Architecture for Mobile based Face Detection / Recognition

Shishir Kumar, Priyank Singh, Vivek Kumar
Department of CSE, Jaypee Institute of Engineering & Technology
Guna, India

***Abstract*— In this paper a novel description of Face Detection-Recognition architecture through mobile devices is presented. Since the face detection/recognition is of importance in real-life scenarios, such as for authentication and security services and implementing it in mobile devices would provide a great value to society. This paper presents client-server architecture through the use of Bluetooth Technology through which face detection/recognition phases can be implemented.**

Key words- Face Detection, Eigenface, Feature Extration

I. INTRODUCTION

Due to limited processing power/resources in the mobile devices the face detection/recognition poses various problems. Currently the face detection algorithms use facial feature or skin color based approach for the face detection purpose. For face recognition Eigenface face recognition algorithm is used.

Several face detection algorithms provides accurate results but when it comes to deploying in real-time conditions such as in mobile devices, due to the limited resources, memory, processing power, only a few of these algorithms can be implemented.

In real-time scenario the face recognition algorithms are difficult to be implemented on the mobile devices because of the high processing power needed as well as the low storing capabilities of the mobile devices.

Real-time implementation of the face detection/recognition algorithms on the mobile devices has its own problems if a dedicated processor is not used.

In this paper we have proposed architecture for the face detection/recognition. Client/Server architecture is used, i.e. the face detection is done on the mobile devices (client) and the face recognition is done on the dedicated computer (server).

The face detection phase on real-time mobile devices is implemented giving fairly good results on frontal face but it noticeably drops down for tilted, merged, partially covered faces. Facial feature based face detection algorithm introduced by Viola and Jones has been adopted, which uses a boosted cascade of facial features. The eigenfaces based face recognition is implemented on a dedicated pc which acts as a server and recognizes the images that are coming to it.

This client-server approach provides advantage of speed, flexibility and scalability. The only time takingstep of this approach is the time taken for transferring an image from the client side to the server side.

The client-server architecture can be implemented in two ways:

1) *Client-Server architecture using Bluetooth.*
2) *Client-Server architecture using HTTP protocol.*

Each of the two approaches has its advantages and disadvantages, usage area and is complementary to each other. In this paper, first Introduction of Face Detection and face recognition and then the explanation of proposed architecture is given with experimental results. At last further enhancements are given.

Viola Jones Face Detection

The Viola-Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones in their "Robust Real-Time Face Detection**"** paper[4] which describes a visual object detection framework that is capable of processing images very rapidly while achieving the high detection rates. The three major contributions in Viola-Jones object detection framework are "Integral Image" which is representation of the image which helps in quick detection of features used by our detectors , learning algorithm using Adaboost which is a simple algorithm that selects one feature at a time and assigns weights to the feature and producing a strong classifier and last one is combining classifiers in a "cascade" which is helpful in discarding the background details of the image and focusing on the promising object like regions to make computation faster and more precise than earlier used face detection algorithms.

Feature extraction and feature evaluation.

Rectangular features are used with a new image representation so that their calculation becomes fast. The white areas are subtracted from the black ones. A special representation of the sample called the integral image makes feature extraction faster and they bear some resemblance to Haar basis functions, which have been used previously in the realm of image-based object detection.

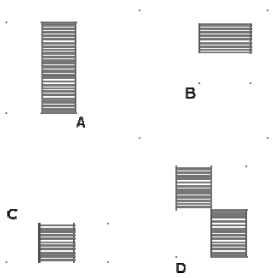

Figure 1: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

Integral Image

The value of any given feature is always simply the sum of the pixels within clear rectangles subtracted from the sum of the pixels within shaded rectangles. As is to be expected, rectangular features of this sort are rather primitive when compared to alternatives such as steerable filters. Although they are sensitive to vertical and horizontal features, their feedback is considerably coarser. However, with the use of an image representation called the integral image, rectangular features can be evaluated in *constant* time, which gives them a considerable speed advantage over their more sophisticated relatives.

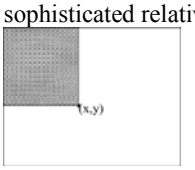

Figure 2.(Adapted from [4]) The value of the integral image at point (*x, y) is the sum* of all the pixels above and to the left.

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y'),$$

Where ii(x,y) is the integral image and i(x,y) is the original image . Using the following pair of recurrences:

$$s(x,y) = s(x,y-1) + i(x,y)$$
$$ii(x,y) = ii(x-1,y) + s(x,y)$$

(where s(x,y) is the cumulative row sum, s(x,-1)=0 , and ii(-1,y)=0) the integral image can be computed in one pass over the original image.

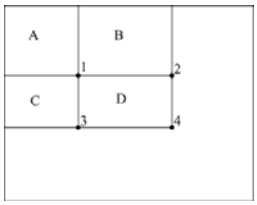

Figure 3(Adapted from [4])**.** The sum of the pixels within rectangle *D can be computed* with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle *A. The value at location 2 is A + B, at location 3 is A + C, and at location 4 is A + B + C + D.* The sum within *D can be computed as 4 + 1 − (2 + 3).*

Feature/Classifier Evaluation

Any number of machine learning approaches could be used to learn a classification function if a feature set and a training set of positive and negative images are given. For example Sung and Poggio uses a mixture of Gaussian model. Rowley, Baluja, and Kanade uses a small set of simple image features and a neural network for learning.

There are 45,396 rectangle features associated with each image sub-window which is much more than the total number of pixels. Each feature can be computed but computing the complete set is much expensive. Viola Jones proposed that a very small number of these features can be combined to form an effective classifier and main aim is to find them. Finally AdaBoost is used both to select the features and to train the classifier. AdaBoost learning algorithm is used to boost the classification performance of a simple learning algorithm (e.g., it might be used to boost the performance of a simple perceptron). It can be done by combining weak classification functions collection and resulting in strong classifier. The simple learning algorithm is called a weak learner in language of boosting because we do not expect even the best classification function to classify the training data well. In order to boost weak classifier to strong classifier, it is called upon to solve a sequence of learning problems. After the first round of learning, the examples are re-weighted in order to emphasize those which were incorrectly classified by the previous weak classifier. The final strong classifier takes the form of a perceptron, a weighted combination of weak classifiers followed by a threshold.

Given a sample set of images.

- For t = 1…T (rounds of boosting)
  - A weak classifier is trained using a single feature.
  - The error of the classifier is calculated.
- The classifier (or single feature) with the lowest error is selected, and combined with the priors to make a strong classifier.
- After a *T* rounds a *T* strong classifier is created.

- It is the weighted linear combination of the weak classifiers selected.

A weak classifier ($h_j(x)$) thus consists of a feature (f_j), a threshold (Θ_j) and a parity (p_j) indicating the direction of the inequality sign:

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

Here x is a 24x24 pixel sub-window of an image.

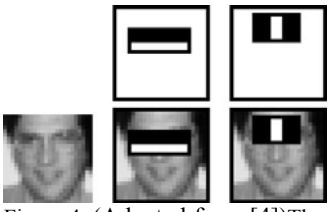

Figure 4. (Adapted from [4])The first and second features selected by AdaBoost.
The two features are shown in the top row and then overlaid on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

The Attentional Cascade

The key feature behind fast computation of algorithm is that simple, smaller and boosted classifiers can reject many of negative sub-windows while detecting all positive instances.
The classifier can significantly reduce the number sub-windows that need further processing with very few operations:
1. Evaluate the rectangle features (requires between 6 and 9 array references per feature).
2. Compute the weak classifier for each feature (requires one threshold operation per feature).
3. Combine the weak classifiers (requires one multiply per feature, an addition, and finally a threshold).

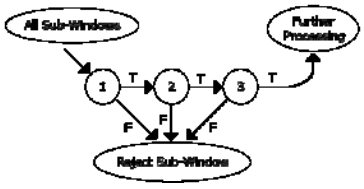

Figure 5. (Adapted from [4]) Schematic depiction of a the detection cascade. A series of classifiers are applied to every subwindow. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing the number of sub-windows have been reduced radically. Further processing can take any form such as additional stages of the cascade (as in our detection system) or an alternative detection system.

More complex classifiers are called to achieve low false positive rates
Series of such simple classifiers can achieve good detection performance while eliminating the need for further processing of negative sub-windows (*in a single image, the majority of sub-windows are negative).
Given a trained cascade of classifiers, the false positive rate of the cascade is

$$F = \prod_{i=1}^{K} f_i,$$

Where F is the false positive rate of the cascaded classifier, K is the number of classifiers, and f_i is the false positive rate of the ith classifier on the examples that get through to it. The detection rate is

$$D = \prod_{i=1}^{K} d_i,$$

Where D is the detection rate of the cascaded classifier, K is the number of classifiers, and d_i is the detection rate of the ith classifier on the examples that get through to it.

Face Recognition

Eigenface face recognition algorithm developed by, Turk and Pentland,[3] is used to recognise the face extracted from the mobile devices. Since the processing capability of standard mobile devices is insufficient for large database so client server architecture is used, recognition is done on the server side (processing device). The algorithm is based on PCA and is fast and robust.

The face recognition through eigenfaces works in the following steps when the face images of persons are given and an unknown face image has to be recognized:

1) Computation of Euclidean distance between the unknown face image and the known face images.

2) Select the known face image that is closest to the unknown one with the help of Euclidean distance (vector).

3) If the vector value is above threshold then consider it recognized otherwise classify as unknown person's face image.

Principal Component Analysis is used for dimensionality reduction, since each face image is a vector and each component of this vector represents pixel, the most correlated components of the vector describes the face distribution. As can be seen from the figure PCA looks for directions that represent efficient data and from Figure 6 it can be seen that scattering has been maximized by PCA.

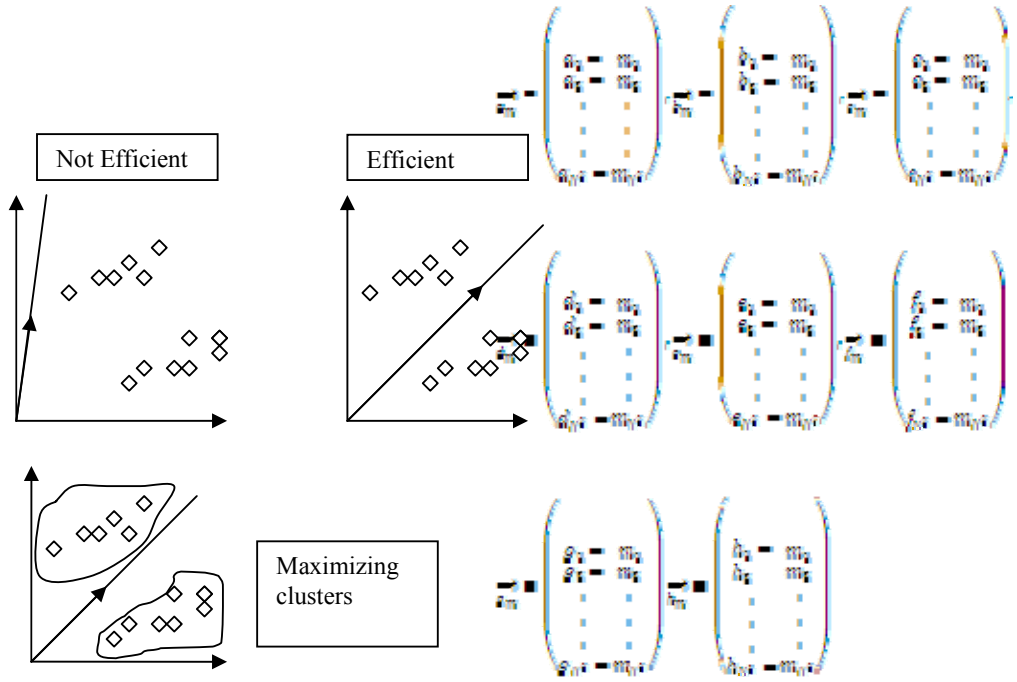

Figure 6.

The face database is represented as matrix of dimension N^2. $\blacksquare$ $\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_{N^2} \end{pmatrix}$

The average face is calculated as:

$$\vec{m} = \frac{1}{M}\begin{pmatrix} a_1 + b_1 + \cdots + h_1 \\ a_2 + b_2 + \cdots + h_2 \\ \vdots \\ a_{N^2} + b_{N^2} + \cdots + h_{N^2} \end{pmatrix}$$

Where M=8.

Each training face is then subtracted from the average face

Now covariance matrix is built of order N^2 by M

$$A = \left[\vec{a}_m \; \vec{b}_m \vec{c}_m \vec{d}_m \vec{e}_m \vec{f}_m \vec{g}_m \vec{h}_m\right]$$

The Covariance of the Matrix is represented by Cov=A A

Now another matrix of order M by M is calculated L=AT A

Now Vector V is made from the eigenvectors from the L matrix, Eigenvectors represents the variations in faces.
The eigenvectors of Cov is represented as U=AV

Now each face is computed for projection on to the face space

$$\Omega_1 = U^T\left(\vec{a}_m\right)\Omega_2 = U^T\left(\vec{b}_m\right)\Omega_3 = U^T\left(\vec{c}_m\right)\Omega_4 = U^T\left(\vec{d}_m\right)$$
$$\Omega_5 = U^T\left(\vec{e}_m\right)\Omega_6 = U^T\left(\vec{f}_m\right)\Omega_7 = U^T\left(\vec{g}_m\right)\Omega_8 = U^T\left(\vec{h}_m\right)$$

Now threshold value is computed:

$$\Theta = \frac{1}{2}max\{\|\Omega_i - \Omega_j\|\} \; for \; i,j = 1 \ldots M$$

Now for the new face[N] that is to be recognized the same steps are performed and result is calculated as:

- IF N>= θ then it is not a face
- IF N< θ and distance>= θ then it is new face.
- IF N< θ and min{distance}< θ then it is known face.

Problems related with Eigenfaces algorithm are with the different illumination areas, head postures, alignment conditions of the camera and different facial expressions.

Client Server Architecture through Bluetooth

In this paper we introduce with client-server architecture through Bluetooth. Application Program Interfaces API's of J2ME are used for providing interaction with the mobile device. The usage of Bluetooth is for the transferring from the mobile device to the processing device and vice-versa in real time scenario.

The training and testing datasets (image database) are required for the face recognition process, these datasets are created/appended at the server (processing device) side. The mobile camera's parameters are important for generation of snapshots for image database. The image database should have sufficient numbers of snapshots for the proper training and testing, snapshots should also be taken under different conditions like different lightening, facial (beard, moustache, etc.)

Now in the client-server architecture mobile device is a client and the processing device (Computer, etc) is a server.

I -CLIENT DESCRIPTION

The client (mobile device) application requires MIDP 2.0[5] profile and Mobile Media API[6] for access to camera of mobile device. The snapshot is taken using methods of Camera Manager Class.

Face detection through voila jones algorithm is done on the snapshot and the ROI (region of interest) which include the detected face is taken. The ROI is then converted into bytes. Bluecove API is used for sending the bytes from the client to server through Bluetooth connection. The converted image bytes are sent to the server for the further recognition phase.

II. SERVER DESCRIPTION

The programming at server side is on Visual C++ and java. Now the bytes are received at the server side by a java bean using bluecove api and converted to jpg image file on the server and an entry is made in either training-images.xml or testing-images.xml. Training-images.xml contains entries of images meant for training purpose for the face recognition phase. Testing-images.xml contains entries of images whose recognition is to be done based on trained images. The face recognition phase at the server side is implemented using eigenfaces algorithm.

The eigenfaces face recognition algorithm is implemented through Intel's OpenCV library.

Table for Training Images.

Image ID	Image Path	PersonID

Table for Testing images

Image ID	Image Path	PersonID

First the training required for the eigenfaces algorithm is done through Training-images.xml entries using the ImageID and Image Path and the person no is assigned to the trained one then the recognition is performed using Testing-images.xml using its IMAGEID and IMAGEPATH and the eigenfaces algorithm is performed on it and if the image is recognized corresponding Person No is assigned to PERSONID field and if it is not recognized then a new PERSONID is generated and assigned to it.

The result containing the PERSONID and name is sent using the bluecove API to the j2me mobile client application and displayed on the User Interface.

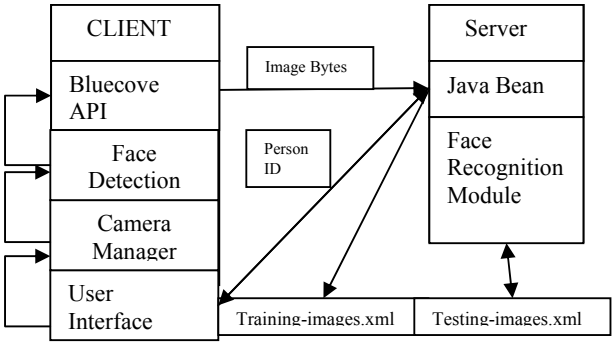

Fig 7. Client Server Architecture through Bluetooth

J2SE PC Application (RCP Server Side)
SUN JSR-82 API
BlueCove Bluetooth Stack
Java Native Interface (JNI)
MS Bluetooth Stack (SP2)
USB Bluetooth Dongle

Figure 8 Adapted from Basta et al., 2006

Experimental Results:

Performance evaluation of the proposed client-server architecture on following hardware and software was done:

Client Mobile device- Sony Ericsson w550i.

Server – Dell Inspiron 1520 Intel Pentium core 2 duo, 2.1 Ghz, 1 Gb RAM, operating system Windows Vista, Dell TrueMobile 355 Bluetooth + EDR

Table shows the average time intervals in ms:

- T1- Image capture time interval
- T2- Face detection time interval
- T3- Connection establishment with the server
- T4- Image Bytes transfer from client mobile device to Server using Bluetooth.
- T5- XML file writing time interval.
- T6- Face recognition time interval.
- T7- Response sent to the client and display on UI.

Interval, ms	Bluetooth
T1	34
T2	19142
T3	566
T4	320
T5	85
T6	18451
T7	642

The results Client Server architecture using Bluetooth is compared with the Client Server architecture using HTTP [4]. Various advantages as well as disadvantages were found out[2]:

- In CSAB(Client Server Architecture using Bluetooth) communication is between client and server without any dependency on third party where as in CSAH (Client Server Architecture using HTTP) the communication between the client and server depends on the availability as well as reliability of the GPRS service provider.
- In CSAB(Client Server Architecture using Bluetooth) the image's bytes transfer time is less than in CSAH (Client Server Architecture using HTTP) and depends mainly on the distance between the distance and obstacles between client and server and range of Bluetooth devices where as in latter case the transfer time mainly depends on the service provider.
- In CSAB the errors/malfunction areas of communication can be easily detected where as in CSAH it again depends on the availability of the service provider.
- CSAB is free but CSAH has price according to the service provider.
- CSAH has higher mobility than CSAB as the former depends on the availability of GPRS and latter depends on the availability of Bluetooth devices at the client and server side as well as on range of those Bluetooth devices. The clients always have to be in Bluetooth range in case of the CSAB.

Conclusion and Future Work

By seeing the results various conclusions can be seen, it can be seen both CSAH (Client Server Architecture using HTTP) and CSAB (Client Server Architecture using Bluetooth) have advantages as well as disadvantages, as can be seen from the results that CSAH and CSAB are complementary to each other so the best architecture would be a combination of these two architecture which provide a great deal of extensibility. For dedicated face detection and recognition areas CSAB should be used and for general purpose CSAH should be used.

For the future, both CSAB and CSAH can be combined together in an application for increased performance as well as availability.

The face recognition algorithm is tested on the AT&T face database (Our database of Faces) from AT&T Laboratories, Cambridge and results are taken according to it. New face recognition algorithms can be incorporated in the application for better performance.

REFERENCES

[1] Java Bluetooth Discussions and Tutorials. (2006). Retrieved July 14, 2007 from http://www.jsr82.com.
[2] R.S. Ivanov, "An application for Image Database Development for Mobile Face Detection and Recognition Systems"
[3] M. Turk, A. Pentland. "Eigenfaces for Recognition". Journal of Cognitive Neuroscience. Vol 3, No. 1. 71-86, 1991.
[4] Paul Viola, Michael J. Jones, Robust Real-Time Face Detection, International Journal of Computer Vision 57(2), 2004.
[5] http://jcp.org/en/jsr/detail?id=118
[6] http://jcp.org/en/jsr/detail?id=135
[7] http://www.gartner.com/it/page.jsp?id=498310
[8] W. Zhao, R. Chellappa, A. Rosenfeld, P.J. Phillips, Face Recognition: A Literature Survey, ACM Computing Surveys, 2003, pp. 399-458
[9] L. Torres, Is there any hope for face recognition?, Proc. of the 5th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2004, 21-23 April 2004, Lisboa, Portugal
[10] M. Turk, A Random Walk through Eigenspace, IEICE Transactions on Information and Systems, Vol. E84-D, No. 12, December 2001, pp. 1586-1595
[11] W.S. Yambor, Analysis of PCA-based and Fisher Discriminant-Based Image Recognition Algorithms, M.S. Thesis, Technical Report CS-00-103, Computer Science Department, Colorado State University, July 2000
[12] W.Y. Zhao, R. Chellappa, Image-based Face Recognition: Issues and Methods, Image Recognition and Classification , Ed. B. Javidi, M. Dekker, 2002, pp. 375-402
[13] M.-H. Yang, D.J. Kriegman, N. Ahuja, Detecting Faces in Images: A Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 1, January 2002, pp. 34-58
[14] P.J. Phillips, H. Moon, S.A. Rizvi, P.J. Rauss, The FERET Evaluation Methodology for Face-Recognition Algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 10, October 2000, pp. 1090-1104
[15] J.W. Tanaka, M.J. Farah, Parts and Wholes in Face Recognition, Quarterly Journal of Experimental Psychology Section A: Human Experimental Psychology, Vol. 46, No. 2, 1993, pp. 225-245