

Where, Gray is the gray value.

Contrast enhancement: The contrast enhancement is to separately process the grayscale values of all the points in the image that have been smoothed. By comparing the grayscale values, level calculation and difference comparison, the difference in grayscale values among the points is more significant. The main application of local mean and standard deviation algorithm is to computer local average of the low-frequency part of the standard deviation, where the high-frequency part of the formula is:

$$\mathrm{m}_{\mathrm{x}}(i, j)=\frac{1}{(2n+1)^2}\sum_{k=i-n}^{i+n}\sum_{l=j-n}^{j+n}x(k,l) \tag{3}$$

The local variance is defined as below:

$$\delta_x^2(i, j)=\frac{1}{(2n+1)^2}\sum_{k=i-n}^{i+n}\sum_{l=j-n}^{j+n}[x(k,l)-m_x(x, j)]^2 \tag{4}$$

Where δx (i, j) is the local standard deviation and f *(i, j)* is used to represent the intensified pixel value corresponding to x *(i, j)*, which can be expressed as follows.

$$f(i, j)=m_x(x, j)+G(x, j)[x(i, j)-m_x(i, j)] \tag{5}$$

The function G (I, j) is not arbitrarily defined. Under normal circumstances, the value of G (I, j) will be greater than 1 to enhance [x (I, j) – mx (I, j)], the high-frequency component. Usually G (I, j) is defined as a constant, it is assumed to be C, and generally C > 1, which will be transformed into:

$$f(i, j)=m_x(i, j)+C[x(i, j)-m_x(i, j)] \tag{6}$$

During the processing of any high-frequency part of the image which are basically of the same magnification, does not rule out that some of the high-frequency components will be over-enhanced or amplified.

Binarization: In the process of grayscale binarization, the values between 0 to 255 to represent grayscale values of the image pixels are used. The whole image has only two colours of with black and white, so that the grayscale features can be handled easily and quickly [20]. In this system, partial adaptive binary image processing is selected, and the grayscale is mainly cut into N modules according to a specified rule. A threshold T is set for each module, and each pixel in the module is respectively adjusted according to a T assigned 0 or 255, thereby completing the binarization of the module, and the remaining N-1 modules are also binarized in the similar way. In

local adaptive binarization, each module threshold T according to a combination of local characteristics formulated threshold calculation formula, as shown in Equation 7.

$$T = a*E + b*P + c*Q \tag{7}$$

Where a, b, and c are free parameters, E is the average of the module's pixels, P is the squared difference between the pixels, and Q is the root mean square between the pixels so that the binarized image can be more pronounced.

Haar-like features: Currently, face detection methods are mainly divided into two types: one is based on existing knowledge and other is based on statistics. Haar-like feature belong to the latter category. In 2004, Viola and Jones proposed face detection using the Haar-like feature and integral graph method [8]. Later, Rainer Lienhart and Jochen Maydt extended the Haar-like feature and eventually formed the Haar classifier which is currently used by OpenCV [21]. Haar-like features, have been commonly used in several applications such as object detection, and face recognition [19]. Haar includes four types of feature templates: edge features, linear features, centre features and diagonal features as the basic elements of decision. The feature template consists of black and white rectangles, and the eigenvalue F of a template is given by white rectangular pixels and Sw minus black rectangular pixels and Sb. By setting the sub-window template category, location and size of the rectangle, a large number of eigenvalues can be collected from the image. For example, using a detection window of 24 * 24-pixel size, the number of rectangular features that can be obtained will reach over 160,000 features.

2.2 Image building algorithm

The integral image needs to pass the image traversal only once, then the pixel sum of all the areas in the image can be calculated. This is a fast algorithm and greatly improves the eigenvalue calculation efficiency. The construction principle of the integral image is that the value I (x, y) at the position (x, y) is the sum of all the pixels in the upper left corner of the original image (x, y). The details of algorithm are as follows.

S(x, y) represents the sum of the row direction, initialize S(x, -1) = 0; an integral image is represented by I (x, y), initialized by I (-1, y) = 0; By progressively scanning the image, the summation S(x, y) and the integral image I (x, y) values in the row direction of each pixel point (x, y) are respectively calculated by the following equations and expressions.

$$S(x , y)=S(x , y-1)+I(x , y) \tag{8}$$

$$I(x , y)=I(x-1 , y)+S(x , y) \tag{9}$$

To do a traversal of the image, it is clear that once the bottom right pixel is reached, the construction of the integral graph I is declared complete. After completion of the integral graph construction, for any matrix area P in the image, the corre-

sponding four vertices are α, β, γ, δ, then the P pixels can be calculated by the following formula:

$$Psum = I(\alpha)+I(\beta)-(I(\gamma)+I(\delta)) \tag{10}$$

The Haar-like feature value is essentially the difference between the sum of two matrix pixels and therefore can be completed in a very short period of time.

2.3 Image building algorithm

The important method of face recognition in the system is the process of template matching, which uses an algorithm provided in OpenCV, named normalized variance matching method. Finding the most similar area in a source image with a known template image is called template matching. The objective function is the Match Template function. Its function is to find the similarity between each position of the source image I and the template image T, and store the similarity result in the result matrix R. The brightness of each point of the matrix represents the similarity with the Template Matching degree. The normalized square difference matching algorithm formula is as shown in Equation 11. Here, T is defined as a template image, I is defined as a source image, and R is defined as a matching result.

$$R(x-y) = \frac{\sum T(x',y') - I(x+x',y+y')^2}{\sqrt{\sum T(x',y')^2 * \sum I(x+x',y+y')^2}} \tag{11}$$

3 Development Environment and Implementation

In this Section, the design details of development environment of the proposed system are described. The steps in this design include building Android development environment [22], using OpenCV libraries, using JNI (Java Native Interface) technology [23], Android NDK (Native Development Kit) [24], and Android SDK [25] components. The interconnection of these tools is shown in Figure 4.

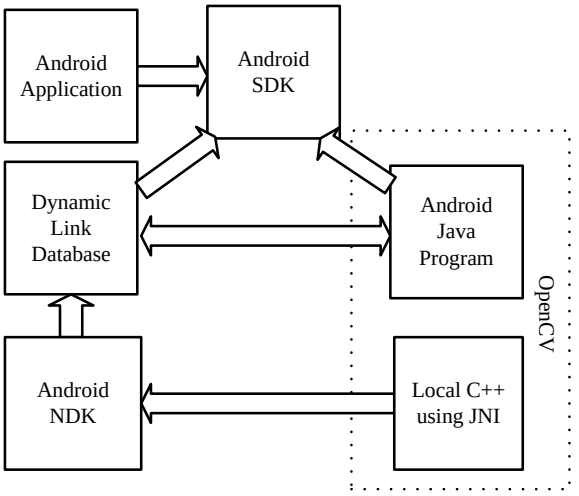


Fig. 4. Overview of development environment

3.1 Building android based development environment

Android Software Development Kit (SDK) support Java and Android Studio provides the integrated development environment for developing Android based applications. In recent years, there are several researchers focused their attention on developing Android APPs and systems based on Android Platform [26,27]. The Android NDK supports native development of C/C++ and allows programmers to develop Android applications using C/C++, and call libraries such as OpenCV to Android platform. So, to use OpenCV in the standard Android development environment, also need tools such as Android NDK, and SDK. Android NDK is a tool set, integrated Android cross compiler environment, and help developers to quickly develop C/C++ shared library. The face recognition algorithm is implemented by C language, and by calling libraries available in OpenCV. This provides higher implementation efficiency than using only the Java language. Android system acts as the application layer, where the programs are written in Java language and it provides JNI interface, so that a Android program can easily call the C language. The JNI interacts between the local library and the Java framework as shown in Figure 4.

3.2 OpenCV library

OpenCV is an open source library of image processing algorithms based on C or C++ programming. OpenCV libraries have the advantages such as: Cross-platform, independent of operating system, hardware, and graphics manager; OpenCV is free of charge for non-commercial or commercial applications; fast and easy to use; good scalability including low-level and high-level application development kits for common image or video load, save and capture modules [21]. As the devices with Android OS has the capabilities to obtain the benefits provided by OpenCV library

which also guarantee real-time, an image preprocessing algorithm is used to form a face recognition system.

3.3 JNI framework

Currently, most of the Android developments are achieved through Java language, and it supports the applications written in C/C++. As a local language C++, and Java will be required to be run on the JVM (Java Virtual Machine) to as a driver software to interact with the hardware directly. So, system libraries are completed in C/C++ and Java language is used to call several functions related multimedia, SQLite, and OpenGL etc., where are they bundled as system packages. In this paper, the image preprocessing, face detection, and face classification authentication algorithm are accomplished in OpenCV class library function and compiled as a library, and JNI framework is used to call them, and the steps involved in JNI call are shown as follows:

1) Create a new JNI folder in the project's root directory.
2) Create a C file in the root folder and add the header file to the folder.
3) In the Java code, create a local method, such as the method named:

```
helloFromC (method body by the C language) such
as :
public native String helloFromC();
```

4) In JNI definition function to achieve this method, the function in the form:

```
jstringJa-
va_com_itheima_helloworldl_MainActivity_helloFr
omC (JNIEnv* env,jobject obj);
```

This returns a string that defines a string in C language:

```
char*cstr= "hello from c";
```

The C language string into java language string:

```
jstringjstr= (*env)- >New-
StringUTF(env,cstr);return jstr;
```

5) In the JNI, create Android.mk and fill out the following in this format:

```
  LOCAL_PATH:=$(callmy-dir)
include $(CLEAR_VARS)
LOCAL_MODULE:=""
```

```
LOCAL_SRC_FLIES:=""
include$(BUILD_SHARED_LIBRARY)
```

6) In the JNI folder under the implementation of ndk-build.cmd instructions, configure the ndk-build environment variable, and refresh the project lib under the armbi file.
7) Use java code to load the so class library, call the local method, by setting the parameter string.
8) Set the support based on implementationarchitecture such as x86 architecture, build it inside JNI, by setting APP_ABI parameter.

3.4 Android native development kit (NDK)

The Android NDK is a set of components based on C/C ++, which can be used to write some of the modules through C/C++ code, and the code of these modules can also run in the Android Virtual Machine. NDK can guarantee higher performance requirements, and often used in applications that require higher security, because these can prevent recompilation or de-compilation. Similarly, the NDK can easily reuse the existing C/C++ modules and can call C++ library. In this paper C/C++ version of OpenCV is used, and compared to OpenCV for android, NDK has more powerful functions. NDK includes API, ARM, x86, MIPS cross compiler, debugger, Java native, and build tools, with dynamic link library. The face recognition algorithm presented in this paper is implemented using C language, and by calling OpenCV library. This has a higher implementation efficiency than using the Java language alone. Android system application layer use Java language, but the Android system also provides JNI interface, so that the Android program can easily call the C language. JNI is located between the local library and the Java framework layer, and this is shown in Figure 4.

3.5 Implementation details

The implementation process is a is a two-step process with face detection as the first step in face recognition. In this section, we will describe the algorithms behind the face detection, and face recognition. Face detection is the first step in face recognition, where it is important to decide whether the image captured is a face or contain information related to human face. The implementation of this method is described as follows. The OpenCV has its own org.opencv.android.JavaCameraView custom control tool, which cyclically crawls data from the camera. In the call-back method, we can obtain the matrix data, and then by calling OpenCV native method. To detect whether there is a face in the obtained image, we will enclose within a rectangle array with face data as shown in Figure 5. The code responsible for this process is shown below:

```
public MatonCameraFrame(CvCameraViewFrameinputFrame){
```

```
  mRgba = inputFrame.rgba();
  mGray = inputFrame.gray();
  if(mAbsoluteFaceSize == 0) {
    int height = mGraw.rows();
    if(Math.round(height* RELATIVE_FACE_SIZE)>0)
   }
      mAbsoluteFaceSize =
Math.round(height*RELATIVE_FACE_SIZE);
    }
  }

  if(mJavaDetector !=null) {
    MatOfRect face = new MatOfRect();
  mJavaDetector.detectMultiScale(mGray,faces,1.1,10,
      |CV_HAAR_FIND_BIGGEST-OBJECT
      |CV_HAAR_DO_ROUGH_SEARCH
      |CV_HAAR_DO_CANNY_PRUNING,
new Size(mAbsoluteFaceSize, mAbsoluteFaceSize),
new Size(mGray.width(),mGray.height()));
    Rect[] facesArray = faces.toArray();
    for (RectaFacesArray : FacesArray){
Core.rectangle(mRgba, aFaceArray.tl(),
AfacesAr
ray.br(),FACE_RECT_COLOR, 3);
      if(null !=mOnFaceDetectorListener){
mOnFaceDetectorListner.on
Face(mRgba.aFacesArray);
      }
    }
  }
  return mRgba;
}
```

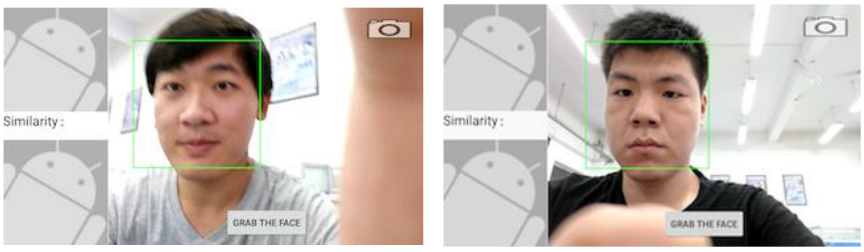


Fig. 5. Initial results after detecting a face

Face recognition is the major step in this process. The face recognition step evaluate the similarity information by obtaining the eigenvalues of the face, and then by comparing the two eigenvalues. In OpenCV eigenvalues are represented as a picture, and after the face detection from the callback method of matrix data, we can extract the eigenvalues and, then compare the eigenvalues. In order to improve the accuracy of identification, face need to be detected, must be identified as a human face, converted into a gray scale, and then normalized. The major sections of program code responsible for this process is shown as below:

```
  public static booleansaveImage(Context context, Mat Im-
age, Rectrect,StringfileName)
  {
    Mat grayMat = new Mat();
  imgproc.cvtColor(image, grayMat, Imgproc.
COLOR_BGR2GRAY);
    Mat sub = grayMat.submat(rect);
    Mat mat = new Mat();
    Size size = new Size(100, 100);
    Imageproc.resize(sub, mat, size);
  return Highgui.imwrite(getFilePath(context, fileName),
mat);
  }
```

The extracted facial features are compared with the facial features of stored in the database. A face is accurately identified based on the most similarity that of the eigenface. The part of implementation coding of this step is shown as following:

```
  public static double compare(Context context, String
fileName1, String fileName2){
    try{
  String pathFile1=getFilePath(context, fileName1);
  String pathFile2=getFilePath(context, fileName2);
  IplImage image1=cvLoadImage(pathFile1,
CV_LOAD_IMAGE_GRAYSCALE);
  IplImage image2=cvLoadImage(pathFile2,
CV_LOAD_IMAGE_GRAYSCALE);
      if(nul==image1|| null==image2){
        return -1;
      }
      int 1_bins = 256;
      int hist_size[] = {1_bins};
      float v_ranges[]={0,255};
      float ranges[][]={v_ranges};
      IplImage imagearr1[]={image1};
```

```
    IplImage imagearr2[]={image2};
CvHistogram Histogram1=CvHistogram.create(1,hist_size,

CV_HIST_ARRAY, ranges, 1);
CvHistogram Histogram2=CvHistogram.create(1,hist_size,

CV_HIST_ARRAY, ranges, 1);
    cvCalcHist(imageArr1,Histogram1,0, null);
    cvCalcHist(imageArr1,Histogram2,0, null);
    cvNormalizeHist(Histogram1,100.0);
    cvNormalizeHist(Histogram2,100.0);
double c1 = cvCompareHist(Histogram1,Histo
gram2,
CV_COMPCORREL)*100;
double c2 = cvCompareHist(Histogram1,Histogram2, CV-

COMP_INTERSECT);
    return (c1+c2)/2;
  } catch(Exception e){
    e.printStackTrace();
    return -1;
  }
}
```

4 Testing and Results

Testing is done with two different smart phones to evaluate the accuracy and acceptability of face detection and recognition methods implemented. In the first case, the testing is carried out on Huawei Mate9 Android 7.0 smart phone, which has a 20-megapixel camera, and Kirin 960 CPU. In the second case, the testing is carried out on Xiaomi 5 Android 6.0 smart phone, which has a 16-megapixel camera, and MSM8996 CPU. Face information of ten individuals are added to the database, and testing the face detection and face recognition is carried out in real-time.

Table 1. Test results in Huawei Mate 9 smart phone

	Frame size	**Success rate**	**Time(*ms*)**	**RAM usage (*MB*)**
Face detection	1280*720	95%	15	26
Face recognition	1280*720	80%	181	30

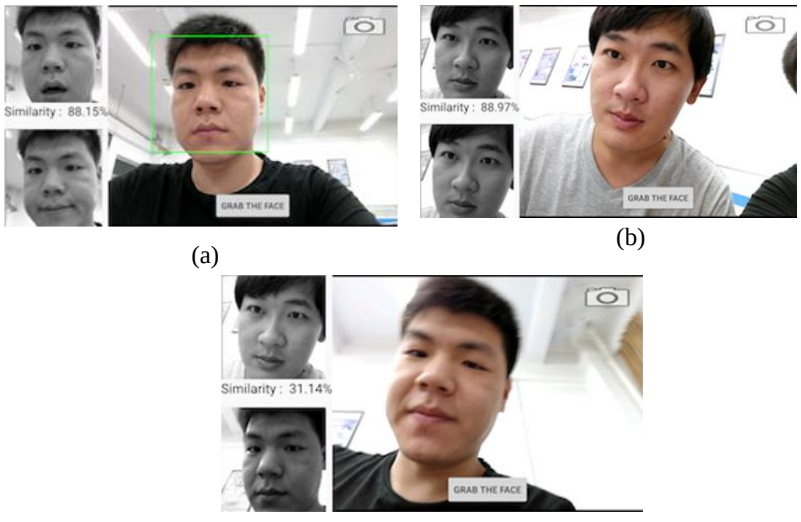
We have extracted different facial eigenvalues and compared their similarity. It is found that the similarity of face eigenvalue of the same person is significantly higher than that of different human face. The Table 1, and Table 2 shows the results obtained in Huawei Mate 9, and Xiaomi 5 smart phones respectively. We have also shown the overall success rate, time taken for face recognition, and RAM usage in megabytes.

The success rates in each case shows successful matching of the images while testing with 20 images. Figure 6 shows the sample display of face recognition results acquired from Huawei mate 10 smart phones. As shown, the similarity level is 88.15% and 88.97% during successful matching, and 31.14% during unsuccessful matching.

Table 2. Test results in Xiaomi 3 smart phone

	Frame size	**Success rate**	**Time(*ms*)**	**RAM usage (*MB*)**
Face detection	1280*720	95%	37	28
Face recognition	1280*720	80%	276	34

We have extracted different facial eigenvalues and compared their similarity. It is found that the similarity of face eigenvalue of the same person is significantly higher than that of different human face. The Table 1, and Table 2 shows the results obtained in Huawei Mate 9, and Xiaomi 5 smart phones respectively. We have also shown the overall success rate, time taken for face recognition, and RAM usage in megabytes. The success rates in each case shows successful matching of the images while testing with 20 images. Figure 6 shows the sample display of face recognition results acquired from Huawei mate 10 smart phones. As shown, the similarity level is 88.15% and 88.97% during successful matching, and 31.14% during unsuccessful matching.



(a) (b)

(c)

Fig. 6. Sample face recognition results, (a) and (b) shows successful matching with similarity rate of 88.97% and 88.15% respectively, whereas (c) shows unsuccessful similarity rate of 31.14% for wrong face.

5 Conclusion

Currently, cameras and microphones are miniature in nature, and can be easily integrated into wearable devices. For the efficient security of the devices from hackers, and confirming authorized users is very important for personal data protection, and to avoid misuse of such information from third parties. So, there are different methods such as speech recognition and face identification methods are very essential. These methods have better advantages over password-based authentication systems, and make only the user to access such devices. Face recognition systems perform well in the limited circumstances, although they exhibit better performance with frontal images and constant illumination. Currently, majority of face recognition algorithms fails in different conditions in which people need to use the idea of this technology. The next generation facial recognition systems should have tools to recognize human face in real-time and in limited and unexpected circumstances.

In this paper, a software development framework for real-time face identification system for mobile devices based on Android platform using OpenCV is presented. We have used supporting software tools such as Java Native Interface (JNI), and NDK (Native Development Kit) along with OpenCV for implementation of the proposed system. For the face detection and recognition, the Gaussian smoothing and gray-scale transformation algorithm is applied to preprocess the image. Then, the Haar-like feature matching method is used to describe the characteristics of the operator and obtain the face characteristic value. The proposed system is tested in real-time in two different brands of smart phones, and results shows average success rate in both devices for face detection and recognition is 95% and 80% respectively.

6 References

[1] Belkhamza, Z., and Niasin, M.A.F (2017). The Effect of Privacy Concerns on Smartphone App Purchase in Malaysia: Extending the Theory of Planned Behavior. International Journal of Interactive Mobile Technologies, 11(5): 178-194. https://doi.org/10.3991/ijim.v11i5.6961

[2] Sato, A., Imaoka, H., Suzuki, T., Hosoi, T. (2005). Advances in Face Detection and Recognition Technologies. NEC Journal of Advanced Technology, 1: 28–34.

[3] Patel, K., Han, H. & Jain, A. K. (2016). Secure Face Unlock: Spoof Detection on Smartphones. IEEE Transactions on Information Forensics and Security, 11(10): 2268–2283. https://doi.org/10.1109/tifs.2016.2578288

[4] Jafri, R. &Arabnia, H. R. (2009). A Survey of Face Recognition Techniques. Journal of Information Processing Systems, 5(2):41–68.

[5] Chen, D., Odobez, J. M. & Bourlard, H. (2004). Text detection and recognition in images and video frames. Pattern Recognition, 37(3): 595–608. https://doi.org/10.1016/j.patcog.2003.06.001

[6] Parlouar,R., Dramas, F., Macé, M. & Jouffrais, C. (2009). Assistive Device for the Blind Based on Object Recognition: an Application to identify Currency Bills. 11th International ACM SIG Access Conference on Computers and Accessibility, Pittsburgh, PA. pp.227–228. https://doi.org/10.1145/1639642.1639688

[7] Chen, X. & Yuille, A.L. (2005). A Time-Efficient Cascade for Real-Time Object Detection: With applications for the visually impaired. IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA. pp. 28–28 https://doi.org/10.1109/cvpr.2005.399
[8] Viola, P.& Jones, M. (2004). Robust Real-Time Face Detection. International Journal of Computer Vision, 57(2):137–154. https://doi.org/10.1023/b:visi.0000013087.49260.fb
[9] Turk, M. & Pentland, A. (1991). Eigenfaces for Recognition. Journal of Cognitive Neuroscience, 3(1): pp. 71–86.
[10] Neto, L. S.B., Maike, V.R.M.L., Koch F.L., et.al. (2015). A Wearable Face Recognition System Built into a Smartwatch and the Visually Impaired User. 17th International Conference on Enterprise Information Systems, SCITEPRESS, pp. 5–12. https://doi.org/10.5220/0005370200050012
[11] Marco, M.D., Fenu,G , Medvet, E. & Pellegrino, F.A. (2017). Computer Vision for the Blind: A Comparison of Face Detectors in a Relevant Scenario. In: Gaggi O., Manzoni P., Palazzi C., Bujari A., Marquez-Barja J. (eds) Smart Objects and Technologies for Social Good. GOODTECHS 2016. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Springer, Cham, 195:145–154. https://doi.org/10.1007/978-3-319-61949-1_16
[12] Elrefaei, L.A.,Alharthi,A., Alamoudi, H., Almutairi, S.& Al-rammah, F. (2017). Real-time face detection and tracking on mobile phones for criminal detection. 2nd International Conference on Anti-Cyber Crimes (ICACC). pp. 75–80. https://doi.org/10.1109/anti-cybercrime.2017.7905267
[13] Mawafo, J.C.T., Clarke, W.A. & Robinson, P.E. (2013). Identification of facial features on android platforms. IEEE International Conference on Industrial Technology (ICIT), Cape Town. pp. 1872–1876. https://doi.org/10.1109/icit.2013.6505962
[14] Zhang, W., Meng, X., Lu, Q., Rao, Y. & Zhou,J. (2013). A Hybrid Emotion Recognition on Android Smart Phones. IEEE International Conference on Green Computing and Communications and IoT and Cyber, Physical and Social Computting, Beijing. pp. 1313–1318. https://doi.org/10.1109/greencom-ithings-cpscom.2013.228
[15] Sudha,V, Viswanath, G., Balasubramanian, A., Chiranjeevi, Basant, K P. & Pratibha, M. (2015). A fast and robust emotion recognition system for real-world mobile phone data. IEEE International Conference on Multimedia and Expo Workshops (ICMEW), Turin, pp. 1–6. https://doi.org/10.1109/icmew.2015.7169787
[16] Eyben, F., Huber,B.,Marchi,E., Schuller, D., & Schuller, B. (2015). Real-time robust recognition of speakers''emotions and characteristics on mobile platforms. International Conference on Affective Computing and Intelligent Interaction (ACII), Xian. pp.778–780. https://doi.org/10.1109/acii.2015.7344658
[17] Grgurina, R., Brestovac, G. & Grbac, T.G. (2011). Development environment for Android application development: An experience report. 34th International Convention MIPRO, Opatija. pp. 1693–1698.
[18] Chillaron, M., Dunai, L.,Fajarnes, G.P. & Lengua, I.L. (2015). Face detection and recognition application for Android. 41st Annual Conference of the IEEE Industrial Electronics Society, Yokohama. pp. 3132–3136. https://doi.org/10.1109/iecon.2015.7392581
[19] Nasrollahi, K. & Moeslund,T.B. (2013). Haar-like features for robust real-time face recognition. IEEE International Conference on Image Processing, Melbourne. pp. 3073–3077. https://doi.org/10.1109/icip.2013.6738633
[20] Chaki, N., Shaikh, S.H. & Saeed, K. (2014). A New Image Binarization Technique Using Iterative Partitioning. In Chaki, N et. al (Ed.), Exploring Image Binarization Techniques, Springer, 560: 17–44. https://doi.org/10.1007/978-81-322-1907-1_3
[21] OpenCV (2018). OpenCV Library.https://opencv.org/

[22] Android Studio (2018). https://developer.android.com/studio/install.html
[23] Java Native Interface (2017). https://docs.oracle.com/javase/8/docs/technotes/guides/jni/
[24] Android NDK. (2018). https://developer.android.com/ndk/index.html
[25] Android SDK. (2018). http://developer.android.com/tools/sdk/ndk/
[26] Huang, H. (2018). Design and Implementation of a College English Listening Learning System Based on Android Platform, International Journal of Emerging Technologies in Learning (iJET): 13(7): 43-56. https://doi.org/10.3991/ijet.v13i07.8779
[27] Feng, L., Yang, C., Zheng, W, and Fu, P. (2017). Android APP Development of Remote Wireless Automatic Meter Reading System based on 3G. International Journal of Online and Biomedical Engineering, 13(12): 18-33. https://doi.org/10.3991/ijoe.v13i02.6439

7 Authors

Laxmisha Rai received Ph.D from Kyungpook National University, South Korea in 2008. From 2008 to 2009, he was a Postdoctoral Researcher with Soongsil University, South Korea. He is a Professor with the College of Electronic and Information Engineering, Shandong University of Science and Technology, Qingdao, China. His research interests include software engineering, real-time systems, embedded systems, autonomous mobile robots, expert systems, wireless sensor networks, MOOC, and Bilingual Education. He has published over 50 research papers and currently serving as Associate Editor of IEEE Access Journal. He is a Senior Member of IEEE, and Member of ACM.

Zhiyuan Wang is currently pursing his Master's degree in College of Electronic and Information Engineering at Shandong University of Science and Technology, Qingdao, China. His research interests are mobile application development, image processing, and real-time systems.

Amila Rodrigo is currently pursing his Master's degree in College of Electronic and Information Engineering at Shandong University of Science and Technology, Qingdao, China. His research interests are real-time systems, antenna applications, and Internet of Things (IOT).

Zhaopeng Deng received his Bachelor's and Master's degrees from Shandong University of Science and Technology, China in 2010 and 2013 respectively. His major research interests include image processing and machine vision. Currently pursuing his Ph.D studies at Shandong University of Science and Technology, China.

Haiqing Liu received the bachelor's degree in automation from Central South University, China, in 2008, the Ph.D in system engineering from Shandong University, China, in 2015. From 2015 to 2017, he was a Postdoctoral Researcher with the Postdoctoral Work Station, Hisense Group, China. He is a lecturer in Shandong University of Science and Technology. His current research interests include traffic engineering and control, cooperative vehicle infrastructure system and traffic intelligent perception.

Article submitted 2019-10-28. Resubmitted 2019-12-16. Final acceptance 2019-12-17. Final version published as submitted by the authors.