International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-3 Number-3 Issue-11 September-2013

Face Recognition using Depth Map Generation for Smart phones

Anshu Sisodhiya[1], Ankita Jain[2], R M Banakar[3]

Abstract

Face Recognition now a day's is one of the most popular biometric technologies because of its ease of use and simplicity. Face recognition using depth map is an emerging field, which is trying to use 2D features along with the depth information of the face for accurate identification of the subject. In this paper, a 3D face recognition system is proposed where depth information of the face is captured by calculating the deviations in the scan pattern because of the non-planarity of the face. By using this information a depth map is generated and 3D features are extracted from it by implementing Fourier Descriptor algorithm. This 3D feature is then matched with the 3D features of different faces stored in the database to identify an individual.

Keywords

Android, Depth map, Face recognition, Fourier descriptor, Mahalanobis distance.

1. Introduction

The identification of the human face in 2D has been studied by many researchers, but comparatively few studies have been carried out in the field of 3D face recognition [1]. The challenge with 2D representation techniques is that every 3D object is mapped into a 2D plane because of which it loses its depth information thereby making it more inaccurate. 3D features provide the information which does not affected by change in translation, illumination and posture. Here, we are proposing a simple method to identify an individual's face from a database comprising of different faces. As this is the first time implementation, so the size of the database is kept concise. The proposed method is a simple, portable, low cost, vision based system for face recognition and depth calculation.

Anshu Sisodhiya, M. Tech (VLSI Design and Testing), BVBCET,Hubli, India.

Ankita Jain, CREST, KPIT Cummins Infosystems Ltd., Pune, India.

R M Banakar, Department of Electronics & Communication, BVBCET, Hubli, India.

2. Motivation

As the technology is becoming advanced, threats for even the most sophisticated security systems are bound to increase. Various government agencies are working on the improvement of security systems by making them depend on body or behavioral characteristics, often called biometrics. It is a very attractive technology because it can be integrated into any application requiring security or access control, effectively eliminating risks associated with less advanced technologies. But, biometrics has some drawbacks too. Iris recognition is extremely accurate, but expensive to implement. Fingerprints are reliable and non-intrusive, but not suitable for non-collaborative individuals. On the contrary, face recognition seems to be a good compromise between reliability and social acceptance and balances security and privacy well [2].

Selection of Android as implementation platform for depth based face recognition is motivated by the increasing need of portability and availability of hardware for the application. As android cell phones have become a very common hardware for the people today, so we have opted android as the implementation platform for our application, so that maximum people can access and utilize the work done by us. As using a cell phone reduces the hardware cost of the whole system because of availability of it with people, it becomes the best choice for the implementation of this application.

3. Literature Review

Facial recognition is a type of biometric computer based software application that can automatically identify or verify a specific individual from a digital image or a frame from a video. One of the ways to do this is by scanning an individual's face and matching it against a library of known faces. The Kinect motion gaming system is one of the best example of it which uses facial recognition to differentiate among various players. Different approaches have been proposed in literature for 3D face recognition [1][2][3]. Some of the approaches are point cloud approach, depth based approach, differential geometry approach and facial curve based approach.

Depth based approach is one of the most popular approaches for 3D face recognition. It can use either camera or sensors for taking facial inputs and constructs a depth image. Each pixel in the depth image represents the distance of the corresponding 3D facial point to the camera. Another important point of concern in depth based approach is that it converts irregularly sampled 3D points to a regular (x, y) grid and to accomplish this task different interpolation methods are used. Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Linear Discriminant Analysis (LDA) are some of the examples for depth based approaches. Depth based approaches are easy to implement as they don't require costly hardware and are translation and rotation invariant providing a little space for face movement while scanning is taking place.

4. Proposed Method

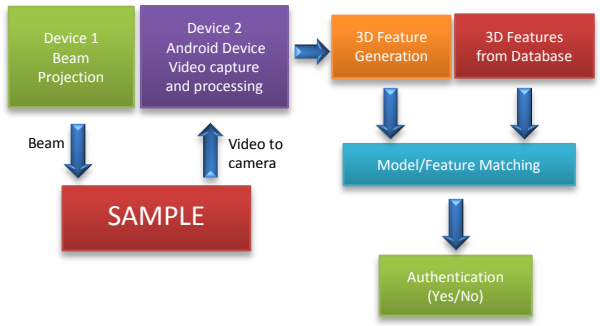

Figure 1: Block Diagram

The block diagram of the proposed method is shown above in Fig. 1. In this system a green beam is projected by the projector on the sample image which is at the same time captured by the camera of the android device. 2D image of object is captured along with the depth information generated by the beam. By using this information depth map for each image is generated which is then processed by implementing Fourier Descriptor algorithm in order to generate 3D features of the face. This 3D feature is then matched with the 3D features of different faces stored in the database. If this generated feature is matched with any of the features present in the database then the user authentication will be successful else user authentication will be failed. We have used Samsung Galaxy S smart phone for the implementation of this system which has android ICS 4.0.3 installed in it.

This section explains the approach implemented by us for face recognition. It explains real time data fetching and extraction of useful information from it. The whole process of fetching, depth map generation and feature extraction is explained in detail here.

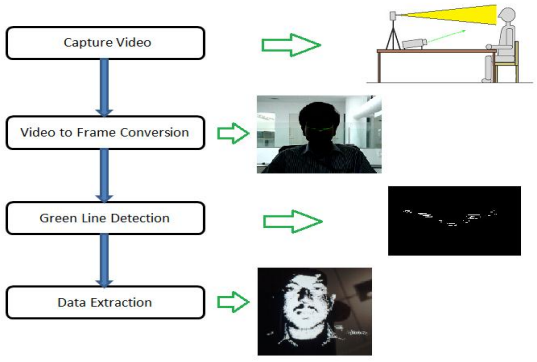

Figure 2: Data Preprocessing

Fig. 2 shows the overview of data preprocessing. The first step in data preprocessing is to capture video. The setup for capturing video is shown in the Fig. 2. As shown, the setup comprises of a projector and a cell phone mounted on a stand at an appropriate distance from the individual [3]. The person whose face is going to be scanned is supposed to sit straight facing the camera. Video is taken at a rate of 30fps which can be varied according to the accuracy required. The green line projected on the face covers 1pixel/frame which means with every successive frame, line moves by 1 pixel which ensures that we capture all the pixels of face for a better data acquisition. Line moves from bottom to top of the face and covers full face within 17 seconds. Speed of projected line can be reduced to get more data which increases the accuracy of the system.

After capturing video, the next step in data preprocessing is to convert it into frames. For this project, we have used an external utility called "FFmpeg" to convert video into frames. Once we get the frames from the video, we are ready with the raw material for processing. In the next step, we have to filter the necessary data using green line detection which includes conversion of each frame from RGB to HSV, followed by setting the range for H, S and V to extract required data points from each frame. Fig. 3 shows the extracted green line from a frame and the complete face, generated by adding the data points obtained from the green lines of all the frames.

Figure 3: Extracted line and face

Depth Map Generation

Depth Map is defined as an image that contains information relating to the distance of the surfaces of scene objects from a viewpoint. In depth map generation, depth information related with various curves of face is extracted and plotted using color contours. Colors in depth map represent the different distances of objects in an image. We are using depth based face recognition technique as it is easy to implement and does not require expensive hardware.

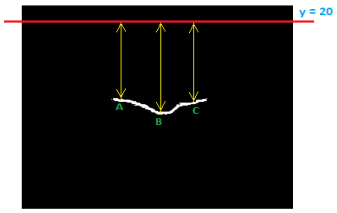

Figure 4: Depth Calculation

For depth map generation, first step is to calculate depth of each pixel of the face extracted in preprocessing. Depth is calculated separately for each frame and then copied into a matrix. In order to calculate depth, first a reference line is defined. At starting, the reference line is taken at y=1 and then it keeps on moving with the frames. The distance between the reference line and a face pixel for a particular frame gives the depth information of that pixel. This process is repeated for all the frames to obtain the depth map of a complete face. Fig. 4 shows the overview of depth calculation. The depth calculation of three points A, B, C is shown here. Consider,

A = (100, 200), B = (150, 220), C = (200, 210)

Then, the depth of A, B, C will be calculated as,

$A = Y_A - Y_R = 200 - 20 = 180$

$B = Y_B - Y_R = 220 - 20 = 200$

$C = Y_C - Y_R = 210 - 20 = 190$

The reference line will be incremented with each frame and the factor with which it gets incremented is calculated by the average distance between two successive frames. This incrementation factor can vary from 1 to 3 pixels. After calculating depth of each pixel using reference line, now the depth map can be generated. The matrix where all the depth values are copied gives the depth map for the desired face. Once the depth map is generated, we can use it to create a color map which displays the depth information of different parts of face in the form of different colors. The normal depth map for a sample face is shown in Fig. 5.

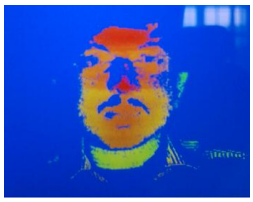

Figure 5: Depth map

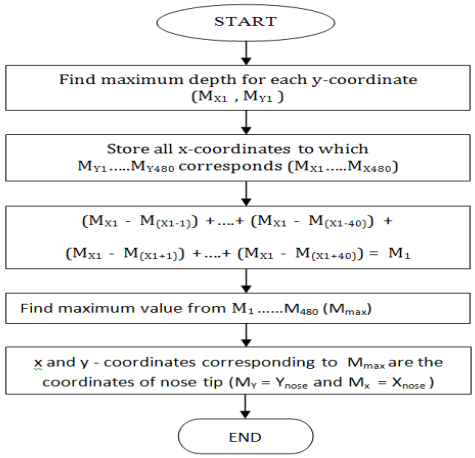

Figure 6: Flowchart for nose tip detection

In this section, the process after depth map generation is explained which includes the detection of features which are necessary for face recognition and then storing these features in the database. Generally, face recognition process takes into account some of those features which have maximum variations and cannot be common between two or more individuals. For this reason, we have considered middle part of the face for feature extraction process. But, to identify the middle part of the face, first we have to detect the nose tip which defines the center of a face.

After depth map, Nose tip location is the next crucial parameter in face recognition process. Flowchart in Fig. 6 explains the algorithm of finding nose tip. The nose tip coordinates for this face is found as below:

X = 321, Y = 229

Based on the nose tip coordinates, nose tip frame is identified which is 132 in this case. Once the nose tip is detected for the given face, now we can find out the required region from where we have to extract features. In this paper, we have considered 40 frames next to nose tip frame and 39 frames previous to nose tip frame for feature extraction.

Before feature extraction, ear removal must be done as it can alter the periphery of face while feature extraction. Ear removal is done by defining two regions on each side of the face and then subtracting all consecutive pixels to find out the maximum difference. The first pixel which gives maximum difference or difference above a preset threshold is the last pixel of the face. Once we have identified the last pixels for each side of the face then the rest of the pixels which comes after it will be removed. After ear removal, now we have to implement Fourier descriptor to extract features of the face. For applying Fourier transform on the shape, shape must be in a continuous closed loop. The shape information obtained from the green line detection does not provide a continuous output, so to achieve that different points on the line must be connected with their neighboring points using straight line equation. Consider two points (x1, y1) and (x2, y2) which need to be connected in x-axis to form a continuous closed loop. Then the equation used is,

Y = (c - a *(x1 + q)) / b
Where,
a = (y2 - y1)
b = - (x2 - x1)
c = - (((x2 - x1) * y1) - ((y2 - y1) * x1))
q = 1, 2……., (x2 - x1)

Figure 7: Green line before and after closed loop

The original green line obtained in data preprocessing and the output after continuous closed loop formation is shown in Fig.7. After closed loop formation total 318 points are obtained on the boundary of the shape but we cannot process all these points. So, we have selected alternate 64 points which falls on the boundary of the shape for normalization. The normalization process successfully eliminates the noise and small details of the shape which can affect robustness of shape matching. Now, in order to find FFT, first we have to find the centroid of the shape and then distance of each sampled point from the centroid is calculated [4]. Here we have used centroid signature for shape description. Formula for centroid calculation is given below:

Assume the shape boundary coordinates as (x(t), y(t)), t = 0, 1, ..., N-1. Then, the centroid (x_c , y_c) of the shape which is the average of the boundary coordinates will be given by,

$$x(c) = \frac{1}{N}\sum_{t=0}^{N-1} \mathrm{x(t)}, \quad y(c) = \frac{1}{N}\sum_{t=0}^{N-1} \mathrm{y(t)}$$

Now, the distance of a data point z(t) from centroid will be calculated as,

$$r(t) = \sqrt{([x(t) - x_c]^2 + i[y(t) - y_c]^2)}$$

The centroid obtained is shown in Fig. 8

Figure 8: Centroid of shape

After obtaining the distance of all the 64 data points from the centroid, first 32 points were chosen for further processing. FD calculation is explained below:

Let x[m] and y[m] be the coordinates of the m^{th} pixel on the boundary of a given shape, a complex number can be formed as z[m] = x[m] + jy[m], and the Fourier Descriptor (FD) of this shape is defined as:

$$Z[k] = DFT[z[m]] = \frac{1}{N}\sum_{m=0}^{N-1} \mathrm{z[m]} e^{-j2\pi mk/N}$$

k = 0,1,2,…….,N-1

Where, N is the total number of pixels along the boundary. FD calculation gives the 32 values for each of the 80 frames. After acquiring this feature information, now we have to store it in a database. For this paper, we have created database in Microsoft Excel rather than using Android's inbuilt Sqlite database because of its ease of use and accessibility from any platform. Data for each frame is in the form of 80x32 matrix which can be stored directly in the Excel sheet's rows and columns. In order to test this system, in this paper we have created a limited database of faces. Number of faces in database can be further increased to N numbers at a later stage.

5. User Identification

User identification is a very important part of face recognition process as the reliability of the system is based on the fact that how efficiently an algorithm matches two samples. In user identification, first, sample face is preprocessed and then depth and feature information are extracted as explained in previous sections. In identification, the extracted features in 80x32 matrix is not stored in database but instead stored in a temporary array or matrix for matching. In order to find out whether sample face is matching with any of the faces in the database or not, distance of sample face with each face in the database is calculated using Mahalanobis distance metric algorithm [5]. The Mahalanobis distance is a descriptive statistic that provides a relative measure of a data point's distance (residual) from a common point. For example, if $\vec{x}$ and $\vec{y}$ are the two vectors which has covariance matrix S then, the Mahalanobis distance in between these two vectors is given by the following equation,

$$d(\vec{x},\vec{y}) = ((\vec{x}-\vec{y})^T S^{-1}((\vec{x}-\vec{y}))^{1/2}$$

Where, Covariance matrix S is given by

$$S = \frac{1}{n-1}\sum_{i=1}^{n}(\mathrm{x_i}-\bar{\mathrm{x}})\,(\mathrm{x_i}-\bar{\mathrm{x}})^{\mathrm{T}}$$

Where, x_i is the i^{th} observation of the (p-dimensional) random variable and $\bar{x}$ is given as,

$$\bar{\mathrm{x}} = \begin{bmatrix}\bar{\mathrm{x}}_1\\ \vdots \\ \bar{\mathrm{x}}_\mathrm{p}\end{bmatrix} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{x_i}$$

After calculating mahalanobis distance of sample face from each face stored in the database, then a face is identified from which distance is minimum. If the minimum distance obtained is more than a preset threshold value then we can say that the face is matched or verified and is present in the database. As, we never get a 0 distance while comparing two faces we have to set a minimum threshold in order to get a nearest match.

6. Results

The output of feature extraction and database creation is shown in Fig. 9.

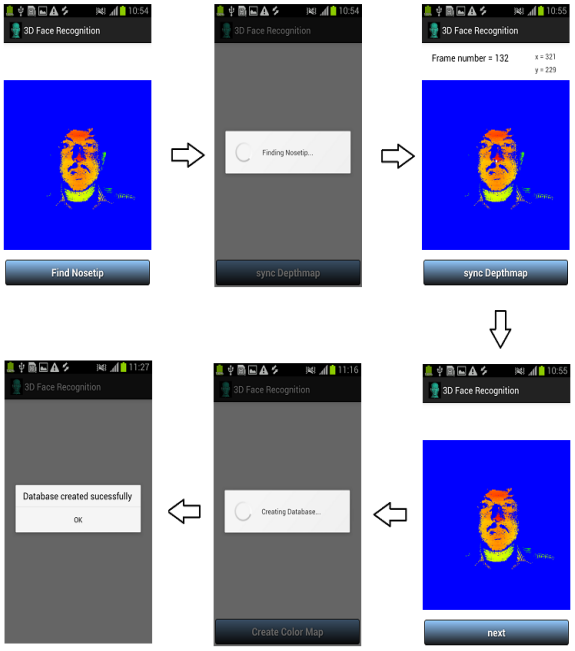

Figure 9: Output of Feature Extraction and Database Creation

After database creation, sample face is matched with the faces in the database. The overall process and output of face recognition system is shown in Fig.10. F1, F2, F3..., Fn are the features of faces stored in database. First, features of face 1 are matched with the features of sample face which gives mahalanobis distance M1 = 45.08783, then face 2 and face 3 are matched with the sample face, resulting in distance M2 = 30.76893 and M3 = 5.098642. In the same manner, distance of all the faces stored in the database is calculated. After getting mahalanobis distance for all the faces, the face with minimum distance is identified. In this case the minimum distance is 5.098642. The face with minimum distance is having nearest resemblance with the sample face. Then, this distance is compared with the threshold value set for the matching process which is taken as 10 in this case. If this distance is less than the threshold value, then we can say that the match is found. As the distance of face 3 is less than the threshold value, then we can say that the sample face is matching with the face 3 stored in the database.

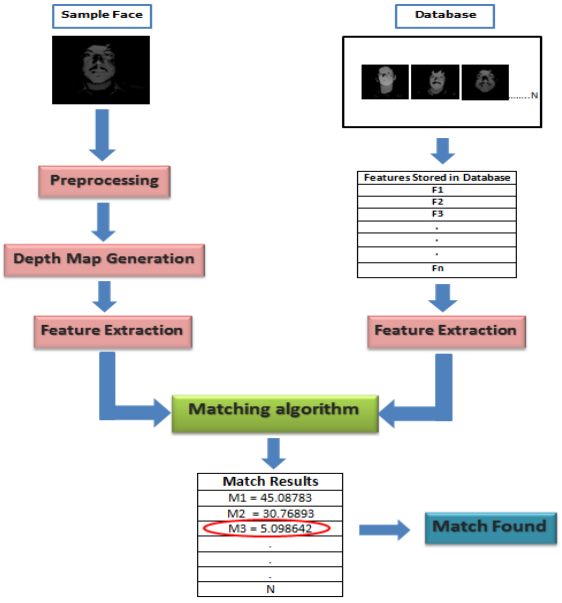

Figure 10: Face recognition system output

7. Conclusion and Future Work

In this paper, we have proposed a portable face recognition system, where based on the results obtained we can say that it is possible to run this system successfully on a portable device running android operating system. The system is tested for a limited database for which it performs reasonably well. So, it can be further extended for a bigger database. As compared to other security systems, our system provides a low cost solution by using a normal cell phone which is already available with people. So, it simply ends the necessity of buying a new hardware which reduces the overall system cost. As the system uses depth based approach, it is much more reliable and accurate as compared to other face recognition system which uses 2D approaches. The scope of future work in this paper definitely can be the testing of the system for a larger database. We can also reduce the time taken for processing by optimizing the code in various ways, like by defining region of interest while scanning the individual's face which can reduce the processing time by a significant amount.

Acknowledgment

Authors thanks KPIT Cummins Infosystems Ltd. and B. V. Bhoomaraddi College of Engineering and Technology, for providing an excellent platform, necessary hardware and valuable guidance for conducting this research.

References

[1] Preeti B. Sharma, Mahesh M. Goyani, "3D Face Recognition Techniques - A Review", International Journal of Engineering Research and Applications (IJERA), pp. 787-793, Jan-Feb 2012.

[2] Andrea F. Abate, Michele Nappi, Daniel Riccio, Gabriele Sabatino, "2D and 3D face recognition: A survey", Elsevier B.V., pp. 1885-1905, 2007.

[3] Philipp Fechteler, Peter Eisert and Jsurgen Rurainsky, "Fast And High Resolution 3d Face Scanning", International Conference on Image Processing-ICIP , pp. 81-84, 2007.

[4] Dengsheng Zhang and Guojun Lu, "A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures," Journal of VisualCommunication and Image Representation, pp. 41-60, 2003.

[5] Pranali Dhane, Ankita Jain and Krishnan Kutty, "A new algorithm for 3D object representation and its application for human face verification", IEEE Inter.

[6] National Conference on Image Information Processing (ICIIP), pp. 1-6, 2011.

Anshu Sisodhiya received the B.E degree in Electronics & Communication Engineering from GEC, Modasa in 2009 and pursuing M.Tech in VLSI Design and Testing from BVBCET, Hubli. She has 2 years of experience in embedded development. Her areas of interest include Embedded Electronics and Android Development.

Ankita Jain is working at KPIT Cummins since May 2010 as a Jr. Scientist. She received M.Tech Degree from ABV-IIITM, Gwalior in VLSI Design. Her areas of interest are image processing and data analytics.

Dr. R M Banakar received the M.Tech Degree from K.R.E.C., Surthkal in 1984 and completed PhD from IIT Delhi in 2004. She has above 25 years of experience in teaching.