Towards Setting up of Software Experimental Laboratory for Image Processing Applications

Shivalika Aggarwal¹, Poonam Sharma² and A. B. Patki³
¹²HMR Institute of Technology & Management, Delhi
³Department of Information Technology MCIT, New Delhi
¹shivalika.aggarwal@gmail.com, ²poonam2_3@yahoo.co.in and ³patki@nic.in

ABSTRACT
The laboratory setup for image processing applications is typical in educational institutions. Image processing is the signal processing where the input is an image. Biometric is the field which involves an extreme use of image processing. Face Recognition is one of the biometric applications that use the principle of image processing. To demonstrate the image processing laboratory setup for Java language based environment to work seamlessly for a series of mathematical computations in milliseconds of time, we are implementing eigenfaces algorithm in face recognition. The approach of using Eigenfaces was developed by Sirovich and Kirby and used by Matthew Turk and Alex Pentland in face classification. A set of eigenfaces is generated using mathematical tool Principal Component Analysis (PCA). The Java development kit, Netbeans, jar libraries, java media framework comprises the efficient environment for image processing. [1,2]

KEYWORDS
Image Processing, Eigenfaces, Principal Component Analysis (PCA), Java, Face Recognition

1.INTRODUCTION
Setting up of a software experimental laboratory in engineering colleges has been always challenging. Typically at undergraduate level, an approach of several small tiny program coding is the goal. While this approach develops on small-step concept towards an applications development, in practice, integration is never done which results in a situation that students always find it difficult to appreciate the end product. We describe a methodology to setup Java based laboratory for advanced applications of image processing techniques, especially in the area of security systems. Such an Image Processing application is Face Recognition. Face recognition has received considerable interest as a widely accepted biometric, because of the ease of collecting samples of a person, with or without subject’s intensity. Our design of the face recognition system COMET is based upon “eigenfaces approach” and has been separated into three major modules – face detection, face normalization with eigen value computation and face recognition [3]. Face detection and normalization are performed in real-time; consistent accuracy in face detection is recorded with video input. Under dynamic mode, where recognition is accomplished from real-time video input, the system has been demonstrated to perform at a maximum speed of two frames per second under Java. This project's emphasis is on the data aspect i.e. the algorithm or technique used in recognising face images [4]. In order to achieve these purposes, the following tasks are identified:

(i) To analyse Eigenface technique on face recognition particularly on improving the accuracy and efficiency of such systems.
(ii) To develop face recognition system, in order to investigate the affect, the accuracy and efficiency of a face recognition system in our Java based Laboratory.

Since, our emphasis is on experimental laboratory activity, we focus in the paper on test plan and performance measurement issues as discussed in the following sections.

2.MODELLING APPROACH
Eigenfaces approach is chosen because it is very straightforward to implement. The Eigenfaces approach is one of the algorithms used in 2-dimensional face recognition. The principal components of the distribution of faces are calculated (also called the eigenvector, which is the covariance matrix of the set of face images) from the test image & training dataset. These eigenvectors represent a set of features, which together is characterized as the variation between the face images. Therefore, the eigenvector can be displayed as a sort of ghostly face, known as eigenface, because each image location contributes more or less to each eigenvector. The eigenfaces forms the axis of a n-dimensional coordinate system called face space. When recognition, a PCA tool (principal Component Analysis) is used to identify face from this face space. The classifying technique used here is minimum distance [1]. Schematic diagram shown in Figure 1 illustrates the basic mechanism.

Copy Right © INDIACom-2011 ISSN 0973-7529 ISBN 978-93-80544-00-7
The Hardware interfaces include digital/web camera, hard disc and secondary storage. The Hardware interfaces include digital/web camera 1.3 megapixel, hard disc 1-3 GB, RAM 64 MB and secondary storage. The software interfaces is configured around Java Media Framework (JMF), Library Jars (Colt, Jai_Codec, Jai_core), NetBeansIDE 6.9.1 and Java Swings APIs. A use case diagram in the Unified Modelling Language (UML) facilitates to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. Figure 2 indicates our methodology.

**Figure 1: Basic approach for Eigen Value Face Analysis**

The pre-processing is performed to eliminate differences among the face images. Library jars (jai_core.jar, jai_codec.jar, colt.jar) are used for image pre-processing & matrix operations in the algorithm [9]. The images are pre-processed to have the following attributes:

a) Each image is stored as 24-bit RGB color jpeg converted in gray scale during the algorithm.

b) Each image has the same size, which is a width and height of 264 and 316 pixels respectively.

c) Only the face image is taken whereby all images are scaled without any background image.

d) For each person, images are taken with expressions like neutral, smiling, speech, etc. in frontal pose.

Pre-processing is necessary to ensure that all images are converted from 2D to 1D vector & of equal dimensions. Rasterization is done for 2D to 1D vector formation. After these images have been pre-processed with similar attributes, the Flowchart: Eigenface Algorithm.

**Figure 2: Use Case Diagram**

**3.SOFTWARE IMPLEMENTATION**

The system is built using core Java as NetBeans IDE 6.9.1 for programming with the incorporation of File System for implementing the dataset while the Eigenface algorithm was used in implementing the recognition and identification process. NetBeans requires a compatible JDK (1.5 or above) for its installation. The system requires JMF (Java Media Framework) 2.1.1e API for incorporating media data (video) to enable webcam for on-time clicking of test image [10]. The test image can also be provided by uploading via a file chooser GUI created in java swings [11].

The pre-processing is necessary to ensure that all images are converted from 2D to 1D vector & of equal dimensions. Rasterization is done for 2D to 1D vector formation. After these images have been pre-processed with similar attributes, the Flowchart: Eigenface Algorithm.

**Figure 3(a): Learning Stage**
Figure 3(b): Recognition Stage

They can be stored in the file system. The face images in the dataset are not classified according to any factor like gender, race, etc. As mentioned above, file system was used to store the dataset & test image.

The implementation of the dataset is as follows:

a) Each on-time clicked image is stored in "img" folder in the system.

b) The Dataset of images is maintained as "testDB" within the system.

c) The frontal view face image is taken as the main picture for the recognition process.

The algorithm is implemented as two stages: learning stage & recognition stage. The flow chart of the algorithm is shown in Figure 3(a) & 3(b).

To enhance the efficiency, in learning stage, cache file is created that incorporates all the eigen vectors, eigen values & weight vectors of images in the dataset. This cache file is used to create eigenfaces stored as eigen folder in the system which forms the training set face space with each prominent feature set highlighted [5].

4. EXPERIMENTAL SETUP:

4.1 Eigenfaces Performance Measurement

a) Processing time
The processing time deals with how long the system takes to recognize and identify a given test image. It is in line with the duration it takes for the system to collect the face images for the training set until the system displays the recognised or identified face. This calculation is performed by the system automatically once the user starts the recognition process.

b) Accuracy of the identified face
The accuracy of the identified face deals with the correctness of the identified face (as a result of the face recognition process) compared to the given test image. It is performed by the user by checking manually whether the presented face is the same as the test image.

However, the experiment has been done for a number of individuals, but here we are explaining only one person's cluster. The Experiment done for an individual’s various types of images, for example, smiling face, neutral expressions, bright light or little hazy ambient light is tabulated in Table 1.

The readings of Table 1 have been plotted in 3D and 2D with parameters selected number of Eigenface (X-axis), Matching Distance (Y-axis), Time (Z-axis) and selected number of eigenface (X-axis), Matching Distance (Y-axis), respectively.

**Table 1: An individual’s cluster table**

```
<table>
<thead>
<tr>
<th>No</th>
<th>Selected Number of Eigenfaces (X-axis)</th>
<th>Matching Distance (Y-axis)</th>
<th>Matched/Non-Matched</th>
<th>Time (milliseconds) (Z-axis)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Same as above</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>Same as above</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>Same as above</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0</td>
<td>Same as above</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>0.138921463504099</td>
<td>Non-Matched</td>
<td>6458</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>0.3784441505644464</td>
<td>Matched</td>
<td>789</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0.263912949890045464</td>
<td>Matched</td>
<td>10390</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>0.8107416490818567</td>
<td>Non-Matched</td>
<td>10923</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>0.9578497935733497</td>
<td>Matched</td>
<td>11325</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>0.4226158506961897</td>
<td>Matched</td>
<td>12867</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>1.5155232440495842</td>
<td>Non-Matched</td>
<td>16271</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>0.158444506042133</td>
<td>Matched</td>
<td>17319</td>
</tr>
</tbody>
</table>
```

**Graph 1: 2D plot of selected number of eigenfaces (X-axis) and Matching Distance (Y-axis)**

**Graph 2: 3D plot of selected number of eigenfaces (X-axis) and Matching Distance (Y-axis), Time (Z-axis)**

Explanation of Graph 1:

a) The first four bold blue dots on X-axis shows that the selected number of Eigenface is not valid.

b) The next two bold blue dots at a significant height from the X-axis shows that the selected number of eigenfaces is valid as well as the face is detected with the known features.
c) The next four bold blue dots which are closely located to the X-axis denote that the face is detected but the feature sets are not matched.

Explanation of Graph 2:

a) The 3D plot is same as the 2D plot and shows that with increasing number of selected eigenfaces the matching time is also increased.

FUTURE SCOPE & CONCLUSION

In this paper we have discussed face recognition’s eigenface algorithm using PCA as an application of image processing to set-up the java based software laboratory for institutional courses. We describe environment using Microsoft Windows platform since it is popular and accessible in many educational institutions. We are in the process of experimenting under Linux environment also. The paper brings out the integrated approach towards instead of merely a collection of experiments typically followed in data structure and algorithms software laboratory in computer science curriculum. It is expected that this route of educating students helps to strengthen the concepts needed in the industry towards a total solution in place of the fragmented assorted laboratory exercise. We believe that our laboratory approach using COMET will open up teaching and adaptation of specialized courses in MCA/Polytechnic level engineering institutions.

In future, this application can be hybrid with other classification techniques like neural network for better accuracy and lifting the level of laboratory higher as more complex and powerful mathematical computation will be done to achieve the goal. Also, it should be noted that the ideas of unsupervised clustering in distance classification technique are not currently implemented on the system on which experiments are being carried out, but rather are an area for future work [3]. Fuzzy logic based face identification is another technique which is being investigated by us.

REFERENCES


