A Face Detection and Recognition System for Color Images using Neural Networks with Boosting and Deep Learning

By

Mohammadreza Hajiarbabi

Submitted to the graduate degree program in Electrical Engineering and Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

_____

Chairperson Arvin Agah

_____

Prasad Kulkarni

_____

Bo Luo

_____

Guanghui (Richard) Wang

_____

Sara Wilson*

Date Defended: 17 November 2017

The Dissertation Committee for Mohammadreza Hajiarbabi
certifies that this is the approved version of the following dissertation:

A Face Detection and Recognition System for Color Images using Neural Networks with Boosting and Deep Learning

_____

Chairperson Arvin Agah

Date approved: 17 November 2017

Abstract—A face detection and recognition system is a biometric identification mechanism which compared to other methods such as finger print identification, speech, signature, hand written and iris recognition, is shown to be more important both theoretically and practically. In principle, the biometric identification methods use a wide range of techniques such as machine learning, computer vision, image processing, pattern recognition and neural networks. The methods have various applications such as in photo and film processing, control access networks, etc. In recent years, the automatic recognition of a human face has become an important problem in pattern recognition. The main reasons are structural similarity of human faces and great impact of illumination conditions, facial expression and face orientation. Face recognition is considered one of the most challenging problems in pattern recognition. A face recognition system consists of two main components, face detection and recognition.

In this dissertation a face detection and recognition system using color images with multiple faces is designed, implemented, and evaluated. In color images, the information of skin color is used in order to distinguish between the skin pixels and non-skin pixels, dividing the image into several components. Neural networks and deep learning methods has been used in order to detect skin pixels in the image. A skin database has been built that contains skin pixels from different human skin colors. Information from different color spaces has been used and applied to neural networks. In order to improve system performance, bootstrapping and parallel neural networks with voting have been used. Deep learning has been used as another method for skin detection and compared to other methods. Experiments have shown that in the case of skin detection, deep learning and neural networks methods produce better results in terms of precision and recall compared to the other methods in this field.

The step after skin detection is to decide which of these components belong to human face. A template based method has been modified in order to detect the faces. The template is rotated and rescaled to match the component and then the correlation between the template and component is calculated, to determine if the component belongs to a human face. The designed algorithm also succeeds if there are more than one face in the component. A rule based method has been designed in order to detect the eyes and lips in the detected components. After detecting the location of eyes and lips in the component, the face can be detected.

After face detection, the faces which were detected in the previous step are to be recognized. Appearance based methods used in this work are one of the most important methods in face recognition due to the robustness of the algorithms to head rotation in the images, noise, low quality images, and other challenges. Different appearance based methods have been designed, implemented and tested. Canonical correlation analysis has been used in order to increase the recognition rate.

Index Terms—Biometric System, Color Images, Skin Detection, Face Detection, Face Recognition, Lip Detection, Eye Detection, Artificial Neural Networks, Boosting Techniques, Appearance-based Methods, Segmentation Methods, Deep Learning, Canonical Correlation Analysis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Recognizing the identity of humans has been important for so many years. Humans recognize each other based on physical characteristic such as face, voice, gait and etc. In the past centuries, the first systematic methods for recognizing were invented and used in police stations for recognizing the villains. These methods measured the different parts of the body. After finding that the finger print is unique for each person, this method became the best way for recognizing humans. In the last decades and because of inventing high-speed computers, a good opportunity has been provided for the researches to work on different methods and to find certain methods for recognizing humans based on unique patterns.

The most common biometric systems, which have been used during these years, are fingerprint, voice, iris, retina, hand written and face, etc. Each of these recognition systems has positive and

negative characteristics. A biometric system is a system, which has an automated measuring part that is robust and can distinguish physical characteristics that can be used to identify a person. By robust we mean that the features should not change significantly with the passing of years. For example, iris recognition is more robust than other biometric systems because it does not change a lot.

Each biometric system has three tasks to perform:

1) Verification / Authentication: Is this person the same as said? In such systems, the system input from the user is needed which could be a password. The system also needs a biometric sample from the user. Using the password and the biometric sample the system can authenticate the person.
2) Identification / Recognition: In this case it is assured that a person is presented in the database and there is a need for recognition.
3) Watch list: In this case there is a need to see whether the image of the person exists in the database and if the answer is yes, who is the person.

Common biometric systems include:

- Iris recognition: The iris patterns are different between humans. Even for each person the patterns of right and left eyes are different and for the twins it differs from one another. The iris will not change during life. High cost and intrusiveness are the major problems of iris recognition systems.

- Retina recognition: In this method the recognition is done by using the patterns from the blood vessels behind the eye. Retina recognition has high accuracy but has the issues of high cost, not being everywhere, and being painful.
- Face recognition: This system uses the features from the human face for recognizing the person. Although its accuracy is less than iris and retina recognitions, it is much easier to use in different places such as Malls, markets and etc.
- Voice and speech recognition: This system uses the voice signals in order to recognize the person. It is not expensive, but the environment's noise can affect it.
- Fingerprint and palm print recognition: These methods use the information that exists on the ridges of the finger and palm of the hand. Finger print recognition is now used in many places. Its recognition rate is close to face recognition systems.
- Others: There are numerous of other methods that can be used for recognition such as gait recognition, odor recognition and etc.

It has been shown that by considering four features of biometric systems, i.e., intrusiveness, cost, easy percept and high performance, face recognition seems to be the best biometric system (Hietmeyer, 2000).

Considering the facts about face recognition, it is not surprising that face recognition has become one of the most active areas of research. The two major issues that a face recognition system has are:

- The images of a person can be very different, and the differences can be due to some changes in face such as facial hair or makeup and that faces change over time.

- Illumination, background and scale differences between images.

A face recognition system is a pattern recognition system. A pattern recognition system has two main components, which are feature extraction and classifier (Webb, 2002). In feature extraction, the most important information is extracted from the original information.

A face recognition system has three main components:

1. Face detection or face localization
2. Feature extraction
3. Classification

In face detection and localization, the main aim is to find the location of the face in the image and to discard the background. In feature extraction, the most important features are extracted from the face, and in classification each person is recognized. Therefore, there is a need for a database consisting of a number of people, images and for each person there must be some training images. The features extracted from each image are compared to the features of the saved images (training images) and they belong to a class (person) whose features are the most similar to the features of the questioned image.

In this dissertation we focus on color images which can include multiple faces. Our first step is to detect the human skin in the color images. This task is crucial because other tasks rely on skin detection. We will introduce a new method for skin detection using neural networks and deep learning. Next step would be the face detection. We have enhanced an algorithm in this field in

order to improve the results. This work can handle faces that are close to each other and also for other problems such as how to detect faces when the background color is similar to human skin color. In addition, a method for eyes and lip detection using color images have been proposed, face can be detected by using the information of the location of eyes and lip. The last step would be face recognition.

This dissertation is organized into right chapters. In chapters 2, 3, and 4 we will discuss the background and related work in skin detection, face detection and face recognition. In chapter 5, our approach to skin detection will be presented. Chapter 6 explains the face detection methodology and includes a discussion on the improvements to the algorithm. Chapter 7 presents the face recognition algorithms. Chapter 8 will discuss the conclusion, limitations and the future work.

Chapter 2

Background and Related Work – Skin Detection

In this chapter we will discuss the methods and algorithms that exist in the area of skin detection. As mentioned before, an active area of research in image processing is face recognition, i.e., the ability to recognize a face and identify the person. Face recognition relies on detecting the face in the image, and the first phase of face detection in color images is segmentation of the image into human skin and non-human skin. Skin color is an indication of race, health, age, etc. (Fink *et. al.*, 2006). In video images, skin color shows the existence of humans in media; therefore, in recent years significant research efforts have been focused on skin detection in images (Elgammal *et. al.*, 2009). Skin detection has numerous applications, such as identifying and filtering nude pictures on the internet (Fleck *et. al.*, 1996), detecting anchors in TV news videos (Abdel-Mottaleb and Elgammal, 1999), face detection, etc. This segmentation is challenging due to the differences of illumination between images, images being taken by different cameras, different characteristics of various lenses, and the ranges of human skin colors due to ethnicity. One of the most important issues is that there are pixels common between human skin and other entities such as soil, desk surfaces, walls, and other common items (Alshehri, 2012). These challenges

make image processing of skin color detection very difficult, and there is no exact mechanism to distinguish between skin color and non-skin color.

There are a number of color spaces that can be used for skin detection. The most common color spaces are RGB, YCbCr, and HSV.

The RGB color space consists of red, green and blue colors from which other colors can be generated. The RGB color space can be shown by a 3-dimentional cube with Red, Green, and Blue (the three base colors), or Cyan, Magenta, and Yellow (the secondary colors). The black and white are the corners of this cube. Black is placed in the center of the Cartesian coordinate system. Although this model is simple, it is not suitable for all applications (Singh *et al.*, 2003).

In the YCbCr color space, Y is the illumination (Luma component), and Cb and Cr are the chroma components. In skin detection, the Y component can be discarded because illumination can affect the skin. YCbCr belongs to color spaces that are used for television color spaces. YUV and YIQ are two other color spaces used in television. YCbCr is used in digital systems and YUV and YIQ are used in analog systems. The equations for converting RGB to YCbCr are as follows:

$$Y = 0.257R + 0.504G + 0.098B + 16$$

$$Cb = -0.148R - 0.291G + 0.439B + 128$$

$$Cr = 0.439R - 0.368G - 0.071B + 128$$

The Y component ranges are from 16 to 235 so it has 220 different levels, Cb and Cr ranges are from 16 to 240, so have 225 levels.

The HSV color space has three components, namely, H (Hue), S (Saturation), and V (Value). Because V specifies the brightness information, it can be eliminated in skin detection. In the HSV color space, every color can be represented by just adjusting the Hue. With Saturation, we can change a color to another color that is similar to it—for instance, from dark red to pink. By adjusting the value, one can produce darker and lighter color. The equations for converting RGB to HSV are as follows:

$$R' = \frac{R}{255}, G' = \frac{G}{255}, B' = \frac{B}{255}$$

$$Cmax = max(R', G', B'), Cmin = min(R', G', B'), x = Cmax - Cmin$$

$$H = 60 * \left(\frac{G' - B'}{x} \ mod 6\right), Cmax = R'$$

$$H = 60 * \left(\frac{B' - R'}{x} + 2\right), Cmax = G'$$

$$H = 60 * \left(\frac{R' - G'}{x} + 4\right), Cmax = B'$$

$$S = \frac{x}{Cmax} \ for\ x\ not\ being\ zero\ other\ wise\ S = 0$$

$$V = Cmax$$

The three methods of Gaussian, rule-based, and neural networks are discussed in this section.

2.1 Gaussian Method

This technique which was first proposed in (Wu and Ai, 2008), uses the Gaussian model in order to find the human skin in an image. The RGB image is transformed to the YCbCr color space. The density function for Gaussian variable $X = (Cb\ \ Cr)^T \in R^2$ is:

$$f(Cb\ \ Cr) = \frac{1}{2\pi|C|^{1/2}} exp\{-\frac{1}{2}(X-\mu)^T C^{-1}(X-\mu)\}$$

Where $X = \begin{pmatrix} Cb \\ Cr \end{pmatrix}$, $\mu = \begin{pmatrix} \mu_{Cb} \\ \mu_{Cr} \end{pmatrix}$, $C = \begin{pmatrix} C_{CbCb} & C_{CbCr} \\ C_{CrCb} & C_{CrCr} \end{pmatrix}$, and the parameters are:

$$\mu_{skin} = \begin{pmatrix} 112.1987 \\ 151.3993 \end{pmatrix} \quad , \quad C_{skin} = \begin{pmatrix} 89.3255 & 32.2867 \\ 32.2867 & 252.9336 \end{pmatrix}$$

The parameters are calculated using training images. For each pixel value, the density function is calculated. Just the (Cb Cr) value is used because the Y component has the illumination information, which is not related to skin color. The probability value of more than a specified threshold is considered as skin. The final output is a binary image where the non-skin pixels are shown by black and human skin by white. It should be mentioned that the value of parameters μ and C are calculated using specified training samples and can be varied by using other training samples.

2.2 Rule-Based Methods

Skin detection based on rule-based methods has been used in several research efforts as the first step in face detection. Chen *et al.* analyzed the statistics of different colors (Chen *et al.*, 2008). They used 100 images for training, consisting of skin and non-skin in order to calculate the conditional probability density function of skin and non-skin colors.

After applying Bayesian classification, they determined the rules and constraints for the human skin color segmentation. The rules are:

$$r(i) > \alpha \ , \quad \beta_1 < \big(r(i) - g(i)\big) < \beta_2 \ , \ \gamma_1 < \big(r(i) - b(i)\big) < \gamma_2$$

$$\sigma_1 < \big(g(i) - b(i)\big) < \sigma_2$$

With $\alpha = 100$, $\beta_1 = 10$, $\beta_2 = 70$, $\gamma_1 = 24$, $\gamma_2 = 112$, $\sigma_1 = 0$ and $\sigma_2 = 70$

Although this method worked on some images perfectly, the results are not reliable on images with complex background or uneven illumination.

Kovac *et al.* (2003) introduced these rules for skin segmentation in RGB space, where each pixel that belongs to human skin must satisfy these relations:

For indoor images:

$$R > 95 \ , \quad G > 40 \ , \quad B > 20 \ , \quad max\{R,G,B\} - min\{R,G,B\} > 15 \ ,$$

$$|R - G| > 15 \ , \ R > G \ , \ R > B$$

For images taken in daylight illumination:

$$R > 220 \ , \quad G > 210 \ , \quad B > 170 \ , \quad |R - G| \leq 15 \ , \ R > B \ , \ G > B$$

Kong and Zhe (2007) presented rules that used the information from both HSV and normalized RGB color spaces. They suggested that although in normalized RGB the effect of intensity has been reduced, it is still sensitive to illumination. Therefore, they also used HSV for skin detection. Each pixel that satisfies these rules is considered to be a human skin pixel:

$$0.4 \le r \le 0.6 \quad , \quad 0.22 \le g \le 0.33 \ , \ r > g > \frac{1-r}{2} \quad ,$$

$$0 \le H \le 0.2 \quad , \ 0.3 \le S \le 0.7 \quad , \ 0.22 \le V \le 0.8$$

$$r = {}^{R}/_{(R+G+B)} \ , \qquad g = {}^{G}/_{(R+G+B)} \ , \qquad b = {}^{B}/_{(R+G+B)}$$

Some researchers belive the normalized RGB is more effective than RGB because it does not depend on illumination.

Gomez and Morales (2002) introduced these equations for finding human skin pixel:

$$\frac{r}{g} > 1.185 \ , \qquad \frac{r.b}{(r+g+b)^2} > 0.107 \ , \qquad \frac{r.g}{(r+g+b)^2} > 0.112$$

Chai and Ngan (1998) proposed these equations using YCbCr color space:

$$77 \le Cb \le 127 \ , \qquad 133 \le Cr \le 173$$

Some reserchers do not use Y, because they belive that it contains illumination information and so using it does not improve the skin detection quality.

Kukharef and Novosielsky (2004) used these equations:

$$Y > 80 \ , \qquad 85 \le Cb \le 135 \ , \qquad 135 \le Cr \le 180$$

Berbar *et al.* (2006) introduced these equations:

$$85 \le \mathrm{Cb} \le 127 \ , \qquad 137 \le \mathrm{Cr} \le 177 \ , \qquad 183 < (Cb + 0.6Cr) < 219$$

2.3 Neural Network Methods

Neural network has been used in skin color detection in a number of research projects. Doukim *et al.* used YCbCr as the color space with a Multi-Layer Perceptron (MLP) neural network (Doukim *et al.*, 2010). They used two types of combining strategies, and several combining rules were applied. A coarse to fine search method was used to find the number of neurons in the hidden layer. The combination of Cb/Cr and Cr features produced the best result. Brancati *et al.* (2016) introduced a method based on correlation rules between different color spaces.

Seow *et al.* used the RGB as the color space which was used with a three-layered neural network. Then the skin regions were extracted from the planes and were interpolated in order to get an optimum decision boundary and the positive skin samples for the skin classifier (Seow *et al.*, 2003).

Yang *et al.* (2010) used YCbCr color space with a back propagation neural network. They took the luminance Y and sorted it in ascending order, dividing the range of Y values into some intervals. Then the pixels whose luminance belonged to the same luminance interval were collected. In the next step, the covariance and the mean of Cb and Cr were calculated and were used to train the back propagation neural network. Another example of methods of human skin color detection using neural network can be found in (Al-Mohair *et al.*, 2012). Zuo *et al.* (2017) used convolutional and recurrent neural networks for human skin detection.

Chapter 3

Background and Related Work – Face Detection

The next step of a face recognition system is face detection. After the skin detection step, the next step is to use a face detection algorithm to detect the faces in the image. Several methods have been developed for face detection. In this chapter, we present the common methods which have been used in this field. Rowley and Viola methods are the most famous methods in this field (Rowley *et. al.*, 1998) (Viola *et al.*, 2001).

3.1 Rowley Method

Rowley used neural network and the method detected upright frontal faces in gray-scale images. One or more neural networks are applied to portions of an image and absence or presence of a face is decided. They used a bootstrap method for the training, which means that they add the images to the training set as the training progresses. Their system had two stages. First, a set of

neural network-based filters were applied to an image. These filters looked for faces in the image at different scales. The filter is 20*20 pixel region of the image and the output of the image is 1 or -1 which 1 shows that the region belongs to face and -1 shows that the region contains no face. This filter moves pixel by pixel through the whole image. To solve the problem for faces bigger than this size, the image was subsampled by a factor of 1.2.

They used a preprocessing method from (Sung, 1996). In the preprocessing step, first the intensity values across the window are equalized. A linear function was fitted to the intensity values in the window and then it was subtracted, which corrected some extreme lightening conditions. Then histogram equalization was applied, in order to correct the camera gain and also to improve the contrast. In addition, an oval mask was used to ignore the background pixels.

The window then was passed to a neural network. Three types of hidden units were used. Four units which looked at 10*10 sub regions, 16 which looked at 5*5 sub regions, and 6 which looked at overlapping 20*5 horizontal stripes. These regions were chosen in order to detect different local features of the face. For example, the horizontal stripes were chosen to detect eyes, eyebrows, and etc.

Approximately 1050 face images were used for training. Images were chosen from the Web and some popular databases. As mentioned before, the images were black and white. For the non-face images another method was used. The reason is that, face detection is quite different from other problems. The set of non-face images is much bigger than face images. Their method was:

- An initial set consisting of 1000 random images were created. The preprocessing method was applied on these images.
- A network was trained using the face and non-face images.
- The network was tested on an image that contained no face. The misclassified sub images were chosen. Those which were considered as faces wrongly.
- 250 of these sub images were chosen randomly, the preprocessing methods were applied on them and these sub images were added into the training set as negative examples. The process was continued from second step.

Other new ideas were used by Rowley. In the areas which contain face, there are lots of detection because of the moving pixel by pixel of the window over the image and also several detections because of using different scales over the image. They used a threshold and counted the number of the detections in each location. If the number of detections were above a specified number, then that location was considered as a face, otherwise rejected. Other nearby detections that overlapped the location classified as face was considered as error, because two faces cannot overlap. This method was called overlap elimination.

Another method that was used to reduce the number of false positives was using several networks and an arbitration method between them to produce the final decision. Each network was trained with different initial weights, different random sets of non-face images, and different permutations of the images that were presented to the network. Although the networks had very close detection and error rates, the errors were different from each other. They used combination of the networks using AND, OR and voting method.

3.2 Viola Method

Viola *et al.* (2001) trained a classifier using a large number of features. A set of large weak classifiers was used, and these weak classifiers implemented a threshold function on the features. They used three kinds of Haar features. In two-rectangle feature the value is the subtraction between the sum of the pixels within two regions. In three-rectangle feature the value is the sum within two outside rectangles subtracted from the inside rectangle. In four-rectangle feature the value is the difference between diagonals of the rectangle.

One of important things was how to compute the features very fast. Viola *et al.* used the integral image that allowed the feature to be computed very fast. The value of an integral image at point (x,y) is the sum of all pixels that are above and left of the (x,y) pixel. They showed that the integral image can be computed in one pass (Viola *et al.*, 2001).

Their AdaBoost learning algorithm is as follows (Viola *et al.*, 2001):

- The first step is initializing the weights

$$w_{1,i} = \frac{1}{2m} \quad , \quad \frac{1}{2l} \quad for\ y_i = 0,1$$

  Where m is the number of positive examples and l is the number of the negative examples. $y_i = 0$ for negative examples and $y_i = 1$ for positive examples.

- For t=1,…,T

- Normalize the weights using

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

Now the value of the weights will be between 0 and 1 and so is a probability distribution.

- For each feature j a classifier h_j is trained and uses just a single feature. The error is:

$$\epsilon_j = \sum_i \omega_i \left| h_j(x_i) - y_i \right|$$

- The classifier h_t with the lowest error ε_t is chosen and the weights are updated using

$$\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}$$

If example x_i is classified correctly then $e_i = 0$, otherwise $e_i = 1$ and

$$\beta_t = \frac{\epsilon_t}{1 - \varepsilon_t}$$

- The final classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases} \qquad and \qquad \alpha_t = \log \frac{1}{\beta_t}$$

Viola *et al.* (2001) had another contribution which was constructing a cascade of classifiers which was designed to reduce the time of finding faces in an image. The beginning cascades reject most of the images, images which pass the first cascade will go to the second one and this process continues till to the last cascade of classifiers.

Like Rowley method, in this method a window is moving on the image and deciding if that window contains a face. However, Viola showed that their method is faster than Rowley method (Viola *et al.*, 2001).

3.3 Other Methods

There are also some other methods which have been used for face detection. Principal component analysis (PCA) method which generates Eigen face has been used in some work for detecting faces (Bakry *et al.*, 2006). Other types of neural networks have also been used (Yang K. *et al.*, 2006). Support Vector Machines (SVM) has been used for face detection (Shavers *et al.*, 2006). Geometrical facial features for face detection have been studied (Jeng *et al.*, 1996). They showed that their method works for detecting faces in different poses. Hjelmas has a survey in face detection methods from single edge based algorithms to high level approaches (Hjelmas and Kee Low, 2001). Yang also has published another survey in face detection and numerous techniques have been described (Yang *et al.*, 2002).

3.4 Eyes and Lips Detection Methods

Another method for detecting faces is to detect eyes and lips in the image and then find the face by using the fact that eyes and lips are at a certain part of a face. A number of methods have been proposed and used for eyes and lip detection. In (Peng *et. al.*, 2005) an algorithm has been introduced which detects eyes on gray intensity face without spectacles. Their algorithm has two parts. First, a feature based method is used to find two rough locations of eyes, and second, the eyes are detected using template based method. In (Rajpathak *et. al.*, 2009) morphological and

color image processing have been used to detect eyes based on the fact that eye regions in an image are characterized by low illumination with high density edges. In (Sidbe *et. al.*, 2006) some constraints based on the anthropological characteristics of human eyes are used in order to detect eyes. In (Soetedjo, 2011) a three step method using eye localization, white color thresholding, and ellipse fitting has been used for eye detection. Information from Entropy has been used in (Hassaballah *et. al.*, 2011) and eyes and mouth detection using Viola-Jones method has been used in (Khan *et. al.*, 2013).

A variety of methods have been used for lip detection: (Chin *et. al.*, 2009) used watershed segmentation, (Kalbkhani and Amirani, 2012) used statistical methods based on local information, and in (Chiang *et. al.*, 2003) a simple quadratic polynomial model has been used for detecting skin and lips.

Chapter 4

Background and Related Work – Face Recognition

The third phase of a face biometric system is face recognition. Face recognition can be classified into three categories: model based, template based, and appearance based.

4.1 Model Based and Template Based

Model based and template based methods use a model and then place it on the test images, computing some parameters to recognize the person. Elastic bunch graph (Wiskott *et al.*, 1997), is a popular method in this field that uses a graph which is placed on the specified parts of face like eyes, noses, etc. Each vertex contains 40 Gabor wavelets and is called jet. These coefficients are used for recognition. This method needs frontal faces and so acts poorly on profile faces. Another problem is that these methods need images with high resolution.

Active Appearance Model (AAM) (Cootes *et al.*, 2001) and 3D morphable model are other model based methods (Blanz and Vetter, 2002) (Huang *et al.*, 2003).

Template based methods first find the location of each part of the face for example eyes, nose etc. and then by computing the correlation between parts of the training images and the test images the face can be recognized (Brunelli and Poggio, 1993).

As mentioned before, the model based and template based methods need images with high resolution and the images must be frontal images.

4.2 Appearance Based

Appearance based methods start with the concept of image space. A two dimensional image can be shown as a point or vector in a high dimensional space which is called image space. In this image space, each dimension is compatible with a pixel of an image. In general, an image with m rows and n columns shows a point in an N dimensional space where $N = m \times n$. For example, an image with 32 rows and 32 columns describes a point in a 1024 dimensional space. One important characteristic of image space is that changing the pixels of one image with each other does not change the image space. Also image space can show the connection between a set of images (Turk, 2001). As mentioned above, the image space is a space with high dimensions. The appearance based methods extracts the most important information from the image and lower the dimension of the image space. The produced subspace under this situation is called feature space or face space.

The origin of appearance based methods dated back to 1991 when Turk and Pentland introduced the Eigen face algorithm which is based on a famous mathematical method, Principal Component Analysis and this was the start of appearance based methods. In 2000, Scholkopf by introducing kernel principal component analysis (kernel Eigen face) expanded the concept of appearance based method into non-linear fields. Appearance based methods have been divided into two categories which is linear and non-linear methods. In the following sections these methods are presented.

4.2.1 Principal Component Analysis

Principal Component Analysis (PCA) is a method that is used to represent a collection of sample points efficiently, reducing the dimensionality of the space by projecting the points onto the principal axes, where an orthonormal set of axis, points in the direction of maximum covariance in the data. These vectors best account for the distribution of face images within the entire image space. PCA minimizes the mean squared projection error for a given number of dimensions, and provides a measure of importance (in terms of total projection error) for each axis (Turk and Pentland, 2001).

We now describe the PCA method (Turk and Pentland, 2001). Considering that Z_i is a two dimensional image with size $m \times m$. First we convert the matrix into a vector of size m^2. The training set of the n face can be written as (Turk and Pentland, 2001):

$$Z = (Z_1, Z_2, ..., Z_n) \subset \Re^{m^2 \times n}$$

Each of the face images belongs to one of the c classes. In face recognition, the total images that belong to one person are considered as one class. For the training images the covariance matrix can be computed by (Turk and Pentland, 2001):

$$\Gamma = \frac{1}{n}\sum_{i=1}^{n}\left(Z_i - \overline{Z}\right)\left(Z_i - \overline{Z}\right)^T = \Phi\Phi^T$$

Where $\Phi = \left(\Phi_1, \Phi_2, ..., \Phi_n\right) \subset \Re^{m^2 \times n}$ and $\overline{Z} = \left(1/n\right)\sum_{i=1}^{n} Z_i$ is the average of the training images in the database. The face images have been centered which means that the mean of the images is subtracted from each image.

After computing covariance matrix, the eigenvectors and eigenvalues of the covariance matrix are computed. Considering that $U = \left(U_1, U_2, ..., U_r\right) \subset \Re^{m^2 \times r}\left(r \prec n\right)$ be the eigenvectors of the covariance matrix, only small parts of this eigenvectors, for example r, that have the larger eigenvalues will be enough to reconstruct the image. So by having an initial set of face images $Z \subset \Re^{N^2 \times n}$ the feature vector corresponding to its related Eigen face $X \subset \Re^{r \times n}$ can be calculated by projecting Z in the Eigen face space by (Turk and Pentland, 2001):

$$X = U^T Z$$

4.2.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is used for projecting a set of training data. In face recognition these training data are the images which belong to a person. In face space which is a $m \times n$ dimension and m, n are the image dimensions, $X = \left(X_1, X_2, ..., X_n\right) \subset \Re^{m \times n}$ is a matrix containing the images in the training set. X_i is an image that has converted to a column vector.

In LDA two matrixes within class scatter matrix and between class scatter matrixes are defined. This method finds an optimal subspace in which the between class scatter matrix to the within class scatter matrix will be maximized (Fukunaga, 1990). The between class scatter matrix is computed by:

$$S_B = \sum_{i=1}^{c} n^i \left(\overline{X}^i - \overline{X}\right)\left(\overline{X}^i - \overline{X}\right)^T$$

Where $\overline{X} = \left(1/n\right)\sum_{j=1}^{n} X_j$ is the mean of the images in the training set and $\overline{X}^i = \left(1/n^i\right)\sum_{j=1}^{n^i} X_j^i$ is the mean of class i, and c is the number of the classes. In face recognition, each class is the total images that belong to one person. The between class scatter matrix defines the average scattering of one class across the average of the total classes. The within class scatter matrix is computed by

$$S_W = \sum_{i=1}^{c} \sum_{X_i \in n^i} \left(X_i - \overline{X}^i\right)\left(X_i - \overline{X}^i\right)^T$$

The within class scatter matrix defines the data of one class across the average of the class. The optimal subspace is computed by

$$E_{optimal} = \arg\max_E \frac{\left\|E^T S_B E\right\|}{\left\|E^T S_W E\right\|} = \left[c_1, c_2, ..., c_{c-1}\right]$$

Where $\left[c_1, c_2, ..., c_{c-1}\right]$ is the set of Eigen vectors of S_B and S_W correspond to $c-1$ greatest generalized Eigen value λ_i and $i = 1,2,...,c-1$

$$S_B E_i = \lambda_i S_W E_i \quad i = 1,2,...,c-1$$

$E_{optimal}$ is an optimal matrix which maximize the proportion of between class scatter matrix to the within class scatter matrix. This means that it maximizes the scattering of the data that belong to different classes and minimizes the scattering of the data belonging to the same class.

Thus, the most discriminant answer for face images X would be (Fukunaga, 1990):

$$P = E_{optimal}^{T} \cdot X$$

One of the drawbacks that LDA encounters in face recognition is the small sample size problem. In order to avoid singularity of the S_w matrix, the number of the training images must be much larger than the dimension of the subspace, a situation that rarely occurs in face recognition applications. In order to avoid the singularity problem first we have to reduce the dimension of the problem and then apply LDA. Principal component analysis (PCA) is the most popular method which has been used for dimension reduction. In addition to PCA, there are other effective methods that can be used as dimension reduction before LDA, such as Discrete Cosine Transform (DCT).

Some researchers have observed that applying PCA to reduce the dimension of the space can cause another problem which is eliminating some useful information from the null space. 2FLD has been introduced to avoid this problem and also the computational problem that applying PCA produces. But 2FLD algorithm produces other problems which have not been mentioned. The output of 2FLD method is a matrix and its dimension for an $m \times n$ image could be $n \times n$. This high dimension when we want to use a neural network for classification causes issues. A two and high dimension matrix cannot be applied to a neural network. If we change the matrix into a vector, we will have a vector with size n^2 and because of low sample test of each face the

network cannot be trained well. A direct LDA method that does not need to use PCA method before applying LDA has been proposed, but this method is time inefficient (Yu and Yang, 2001).

The problem of face recognition differs from other pattern recognition problems and so it needs different discriminant methods rather than LDA. In LDA the classification of each class is based on just one sample and that is the mean of each class. Because of shortage of samples in face recognition applications, it is better to use all the samples instead of the mean of each class for classification. Rather than minimizing within class distance while maximizing the between class distance, multiple exemplar discriminant analysis (MEDA) finds the projection directions along which the within class exemplar distance (i.e., the distances between exemplars belonging to the same class) is minimized while the between-class exemplar distance (i.e., the distances between exemplars belonging to different classes) is maximized (Zhou and Chellappa, 2003).

In MEDA the within class scatter matrix is computed by:

$$S_W = \sum_{i=1}^{C} \frac{1}{n^{i2}} \sum_{j=1}^{n^i} \sum_{k=1}^{n^i} \left(X_j^i - X_k^i\right)\left(X_j^i - X_k^i\right)^T$$

Where X_j^i is the $j\,th$ image of $i\,th$ class. By comparing it with the within class scatter matrix of LDA it can be seen that in this method all the images in a class have participated in making the within class scatter matrix instead of using just the mean of the class, as in LDA method. The between class scatter matrix is computed by:

$$S_B = \sum_{i=1}^{C} \sum_{j=1, j\neq i}^{C} \frac{1}{n^i n^j} \sum_{k=1}^{n^i} \sum_{l=1}^{n^j} \left(X_k^i - X_l^j\right)\left(X_k^i - X_l^j\right)^T$$

Dissimilarly to LDA in which the means of each class and means of all samples made the between class scatter matrix, in MEDA all the samples in one class are compared to all samples of the other class. The computation of $E_{optimal}$ is the same as LDA.

4.2.3 Fuzzy Linear Discriminant Analysis

A fuzzy model for linear discriminant analysis has been introduced (Kwak and Pedrycz, 2005). By having c class and N feature vectors and matrix $U = [\mu_{ij}]$ we will have:

$$i = 1,2,...,c \qquad j = 1,2,...,N$$

$$\sum_{i=1}^{c} \mu_{ij} = 1 \quad , \quad 0 < \sum_{j=1}^{N} \mu_{ij} < N$$

The membership grade is computed by:

- Computing the distance between every two feature vectors in the training process. The distance can be computed using Euclidean distance or other distances.
- The distance between every feature vector to itself is considered infinity.
- The columns of the distance matrix are sorted in an ascending order; keeping the labels of the nearest feature vectors.
- The membership grade is calculated using:

$$\mu_{ij} = \begin{cases} 0.51 + 0.49\left(n_{ij}/k\right) & \text{if } i = \text{the same as the label of the } j\text{th pattern.} \\ 0.49\left(n_{ij}/k\right) & \text{if } i \neq \text{the same as the label of the } j\text{th pattern.} \end{cases}$$

The mean vector of each class is calculated by:

$$\tilde{X}_i = \frac{\sum_{j=1}^{N} \mu_{ij} X_j}{\sum_{j=1}^{N} \mu_{ij}}$$

The between class fuzzy scatter matrix S_{FB} and within class fuzzy scatter matrix S_{FW} are calculated by:

$$S_{FB} = \sum_{i=1}^{c} n^i \left(\tilde{X}_i - \overline{X}\right)\left(\tilde{X}_i - \overline{X}\right)^T$$

$$S_{FW} = \sum_{i=1}^{c} \sum_{X_i \in n^i} \left(X_i - \tilde{X}_i\right)\left(X_i - \tilde{X}_i\right)^T = \sum_{i=1}^{c} S_{FW_i}$$

$$\overline{X} = \left(1/n\right) \sum_{j=1}^{n} X_j$$

Where $\overline{X}$ is the mean of all the training vectors, and n^i is the number of samples in class *i* and *c* is the number of classes. The optimal fuzzy projection and the new feature vector can be calculated by:

$$W_{F-FLD} = \arg\max_{W} \frac{\left|W^T S_{FB} W\right|}{\left|W^T S_{FW} W\right|}$$

$$\tilde{v}_i = W_{F-FLD}^T X_i = W_{F-FLD}^T E^T \left(z_i - \bar{z}\right)$$

E is the new subspace vector after applying Eigen method, and z_i is the image vector and $\bar{z}$ is the mean of the images vector.

4.2.4 Face specific subspace

The face specific subspace method was introduced by Shan *et al.* (2003). This method is newer compared to Eigen face method. The difference between this method and the Eigen face method is that this method is applied on the training images of each class separately. Considering a set of face classes

$$C = \{\Omega_1, \Omega_2, ..., \Omega_p\}$$

Where p is the number of faces that should be recognized. For the *kth* face class Ω_k , $k = 1,2,..., p$ the Eigen decomposition is computed by:

$$U_k^T \Sigma_k U_k = \Lambda_k$$

Where Σ_k is the covariance matrix of the *kth* face, Λ_k is a diagonal matrix and the eigenvalues $\lambda_1^k, \lambda_2^k, ..., \lambda_{d_k}^k$ of Σ_k are the diagonal elements in a decreasing order. $U_k = \left[U_1^{(k)}, U_2^{(k)}, ..., U_{d_k}^{(k)}\right]$ is a matrix which is calculated from the eigenvectors of matrix Σ_k and $U_1^{(k)}, U_2^{(k)}, ..., U_{d_k}^{(k)}$ are the Eigen vectors corresponding to $\lambda_1^k, \lambda_2^k, ..., \lambda_{d_k}^k$. The *kth* face specific subspace would be:

$$U_k = \left(U_1^{(k)}, U_2^{(k)}, ..., U_{d_k}^{(k)}\right)$$

The *kth* face which is the *kth* face specific subspace can be presented by:

$$\Re_k = \left(U_k, \Psi_k, \Lambda_k, d_k\right)$$

Where Ψ_k is the mean of the class *k* and d_k is the dimension of the face specific subspace, and Γ is an input image. The projection to the *kth* face specific subspace is:

$$W^{(k)} = U_k^T \Phi^{(k)}$$

$$\Phi^{(k)} = \Gamma - \Psi_k$$

And $\Phi^{(k)}$ can be constructed using:

$$\Phi_r^{(k)} = U_k W^{(k)}$$

The distance of the Γ image from the *kth* face specific subspace can be calculated by:

$$\varepsilon^{(k)} = \left\| \Phi^{(k)} - \Phi_r^{(k)} \right\|$$

The distance from the *kth* face specific subspace shows the quantity of the *kth* pattern which is hidden in the input image.

Also these relations can be used:

$$\varepsilon^{(k)} = \left\| \Gamma_r^{(k)} - \Psi^{(k)} \right\|$$

$$\varepsilon^{(k)} = \left\| \Gamma W^{(k)} - \Psi^{(k)} W^{(k)} \right\|$$

$$\Gamma_r^k = \Gamma W^{(k)} W^{(k)^T}$$

For reconstructing the original image:

$$\Gamma' = \left\| \Gamma - \Psi^{(k)} \right\| . \left(\Phi_r^{(k)} + \Psi^{(k)} \right)$$

4.2.5 Kernel Methods

Kernel methods are more recent methods compared to linear algorithms (Boser *et al.*, 1992). This method finds the higher order correlations between instances and the algorithm is as follows:

It is considered that patterns $x \in \Re^N$ are available, and that the most information lies in the d*th* dimension of pattern x.

$[x]_{j_1} \ldots [x]_{j_d}$

One way to extract all the features from data is to extract the relations between all the elements of a vector. In machine vision problems where all the images are converted to vectors this feature extraction shows the relations between all pixels of the image. For example in $\Re^2$ (an image) all the second order relations can be mapped to a non-linear space:

$$\Phi : \Re^2 \rightarrow F = \Re^3$$
$$([x]_1, [x]_2) \mapsto ([x]_1^2, [x]_2^2, [x]_1 [x]_2)$$

This method is useful for low dimension data but can cause problems for high dimensional data. For N dimensional data there is:

$$N_F = \frac{(N + d - 1)!}{d!(N - 1)!}$$

different combinations that make a feature space with N_F dimension. For example a 16*16 image and $d = 5$ have a feature space of moment 10^{10}. By using kernel methods there is no need to compute these relations explicitly.

For computing dot products $(\Phi(x).\Phi(x'))$ the kernel method is defined as follows:

$$k(x, x') = (\Phi(x).\Phi(x'))$$

That let us compute the dot product F without any need to map Φ. This method was first used in Boser *et al.* (1992). If x is an image then the kernel $(x.x')^d$ (or any other kernels) can be used

to map onto a new feature space. This feature space is called Hilbert space. In Hilbert space all the relations between any vectors can be shown by dot products. The input space is denoted χ and the feature space by F, and the map $\phi: \chi \to F$. Any function that returns the inner product of two points $x_i \in \chi$ and $x_j \in \chi$ in the F space is called a kernel function.

Some of the popular kernels are (Scholkopf, 2000):

Polynomial kernel:

$$k(x, y) = (x.y)^d$$

RBF kernel:

$$k(x, y) = \exp\left(- \|x - y\|^2 \Big/ 2\sigma^2 \right)$$

Sigmoid kernel:

$$k(x, y) = \tanh\left(\kappa (x.y)^d + \theta\right) d \in \aleph, \kappa > 0, \theta < 0$$

Also kernels can be combined using:

$$\alpha k_1(x, y) + \beta k_2(x, y) = k(x, y)$$

$$k_1(x, y) k_2(x, y) = k(x, y)$$

In order to produce new kernels.

By having m instances x_k with zero mean and $x_k = [x_{k1}, x_{k2}, ..., x_{kn}]^T \in \Re^n$, principal component analysis method finds the new axis in the direction of maximum variances of the data and this is equivalent to finding the eigenvalues of the covariance matrix C:

$$\lambda w = Cw$$

For eigenvalues $\lambda \geq 0$ and eigenvectors $w \in \Re^n$ in kernel principal component analysis, each vector x from the input space $\Re^n$ to the high dimensional feature space $\Re^f$ is mapped by a non-linear mapping function $\Phi : \Re^n \to \Re^f$, $f > n$. In $\Re^f$ the eigenvalue problem is as follows:

$$\lambda w^{\Phi} = C^{\Phi} w^{\Phi}$$

$$C^{\Phi} = \frac{1}{m} \sum_{j=1}^{m} \Phi(x_j) \Phi(x_j)^T$$

Where C^{Φ} is the covariance matrix. The eigenvalues $\lambda \geq 0$ and eigenvectors $w^{\Phi} \in F \setminus \{0\}$ (the eigenvectors that their eigenvalues are not zero) must be found in a manner that qualifies $\lambda w^{\Phi} = C^{\Phi} w^{\Phi}$. By using it we reach to:

$$\lambda(\Phi(x_k) \cdot w) = (\Phi(x_k) \cdot C^{\Phi} w) \quad k = 1,2,...,m$$

Also the coefficients α_i exists such that:

$$w^{\Phi} = \sum_{i=1}^{m} \alpha_i \Phi(x_i)$$

By combining the last three formulas and by introducing K as a $m \times m$ matrix:

$$K_{ij} = k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$$

The equation:

$$\lambda \sum_{i=1}^{m} \alpha_i (\Phi(x_k) \cdot \Phi(x_i)) =$$
$$\frac{1}{m} \sum_{i=1}^{m} \alpha_i \left(\Phi(x_k) \cdot \sum_{j=1}^{m} \Phi(x_j) \right) (\Phi(x_j) \cdot \Phi(x_i))$$
$$\Rightarrow m\lambda K\alpha = K^2 \alpha$$

is reached and the kernel principal component analysis will become:

$$m\lambda K\alpha = K^2\alpha \equiv m\lambda\alpha = K\alpha$$

Where α is a column vector with values $\alpha_1,...,\alpha_m$ (Scholkopf *et al.*, 1998).

For normalizing the eigenvectors in F that is $\left(w^k \cdot w^k\right)=1$ the equation used is:

$$1=\sum_{i,j=1}^{m}\alpha_i^k\alpha_j^k\left(\Phi(x_i)\cdot\Phi(x_j)\right)=\left(\alpha^k\cdot K\alpha^k\right)=\lambda_k\left(\alpha^k\cdot\alpha^k\right)$$

For extracting the principal components from the test instance x (and its projection in the $\Re^f$ space is $\Phi(x)$) only the projection must be computed of $\Phi(x)$ on the eigenvectors w^k in the feature subspace F by (Scholkopf *et al.*, 1998):

$$\left(w^k\cdot\Phi(x)\right)=\sum_{i=1}^{m}\alpha_i^k\left(\Phi(x_i)\cdot\Phi(x)\right)=\sum_{i=1}^{m}\alpha_i^k k(x_i,x)$$

It should be noted that none of equations need $\Phi(x_i)$ in an explicit way. The dot products must only be calculated using the kernel function without the need to apply the map Φ. In face recognition, each vector x shows a face image and this is why the non-linear principal component is called kernel Eigen face in face recognition.

In summary the kernel principal component analysis can be calculated as below:

1. Calculate the gram matrix by using:

$$K_{training}=\begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & ... & k(x_1,x_m) \\ k(x_2,x_1) & k(x_2,x_{21}) & ... & k(x_2,x_m) \\ ... & ... & ... & ... \\ k(x_m,x_1) & k(x_m,x_2) & ... & k(x_m,x_m) \end{bmatrix}$$

2. Calculate $m\lambda\alpha = K\alpha$ and compute α
3. normalize α^n using $1=\sum_{i,j=1}^{m}\alpha_i^k\alpha_j^k\left(\Phi(x_i)\cdot\Phi(x_j)\right)=\left(\alpha^k\cdot K\alpha^k\right)=\lambda_k\left(\alpha^k\cdot\alpha^k\right)$
4. Calculate the principal component coefficients for test data x using:

$$\left(w^k \cdot \phi(x)\right) = \sum_{i=1}^{m} \alpha_i^k k(x_i, x)$$

The classical principal component analysis is also a special version of kernel principal component analysis in which the kernel function is a first order polynomial. So the kernel principal component analysis is a generalized form of principal component analysis that has used different kernels for non-linear mapping. For computing kernel principal component analysis first the projection to new subspace is done and then principal component analysis is applied on the data in the new subspace (Scholkopf *et al.*, 1998).

Another important matter is using data with zero mean in the new subspace, which can be done using:

$$\tilde{\Phi}(x_i) = \Phi(x_i) - \left(1/m\right)\sum_{i=1}^{m} \Phi(x_i)$$

As there are no data in explicit form in the new space the following method can be used. By considering that for each i and j $1_{ij} = 1$.

$$\begin{aligned}
\tilde{K}_{ij} &= \tilde{\Phi}(x_i)^T \tilde{\Phi}(x_j) \\
&= \left(\Phi(x_i) - \sum_{l=1}^{m} \Phi(x_l)\right)^T \left(\Phi(x_j) - \sum_{n=1}^{m} \Phi(x_n)\right) \\
&= \Phi(x_i)^T \Phi(x_j) - \frac{1}{m}\sum_{l=1}^{m} \Phi(x_l)^T \Phi(x_j) \\
&\quad - \frac{1}{m}\sum_{n=1}^{m} \Phi(x_i)^T \Phi(x_n) + \frac{1}{m^2}\sum_{l,n=1}^{m} \Phi(x_l)^T \Phi(x_n) \\
&= K_{ij} - \frac{1}{m}\sum_{l=1}^{m} 1_{il} K_{lj} - \sum_{n=1}^{m} K_{in} 1_{nj} + \frac{1}{m^2}\sum_{l,n=1}^{m} 1_{il} K_{\ln} 1_{nj}
\end{aligned}$$

The formula can be rewritten as (Scholkopf, 2000):

$$\tilde{K}_{ij} = K - 1_m K - k1_m + 1_m K 1_m \qquad (1_m)_{ij} = 1/m$$

For kernel Fisher face first kernel principal component analysis is applied on the image and then LDA is applied on the new vector (Yang *et al.*, 2004).

4.2.6 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) is a method for calculating the linear relation between two sets of multidimensional variates. This method was introduced by Hotelling (1936). Assuming that X is a $m \times n$ matrix with m arrays of *n* dimensional vector of random variable *x.* The correlation coefficient ρ_{ij} which shows the correlation between variables x_i and x_j can be defined as:

$$\rho_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} C_{jj}}}$$

Where C_{ij} is the covariance matrix between x_i and x_j that is computed by:

$$C_{ij} = \frac{1}{m-1} \sum_{k=1}^{m} \left(X_{ki} - \mu_i \right) \left(X_{kj} - \mu_j \right)$$

and μ_i is the mean of x_i. Matrix A_x is defined as:

$$a_{ij} = X_{ij} - \mu_j$$

Therefore, the covariance matrix can be defined as:

$$C == \frac{1}{m-1} A_x^T A_x$$

The CCA method tries to find basis vectors for two sets of multi-dimensional variables in such a way that the linear correlation between the projections of the vectors on the basis vectors becomes maximum. Assuming the zero mean vectors X and Y, the CCA method would find vectors α and β in such a way that the correlations between the projections of $a_1 = \alpha^T X$ and $b_1 = \beta^T Y$ become maximum. The projections a_1 and b_1 are called the first canonical variables. Then a_2 and b_2 the second canonical variables are calculated which are uncorrelated with canonical variables a_1 and b_1, and this process can be continued.

$\omega_1, \omega_2, ..., \omega_c$ are samples belonging to class c and the training data space is defined as $\Omega = \{\xi \mid \xi \in \Re^N\}$, and $A = \{x \mid x \in \Re^P\}$ and $B = \{y \mid y \in \Re^q\}$ where x and y are two feature vectors extracted by two different method from one sample ξ. For calculating the first canonical correlations $\alpha_1^T x$ and $\beta_1^T y$ and the second canonical correlations $\alpha_2^T x$ and $\beta_2^T y$ the following can be used:

$$X^* = \left(\alpha_1^T x, \alpha_2^T x, ..., \alpha_d^T x\right)^T = (\alpha_1, \alpha_2, ..., \alpha_d)^T x = W_x^T x$$

$$Y^* = \left(\beta_1^T y, \beta_2^T y, ..., \beta_d^T y\right)^T = (\beta_1, \beta_2, ..., \beta_d)^T y = W_y^T y$$

$$Z_1 = \begin{pmatrix} X^* \\ Y^* \end{pmatrix} = \begin{pmatrix} W_x^T x \\ W_y^T y \end{pmatrix} = \begin{pmatrix} W_x & 0 \\ 0 & W_y \end{pmatrix}^T \begin{pmatrix} x \\ y \end{pmatrix}$$

$$W_1 = \begin{pmatrix} W_x & 0 \\ 0 & W_y \end{pmatrix}$$

$$W_x = (\alpha_1, \alpha_2, ..., \alpha_d), W_y = (\beta_1, \beta_2, ..., \beta_d)$$

Where α_i and β_i are called the *ith* canonical projective vectors, and W_1 and W_2 are canonical projective matrix and Z_1 is canonical correlation discriminant feature (Borga, 1998)(Sun *et al.*, 2005).

For determining the CCA coefficients, it is assured that x and y are two random variables with zero mean. The covariance matrix is defined by:

$$C = \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix} = E\left[\begin{pmatrix} x \\ y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}^T \right]$$

The correlation degree between x and y can be calculated using (Sun *et al.*, 2005):

$$\begin{cases} C_{xx}^{-1} C_{xy} C_{yy}^{-1} C_{yx} \alpha = \rho^2 \alpha \\ C_{yy}^{-1} C_{yx} C_{xx}^{-1} C_{xy} \beta = \rho^2 \beta \end{cases}$$

The solutions to the above formula are related to each other using:

$$\begin{cases} C_{xy} \beta = \rho \lambda_x C_{xx} \alpha \\ C_{yx} \alpha = \rho \lambda_y C_{yy} \beta \end{cases}$$

$$\lambda_x = \lambda_y^{-1} = \sqrt{\frac{\beta^T C_{yy} \beta}{\alpha^T C_{xx} \alpha}}$$

The above relations can be shown as:

$$B^{-1} A \hat{w} = \rho \hat{w}$$

$$A = \begin{bmatrix} 0 & C_{xy} \\ C_{yx} & 0 \end{bmatrix}, \; B = \begin{bmatrix} C_{xx} & 0 \\ 0 & C_{yy} \end{bmatrix}, \; \hat{w} = \begin{pmatrix} \mu_x \alpha \\ \mu_y \beta \end{pmatrix}$$

For face recognition, the feature vectors can be calculated by using two different methods and then the correlation between these two vectors can be computed using CCA. The new vectors can be used for recognizing faces.

4.2.7 Fractional Step Dimensionality Reduction

The Fractional Step Dimensionality Reduction method was introduced by Lotlikar and Kothari (2005). As mentioned before the optimal discriminant analysis is:

$$E_{optimal} = \arg\max_E \frac{\left\| E^T S_B E \right\|}{\left\| E^T S_W E \right\|}$$

Where S_W is the within class scatter matrix and S_B is the between class scatter matrix. It is important to mention that $E_{optimal}$ does not show the accuracy of the classifying. The reason is that Linear Discriminat Analysis method does not differ between the data of different classes that are close to each other with the data of different classes which are not close to each other. However, the data from different classes which are close to each other increase the probability of misclassifying and so a larger weight should be assigned to them. Also the dimension of the subspace can be changed from n to m $(m < n)$ fractionally. In this method the between class scatter matrix is:

$$S_B = \sum_{k=1}^{c} \sum_{l=1}^{c} w\left(d^{(kl)}\right)\left(\mu^{(k)} - \mu^{(l)}\right)\left(\mu^{(k)} - \mu^{(l)}\right)^T = \sum_{k=1}^{c} \sum_{l=1}^{c} w\left(d^{(kl)}\right) v^{(kl)} v^{(kl)T}$$

Where $\mu^{(k)}$ is the mean of the data in class k, c is the number of the classes, and $d^{(kl)} = \left\| \mu^{(k)} - \mu^{(l)} \right\|$ is the Euclidean distance between the mean of class k and class l and

$w\left(d^{(kl)}\right)$ is a decreasing function, because a larger weight should be assigned to classes which are closer to each other. The algorithm is as follows:

- Set the value $W = I_{n\times n}$, $I_{n\times n}$ is the identity matrix
- For k from n to $(m+1)$ with step = -1
    - For l from 0 to $(r-1)$ with step = 1
        - Project the data using $y = W^T x$
        - Scale the data using $z = \psi\left(y, \alpha^l\right)$
        - For z patterns calculate the between scatter matrix S_B (which is $k \times k$)
        - Calculate the Eigen values $\lambda_1, \lambda_2, ..., \lambda_k$ and Eigen vectors $\Phi_1, \Phi_2, ..., \Phi_k$ of matrix S_B
        - Update W using $W = W\overline{\Phi}$, $\overline{\Phi} = \left[\Phi_1, \Phi_2, ..., \Phi_k\right]$
    - End for
    - Discard the *kth* column of W
- End for

Where r is the number of fractional step.

The scale transformation compresses the last column of y with a factor of α^l with $\alpha < 1$, $\psi\left(y; \alpha^l\right): y \in R^k \rightarrow z \in R^k$ and:

$$z_i = \begin{cases} \alpha^l y_i & i = k \\ y_i & i = 1,2,...,(k-1) \end{cases}$$

In the *rth* step the reduction factor would be $1, \alpha, \alpha^2, ..., \alpha^{r-1}$. In the implementation numerous steps should be used and α should be chosen due to the number of steps. For smaller steps α should be chosen larger and vice versa. Also in order to increase $v^{(pq)}$ with the decreasing value of $d^{(pq)}$ the weight function should be decreased faster than $\|v\|^2$, so that the weight functions should be chosen as $d^{-3}, d^{-4}, ...$ and not d^{-1}, d^{-2} (Lotlikar and Kothari, 2005).

This method can be combined with linear and non-linear methods discussed earlier in this chapter.

Chapter 5

Methodology – Skin Detection

In this chapter we describe the methods that we have used in order to find skin pixels in an image. We used color images because finding faces in color images seems to be faster than finding faces on black and white images. The reason is that the methods than are applied to black and white images should search every part of the image in order to find faces. As described, Rowley and Viola methods both search all places of an image in order to find faces; however, in color images first we divide the image into two parts, the parts than contain human skin and other parts. After this phase, the search for finding human face would be restricted to those areas that just contain human skins. Therefore, face detection using color images is faster than other methods. It is very important that the skin detection phase works correctly; missing parts that contain human skin can cause missing some faces in the face detection phase. We present a methodology that utilizes neural networks.

5.1 Methodology

Neural network is a strong tool in learning, so we decided to use neural network for learning pixels' colors, so we can distinguish between what is face skin pixel and what is a non-face skin pixel. We decided to use information from more than one color space instead of using just the information from one color space (Hajiarbabi and Agah, 2014). We gathered around 100,000 pixels for face and 200,000 for non-face pixels from images chosen from the Web. Figure 5-1 shows some of the samples for the human skin and Figure 5-2 shows some of the samples for the non-skin.

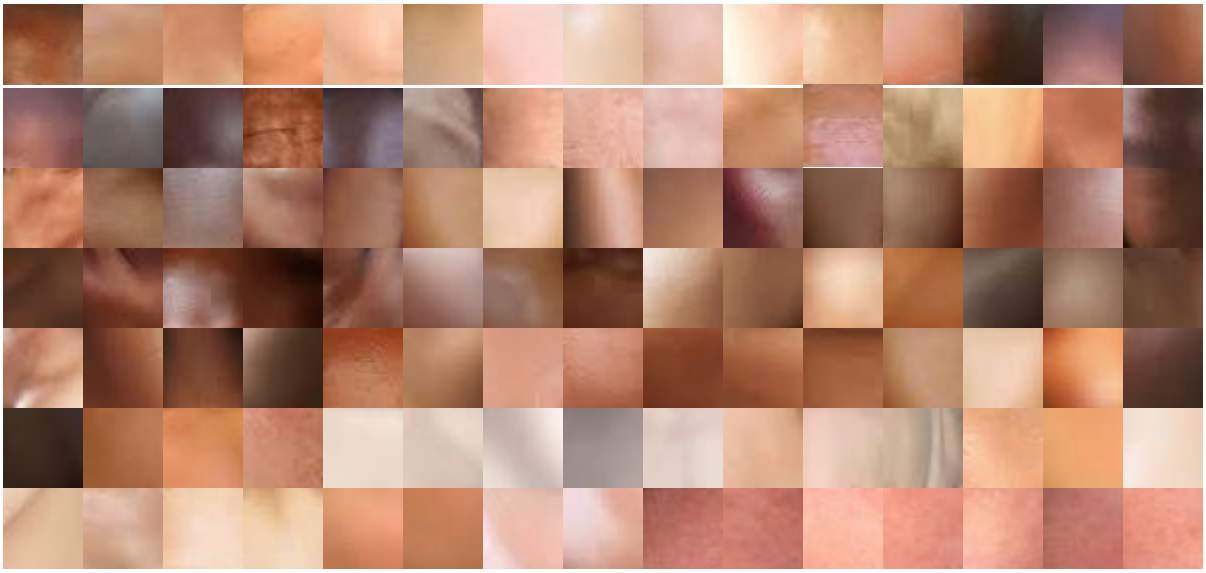


Figure 5-1: Samples of human skin

Figure 5-2: Samples of non-human skin

Choosing images for the non-human skin is a rather difficult task, because that is an enormous category, i.e., everything which is not human skin is non-skin. We tried to choose images from different categories, especially those which are very similar to human skin color, such as sand, surfaces of some desks, etc. We used such samples in training the neural network so that the network can distinguish them from human skin.

For the implementation, the MLP neural networks were used. Several entities can be used as the input of the neural network, namely, RGB, HSV, YCbCr (in this case Y is not used because it has the illumination information which is not suitable for skin detection). The number of outputs can be one or two. If there is just one output, then a threshold can be used. For example, an output greater than 0.5 shows that the input pixel belongs to skin, and less than that shows that it belongs to non-skin. For two outputs, one output belongs to skin and the other to non-skin. The larger value identifies the class for the pixel.

Around 50% of samples were used for training and the rest for testing. Different numbers of neurons were examined in the hidden layer. As mentioned before, there are some pixels which have the same value between the skin and non-skin pixels, these pixels were assigned to skin category. The neural network was trained with different nodes in the hidden layer, ranging from four nodes to 40 nodes. The networks which produced better results were chosen for the test images. Figure 5-3 shows the recognition rate for different number of nodes in the hidden layer, where the neural network has just one output, using the CbCr as the input to the neural network. For most of the networks, having 16 or 20 nodes in the hidden layer produced better results in comparison to other number of neurons in the hidden layer.

The color spaces used included RGB, CbCr (eliminating Y), and HS (eliminating V). Also a combination of the different color space CbCrRGBHS was used as the input.

Another method that was used is the boosting method. We used three different boosting methods, AND, OR and VOTING. The outputs of the three different neural networks (RGB, CbCr and HSV) were used. If the output shows that the pixel belongs to human skin, it is considered as 1 and otherwise as 0. In the AND method all the outputs should confirm that the pixel belongs to

human skin so that the final decision is human skin for that pixel. In the OR, method if just one output shows that the pixel belongs to human skin, then that is enough for the final decision to consider that pixel as human skin. In VOTING method, two or more outputs out of three should show that the pixel belongs to human skin so that the final decision considers that pixel as human skin. More information on boosting and the mathematical methods that show how boosting can increase the detection rate, can be found in (Freund and Schapire, 1999) (Schapire *et al.*, 1997). These methods mentioned are mostly mathematical methods for boosting in machine learning problems, which are not relevant to neural networks and skin detection.

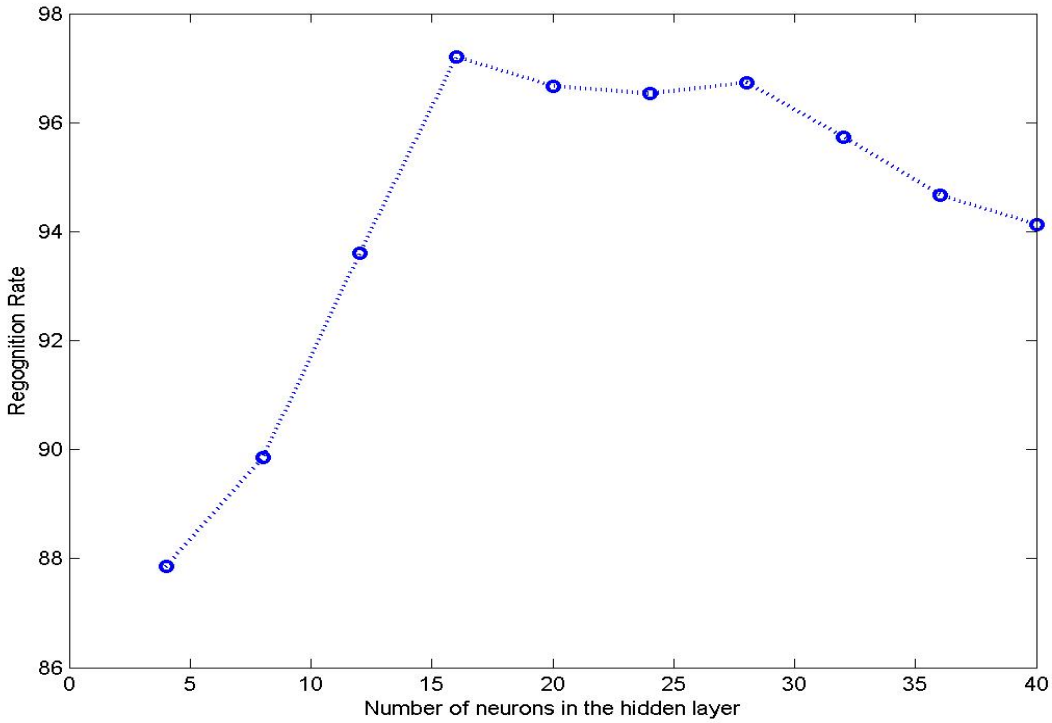


Figure 5-3: Recognition rate for human skin based on the number of nodes in the hidden layer

We trained several different neural networks and tested the results on the UCD database (UCD, 2014) and VT-AAST (Abdallah *et. al.*, 2007) using MATLAB (developed by Math Works) for implementation (Mathwork, 2014). The UCD database contains 94 images from different races. The images vary from one person in the image to multiple people. VT-AAST database contains 286 images which offer a wide range of difference in illumination and race. Both databases also contain the images after cropping the face skin.

The experimental results are reported as precision, recall, specificity and accuracy:

Precision or positive predictive value (PPV):

$$PPV = {TP}/{(TP + FP)}$$

Sensitivity or true positive rate (TPR) equivalent with hit rate, recall:

$$TPR = {TP}/{P} = {TP}/{(TP + FN)}$$

Specificity (SPC) or true negative rate:

$$SPC = {TN}/{N} = {TN}/{(FP + TN)}$$

Accuracy (ACC):

$$ACC = {(TP + TN)}/{(P + N)}$$

In the skin detection experiments, P is the number of the skin pixels, and N is the number of the non-skin pixels. TP is the number of the skin pixels correctly classified as skin pixels. TN is the number of the non-skin pixels correctly classified as non-skin pixels. FP is the number of the

non-skin pixels incorrectly classified as skin pixels. FN is the number of the skin pixels incorrectly classified as non-skin pixels.

5.2 Experimental Results

In the first experiment we chose the CbCr Component of the YCbCr, HS component of the HSV and RGB as the input of the neural network. The Y and V components were excluded because they do not affect skin detection.

The results for CbCr, HS and RGB are listed in Table 5-1. The CbCr experiments have better accuracy rate compared to HS and RGB. The HS experiments detect more skin pixels correctly than other color spaces.

	Precision	**Recall**	**Specificity**	**Accuracy**
CbCr	67.55	46.23	92.34	**80.52**
HS	56.66	**63.06**	83.37	78.17
RGB	**81.06**	26.45	**97.87**	79.56

Table 5-1: Accuracy results for CbCr, HS and RGB on UCD database

In another experiment we trained a neural network using the HSV color space. We included V in order to see the effect of excluding V on detecting skin pixels. The results for HS and HSV are shown in Table 5-2. The accuracy of using HS as input is higher than HSV, and also the Recall has decreased about 30% which is significant. Therefore, removing the V component produced better results.

	Precision	Recall	Specificity	Accuracy
HS	56.66	63.06	83.37	78.17
HSV	58.72	35.28	91.45	77.05

Table 5-2: Accuracy results for HS and HSV on UCD database

In another experiment we used the boosting method in order to increase the performance of the neural networks. We used AND, OR and VOTING among the three outputs of neural networks trained with CbCr, HS and RGB. The results are shown in Table 5-3. The AND operation has the highest precision and specificity compared to the other two methods but much lower recall; which means lots of skin pixels are considered as non-skin pixels. In case of recall, the OR method detects much more skin pixels correctly compared to other two methods, but also more non-skin pixels detected as skin pixel. The voting method has the highest accuracy among these three methods.

	Precision	Recall	Specificity	Accuracy
AND	**78.40**	21.45	**97.96**	78.35
OR	55.62	**70.36**	80.65	78.01
VOTING	73.34	39.53	95.05	**80.82**

Table 5-3: Accuracy results for AND, OR and VOTING on UCD database

We also generated a vector consisting of the information of the color spaces CbCrRGBHS and yielded the results in Table 5-4. This shows an improvement compared to previous methods.

	Precision	Recall	Specificity	Accuracy
CbCrRGBHS	77.73	51.35	95.92	81.93

Table 5-4: Accuracy results for CbCrRGBHS on UCD database

Another neural network was designed with having the same input but different nodes in the output. In this experiment two nodes were chosen for the output, one for the skin and the other for the non-skin. If the value of one output is higher, then it shows that the pixel belongs to that class. The results for CbCr, RGB and HS are listed in Table 5-5. Comparing Table 5-5 with Table 5-1 illustrates that there are improvements when we use two outputs for the neural network instead of one. RGB has higher accuracy than CbCr and HS; however, CbCr detects more human skin correctly than the other two methods.

	Precision	**Recall**	**Specificity**	**Accuracy**
CbCr	62.16	**60.54**	87.30	80.44
HS	69.55	42.28	93.62	80.46
RGB	**78.28**	39.52	**96.20**	**81.93**

Table 5-5: Accuracy results for CbCr, RGB and HS on UCD database

Figure 5-4 shows the ROC graph for CbCrRGBHS vector with two nodes in the output for the validation set. The results for CbCrRGBHS vector are included in Table 5-6. The results show that in the case of accuracy and recall we have some improvement, but the precision has decreased.

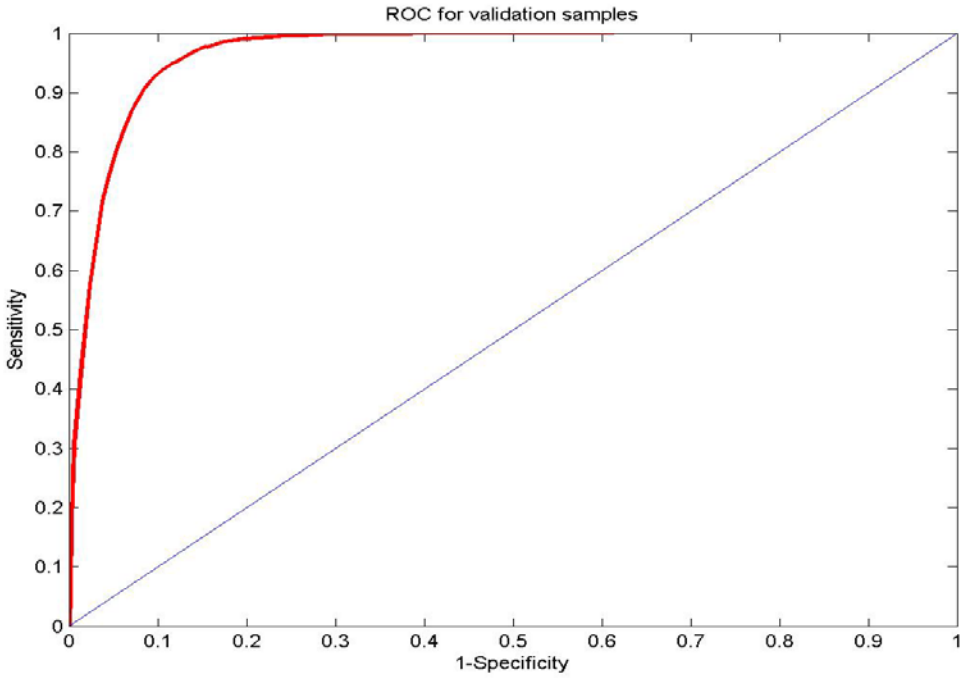


Figure 5-4: ROC graph for CbCrRGBHS on validation samples

	Precision	Recall	Specificity	Accuracy
CbCrRGBHS	71.30	60.25	93.43	82.36

Table 5-6: Accuracy results for CbCrRGBHS on UCD database

Table 5-7 shows the results of other methods discussed compared to our best results, when using the UCD database (UCD, 2014). Comparing the other methods with the result that we have from the CbCrRGBHS vector shows that our result is better in precision, specificity and accuracy. Our method accepts much fewer non-skin pixels as skin, compared with the other methods.

One thing to mention is that there is a tradeoff between precision and recall. If we want to have high recall (detecting more skin pixels correctly) then it is highly possible to detect many non-skin pixels as human skin which will reduce the precision and vice versa. Table 5-3 shows this concept, where AND and VOTING methods have high precision but low recall; while OR method has high recall but low precision.

	Precision	**Recall**	**Specificity**	**Accuracy**
Gaussian	54.96	66.82	81.12	77.46
Chen	63.75	51.13	89.98	80.02
Kovac	62.51	**69.09**	85.71	81.45
Kong	37.47	14.58	91.61	71.87
CbCrRGBHS	**71.30**	60.25	**93.43**	**82.36**

Table 5-7: Accuracy results for other methods on UCD database

Also we applied the designed neural network on VT-AAST database using CbCrRGBHS as the input. Table 5-8 shows the results of our method comparing with other methods on VT-AAST database. Our results are better in precision and accuracy and comparable in recall and specificity with the best results.

	Precision	**Recall**	**Specificity**	**Accuracy**
Gaussian	30.00	60.37	79.84	77.40
Chen	31.62	54.59	83.10	79.53
Kovac	31.81	**65.02**	80.05	78.17
Kong	45.97	29.51	**95.03**	86.83
CbCrRGBHS	**54.26**	59.07	93.36	**88.56**

Table 5-8: Accuracy results for other methods on VT-AAST database

In order to improve the result, two operations were done at the output image from the neural network. First operation was using filling method, in which the holes in the components are filled. This method is useful and cause increase in recall, the reason is that the output image from the neural network may have some undetected part inside the face or other parts of the body components, which can be recovered by using this method.

The other operation is the opening method, which is applying erosion follow by dialation method. Erosion is a morphological operation in image processing. A small disk (which is called structuring element) is defined which will be moved on the image. The center of the structuring element is placed on each pixel of the binary image which has the one value. If all the image pixels which is covered by the structuring element is one then that pixel (which is covered by the center of the structuring element) is considered one otherwise zero in the final image. In dilation process, like erosion, the center of the structuring element is placed on the image (Those pixels which are one). Now any zero pixels which are covered by the structuring element pixels are changed to one in the final image.

The structuring element which was used by us was 3*3. This size had better results than other structuring elements. Table 5-9 and Table 5-10 show the result after applying filling and opening method, the modified method has been called CbCrRGBHS+.

	Precision	**Recall**	**Specificity**	**Accuracy**
CbCrRGBHS	71.30	60.25	93.43	82.36
CbCrRGBHS+	73.43	65.54	93.08	83.45

Table 5-9: Accuracy results for CbCrRGBHS+ on UCD database

	Precision	Recall	Specificity	Accuracy
CbCrRGBHS	54.26	59.07	93.36	88.56
CbCrRGBHS+	54.77	62.61	93.07	88.76

Table 5-10: Accuracy results for CbCrRGBHS+ on VT-AAST database

Figures 5-5 to 5-15 illustrate some of our experimental results on images from the UCD database (UCD, 2014). These are produced using the CbCrRGBHS vector and two outputs for the neural network. The second image is the output from the neural network and the third image is the one after applying morphological operatioan. We first filled the holes that were in the image. After that we applied erosion followed by dialation operation. Figure 5-16 and 5-17 illustrate some of our result on VT-AAST database. There are some false positives in some of images. That is because some objects' colors are very similar to human skin color.

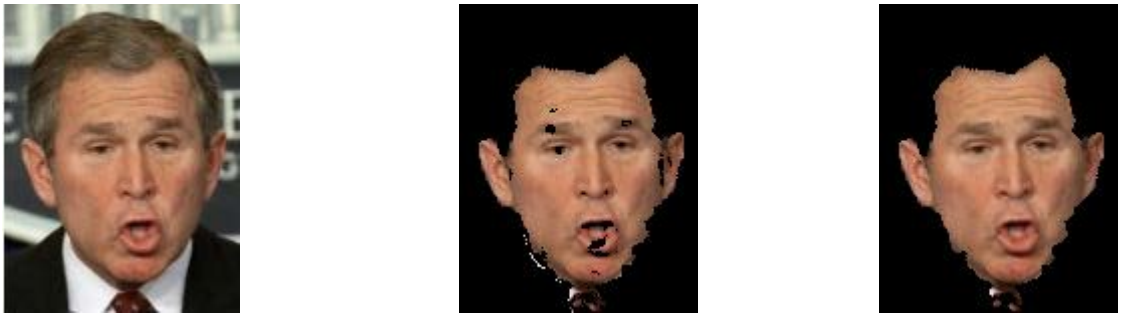


Figure 5-5: Experimental results, image containing one person on UCD database (1)

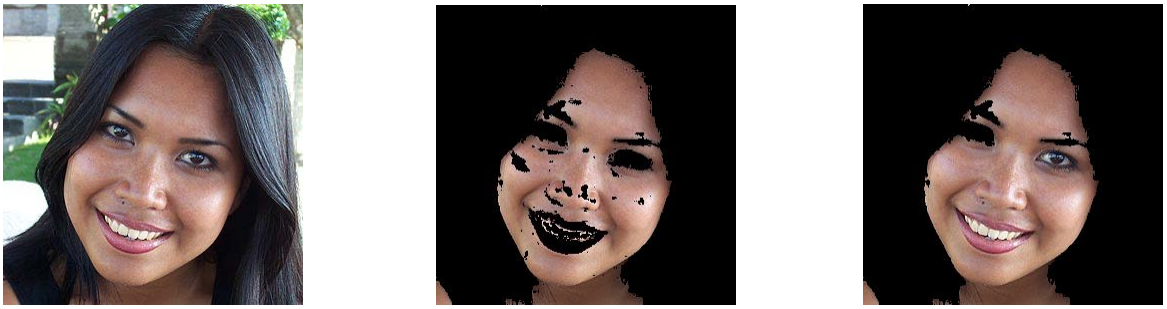


Figure 5-6: Experimental results, image containing one person on UCD database (2)

Figure 5-7: Experimental results, image containing one person on UCD database (3)

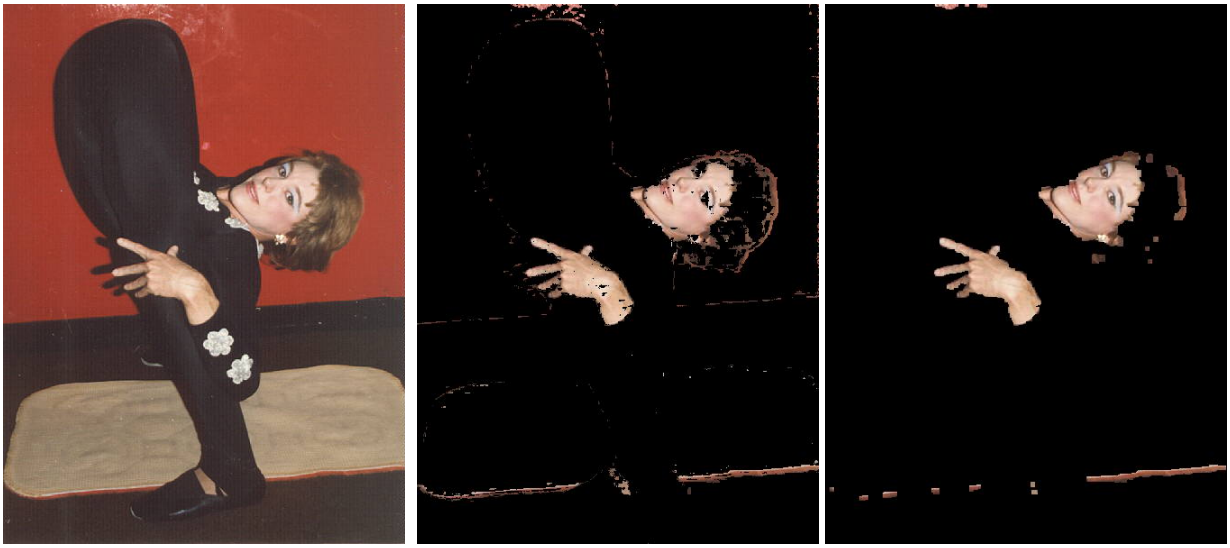


Figure 5-8: Experimental results, image containing one person on UCD database (4)

Figure 5-9: Experimental results, image containing two people on UCD database (1)

Figure 5-10: Experimental results, image containing two people on UCD database (2)

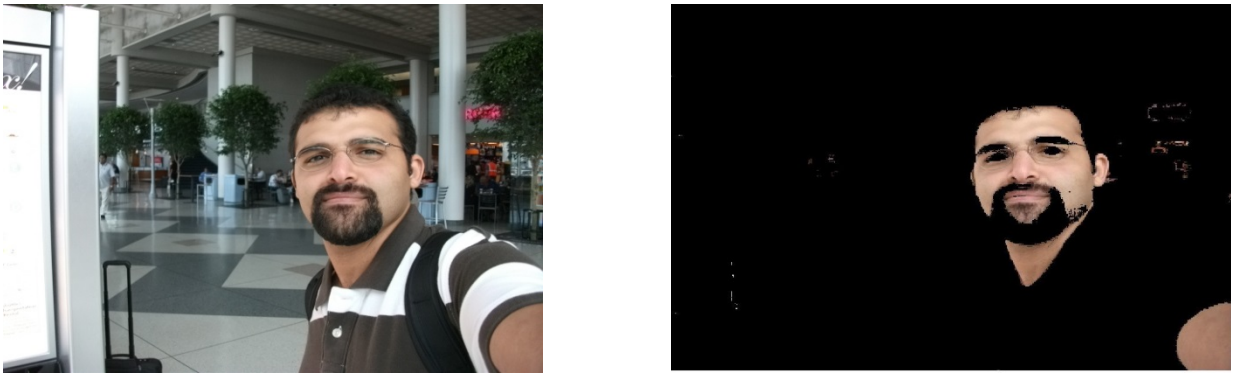
Figure 5-11: Experimental results, image containing three people on UCD database (1)

Figure 5-12: Experimental results, image containing three people on UCD database (2)

Figure 5-13: Experimental results, image containing four people on UCD database (1)

Figure 5-14: Experimental results, image containing four people on UCD database (2)

Figure 5-15: Experimental results, image containing multiple people on UCD database



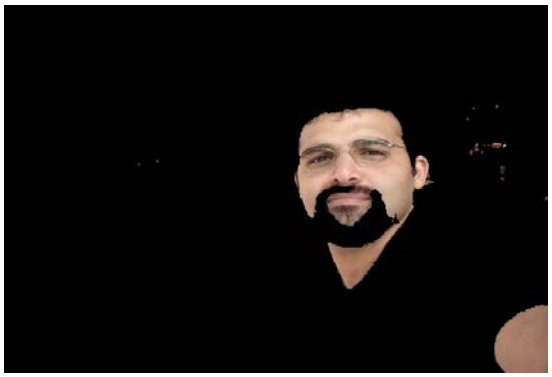
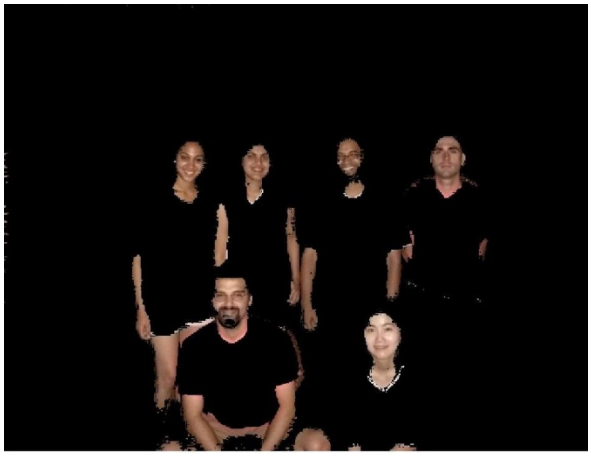


Figure 5-16: Experimental results, image containing one person on VT-AAST database



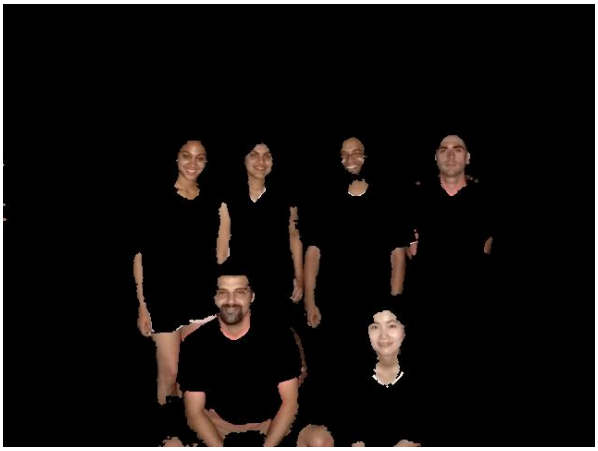


Figure 5-17: Experimental results, image containing multiple people on VT-AAST database

On some images when the illumination of the environment effects the illumination of the image and so the skin colors or when the illumination of the environment is very low the results are not good. Figure 5-18 shows some images from UCD and VT-AAST database which their skin detection results were not acceptable.

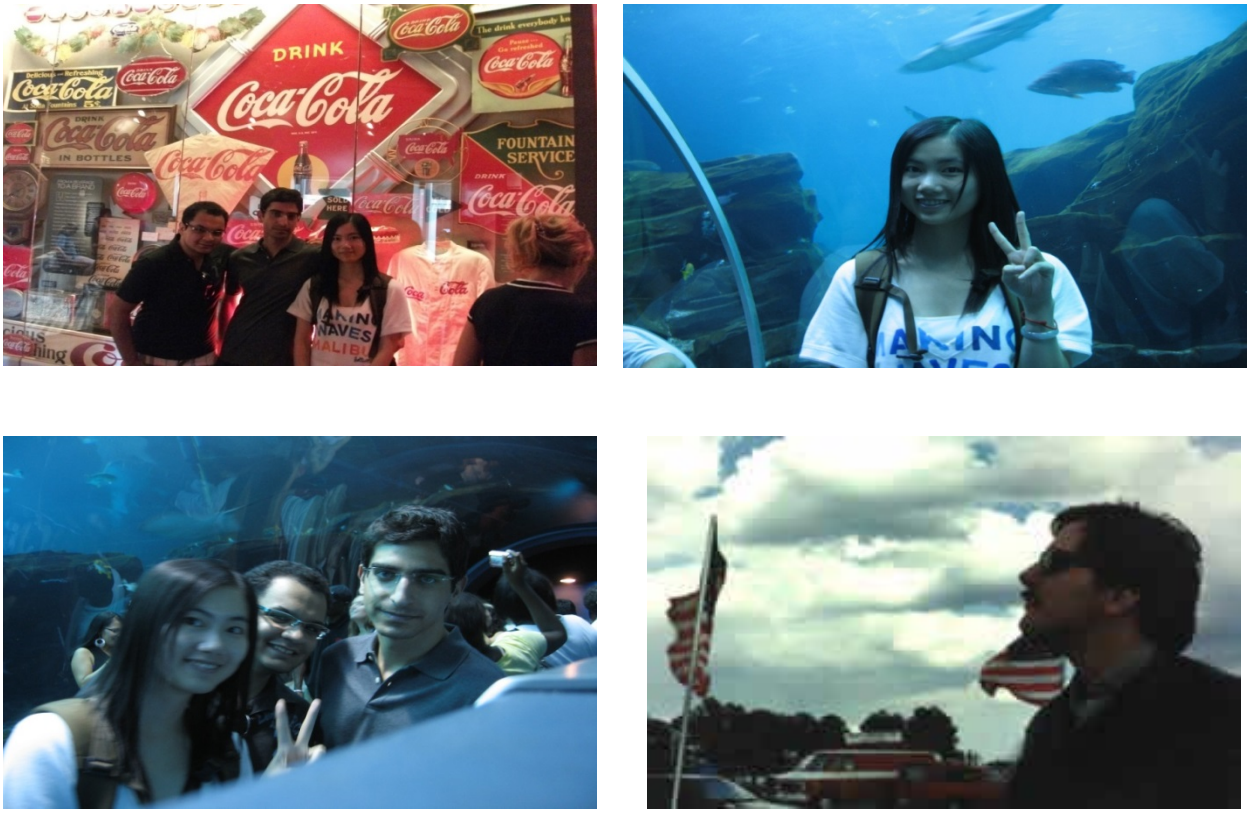


Figure 5-18: Images with poor skin detection result from UCD and VT-AAST database

As can been seen, in one of the images the people are in an environment with domination of red color, which has affected the human skin colors, the same situation can be seen in two of the

images with domination of blue in the environment which has changed the people's skin color in the images. Also low illumination can be noticed in the last image, so to get best results from skin detection the system should be designed for places where the illumination of the environment is under control.

5.3 Increasing the Performance of the Neural Network

We performed a number of experiments in order to increase the power of the neural network. These two methods were suggested by Rowley *et al.* (1992) in their face detection work in order to increase the detection rate of the faces, but here we used it for skin detection.

In the first method, we gathered several images that do not contain human skin. These images were applied to the neural network. The pixels which were considered as human skin were collected and added to the training pixels. The process was repeated two times. By using this method some new pixels were found and added to the non-skin pixels. Also as mentioned before, some of the pixels were common between the human skin color and non-skin color, after using this method more common pixels were found.

We used 10 fold cross validation for training the neural network. Tables 5-11 and 5-12 show the primary results and the results after using bootstrapping method on UCD and VT-AAST databases. CbCrRGBHSI shows the results after applying the first bootstrapping method for the first time and CbCrRGBHSII shows the results after applying the first bootstrapping method for the second time.

	Precision	Recall	Specificity	Accuracy
CbCrRGBHS	63.19	55.55	88.85	80.31
CbCrRGBHS I	64.90	58.04	89.18	81.20
CbCrRGBHS II	67.79	57.02	90.66	82.04

Table 5-11: Accuracy results for CbCrRGBHS on UCD database using 10 fold cross validation

	Precision	Recall	Specificity	Accuracy
CbCrRGBHS	42.47	65.28	87.34	84.58
CbCrRGBHS I	45.53	63.91	89.06	85.91
CbCrRGBHS II	49.13	68.09	89.91	87.18

Table 5-12: Accuracy results for CbCrRGBHS on VT-AAST database using 10 fold cross validation

As can be seen in Tables 5-11 and 5-12 the results have been increased by using bootstrapping method, although it was expected that the results increase more. The results can increase if the number of bootstrapping is increased. Using the samples from the last phase five different neural network we trained using different initial weights, different random sets of skin and non-skin pixels, and different permutations of the pixels that were presented to the network. Although the networks had close detection and error rates, the errors that they made were different from one another. We used a combination of the networks using AND, OR and voting method. The results are shown in Tables 5-13 and 5-14 for the UCD and VT-AAST database.

	Precision	Recall	Specificity	Accuracy
AND	68.52	55.87	**91.15**	**82.11**
OR	59.50	**64.43**	84.89	79.64
VOTE	**63.88**	59.54	88.39	81.00

Table 5-13: Accuracy results for CbCrRGBHS on UCD database using combination of networks

	Precision	Recall	Specificity	Accuracy
AND	**51.11**	61.54	**91.57**	**87.81**
OR	42.17	**70.32**	86.19	84.21
VOTE	45.55	65.26	88.83	85.88

Table 5-14: Accuracy results for CbCrRGBHS on VT-AAST database using combination of networks

Tables 5-15 and 5-16 show the results after applying the two computer vision methods on the images, filling the holes and opening method.

	Precision	Recall	Specificity	Accuracy
AND	**69.93**	60.66	**91.01**	**83.23**
OR	60.17	**69.24**	84.20	80.37
VOTE	65.23	64.93	88.07	82.14

Table 5-15: Accuracy results for CbCrRGBHS on UCD database using combination of networks after applying filling the holes and opening method

	Precision	Recall	Specificity	Accuracy
AND	**50.99**	65.42	**91.00**	**87.80**
OR	42.25	**74.56**	85.41	84.05
VOTE	45.52	69.66	88.07	85.76

Table 5-16: Accuracy results for CbCrRGBHS on VT-AAST database using combination of networks after applying filling the holes and opening method

Table 5-17 shows the comparison between different methods with CbCrRGBHS and VOTE method on UCD database and Table 5-18 shows the same result on VT-AAST database.

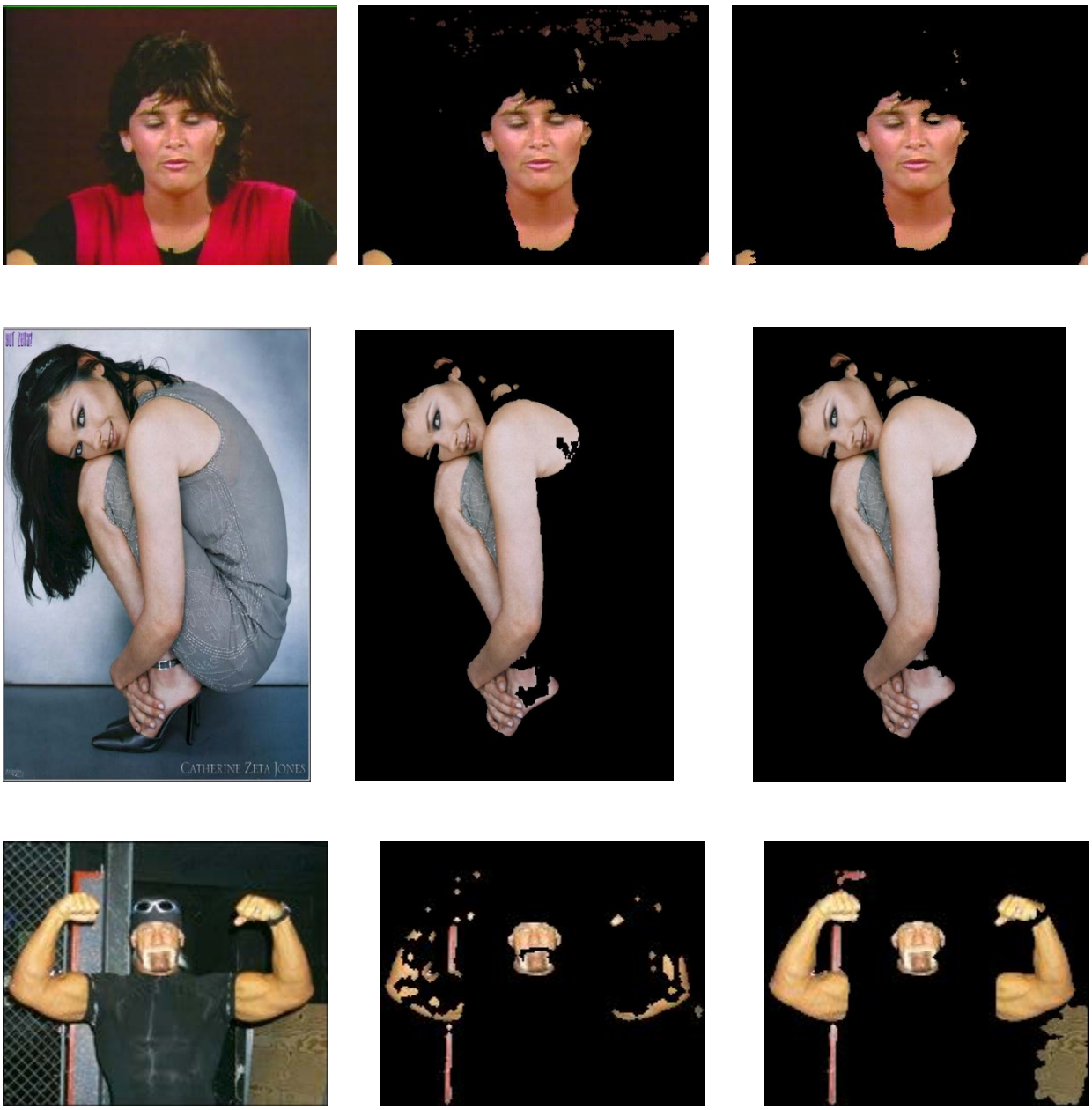
	Precision	Recall	Specificity	Accuracy
Gaussian	54.96	66.82	81.12	77.46
Chen	63.75	51.13	89.98	80.02
Kovac	62.51	**69.09**	85.71	81.45
Kong	37.47	14.58	**91.61**	71.87
CbCrRGBHS	63.19	55.55	88.85	80.31
CbCrRGBHS-VOTE	**67.23**	66.93	90.07	**84.14**

Table 5-17: Accuracy results for other methods on UCD database.

	Precision	Recall	Specificity	Accuracy
Gaussian	30.00	60.37	79.84	77.40
Chen	31.62	54.59	83.10	79.53
Kovac	31.81	65.02	80.05	78.17
Kong	45.97	29.51	**95.03**	86.83
CbCrRGBHS	42.47	65.28	87.34	84.58
CbCrRGBHS-VOTE	**47.52**	**71.66**	90.07	**87.76**

Table 5-18: Accuracy results for other methods on VT-AAST database.

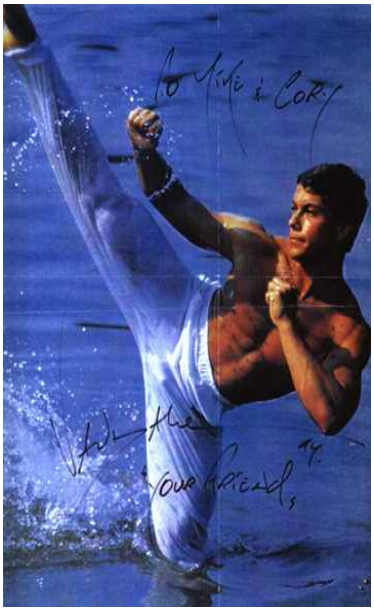
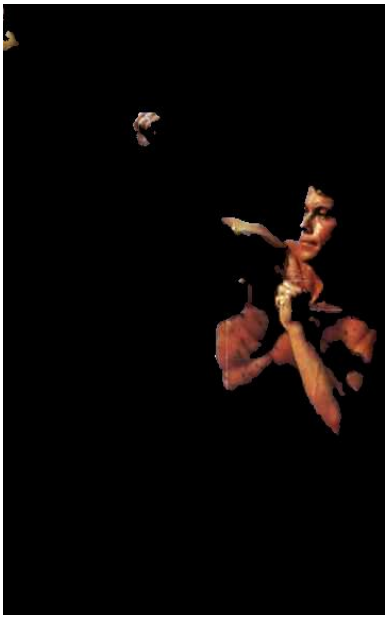
Figure 5-19 and 5-20 show some of the images from the UCD and VT-AAST database, the first image is the original image, the second image is after applying the image to the single neural network and the third image is the output using combination of networks.



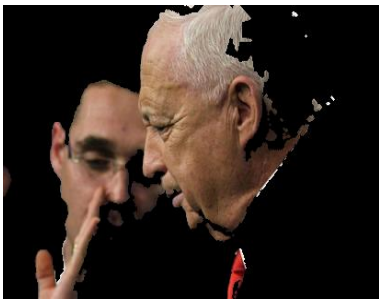

ישראל זוכרת
YITZHAK RABIN CENTER









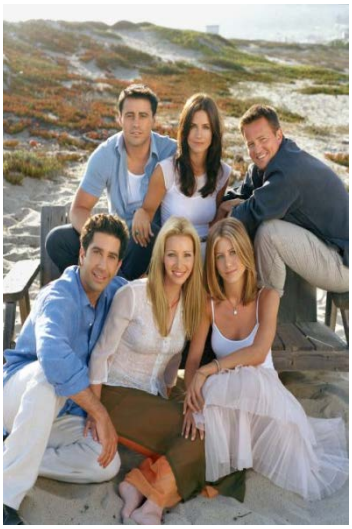
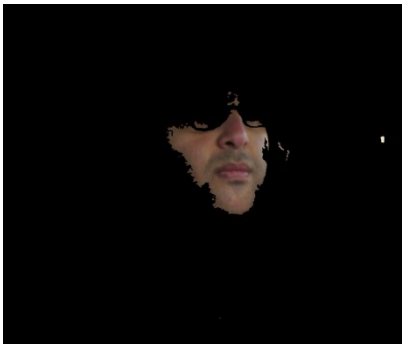





Figure 5-19: Results after increasing the performance of the neural networks from UCD database

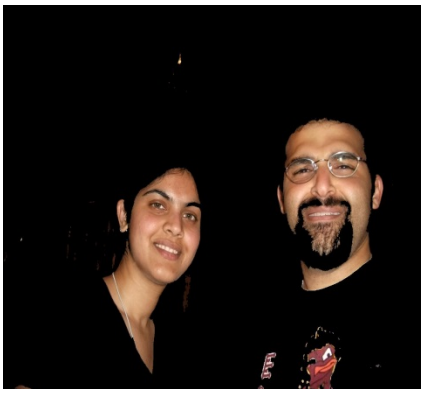



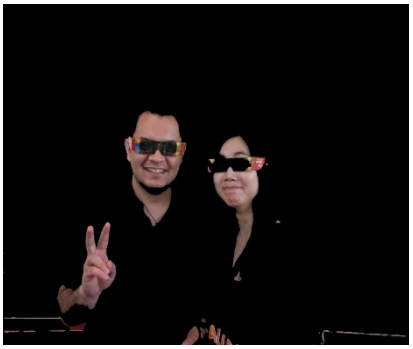


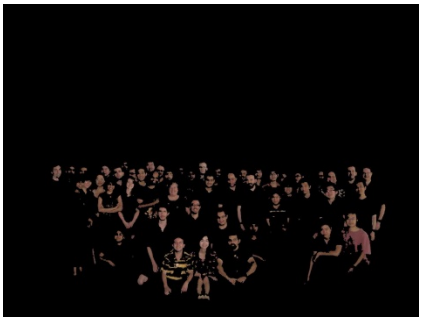
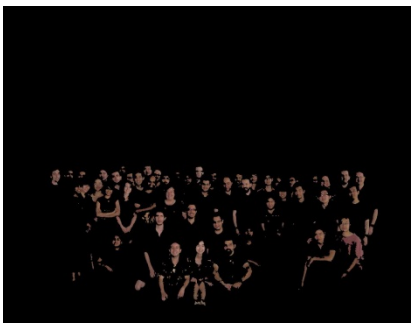






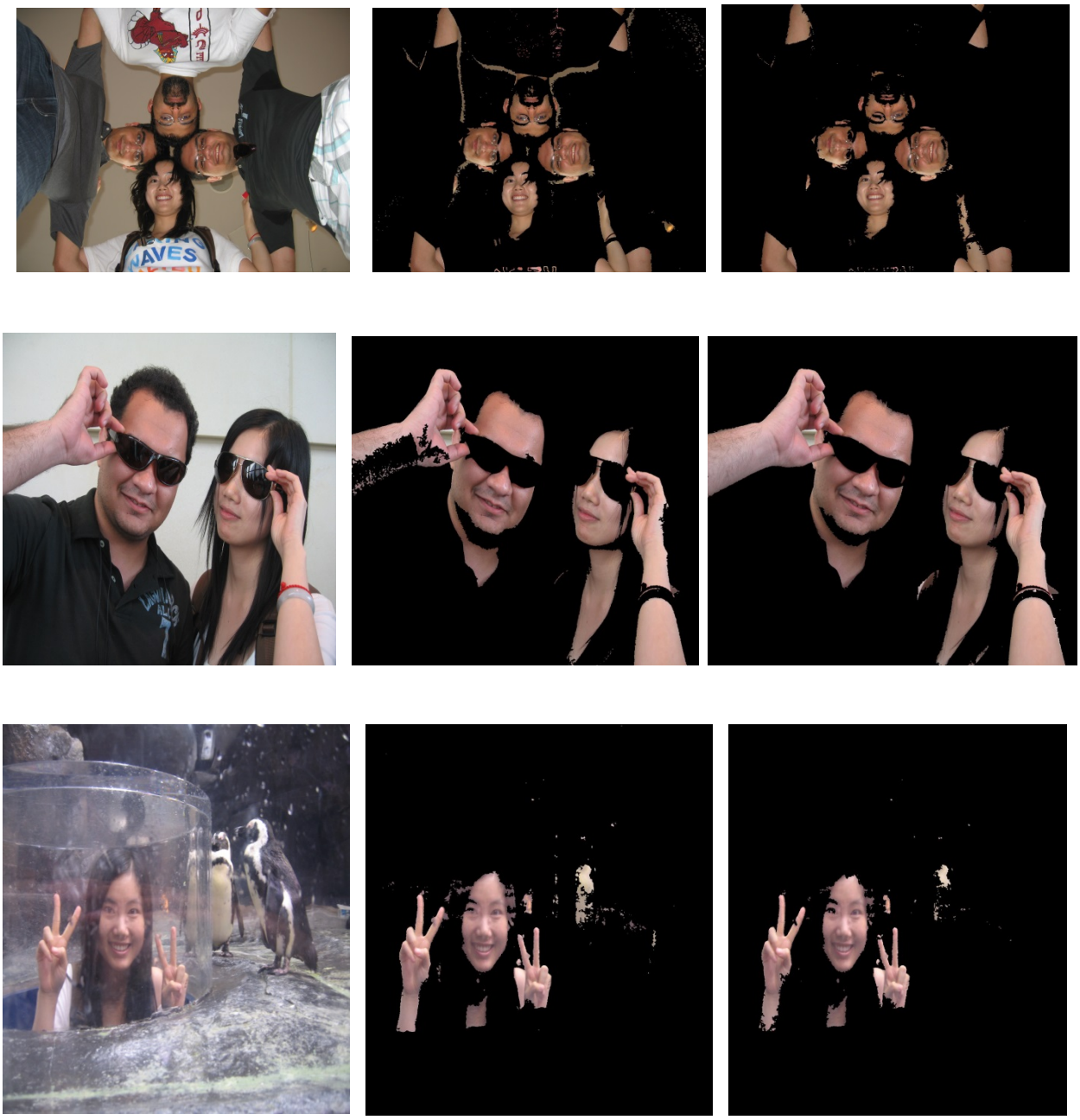


Figure 5-20: Results after increasing the performance of the neural networks from VT-AAST database

5.4 Deep Learning

In artificial neural networks, using more than one hidden layer usually not only decreases the performance rate but also increases the training and testing time (Tesauro, 1992). The reason is that by increasing the layers, it becomes too difficult to train deep multi-layer neural network. A greedy layer wise unsupervised learning algorithm was designed for Deep Belief Networks (DBN) by Hinton *et al.* (2006). The upper layers of the DBN contain abstract concepts of the inputs and the lower layer shows the low level features of the input. The classifier will first learn the low level features and by combining this information it learns the abstract concepts (Hinton *et. al.*, 2006).

A DBN can consist of several Restricted Boltzmann Machine (RBM) networks. A RBM is a special kind of Boltzmann Machine with the difference that the nodes in each layer are not connected to each other (Hinton *et. al.*, 2006). For generating data, Gibbs sampling can be applied between layers. By knowing the states of the units in one layer, the units in other layers are updated in parallel until the system reaches its equilibrium distribution (Hinton *et. al.*, 2006). There are several methods for generating data, among them contrastive divergence is a fast way for generating data using Gibbs sampling (Mayraz and Hinton, 2001).

One problem in the learning process is that learning the weights one layer at each time is not optimal. When the weights in the higher layers are learned, the weights in the lower layers are not optimal anymore (Hinton *et. al.*, 2006). In order to correct the weights an up down fine tuning is also needed (Hinton *et. al.*, 2006). Hinton et. al. (2006) and Bengio *et al.* (2007 have done experiments on MNIST database using deep learning methods. MNIST is a handwritten

digits database which contains 60,000 samples for training and 10,000 samples for testing. Researches have published results on MNIST database using different techniques. Hinton *et al.* (2006) and Bengio *et al.* (2007) experiments show that deep learning has much better performance compared to other techniques.

To our knowledge, deep learning has not been applied to the skin detection problem, and here experiments performed on the skin database are presented. As mentioned before this database was generated using images on World Wide Web. (Hajiarbabi and Agah, 2016).

Different Deep Belief Nets with different number of nodes in each layer (The number of nodes in each Restricted Boltzman Machine) were considered in this experiment. The number of nodes in each layer ranged from 250 to 500 with a 50 interval between each experiment. Figures 5-21 and 5-22 show the accuracy, recall, precision and specificity, compared to the number of nodes used in the hidden layer on UCD and VT-AAST database.

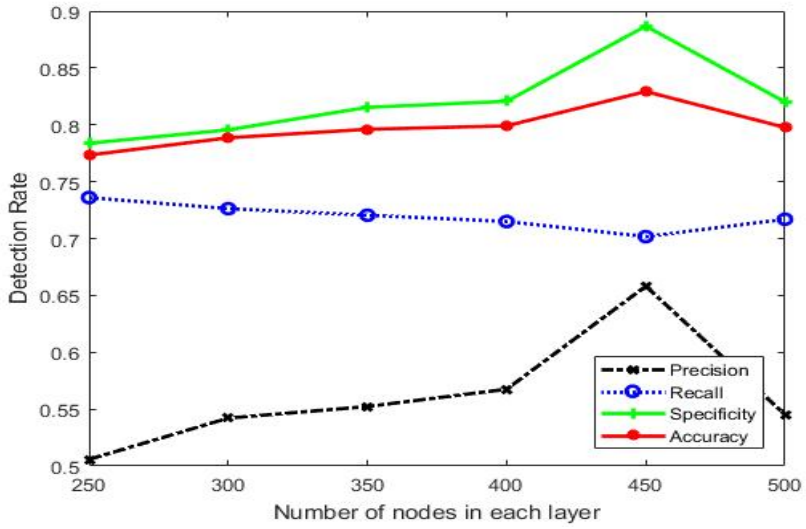


Figure 5-21: The performance of Deep Belief Net using different nodes in each layer on UCD database

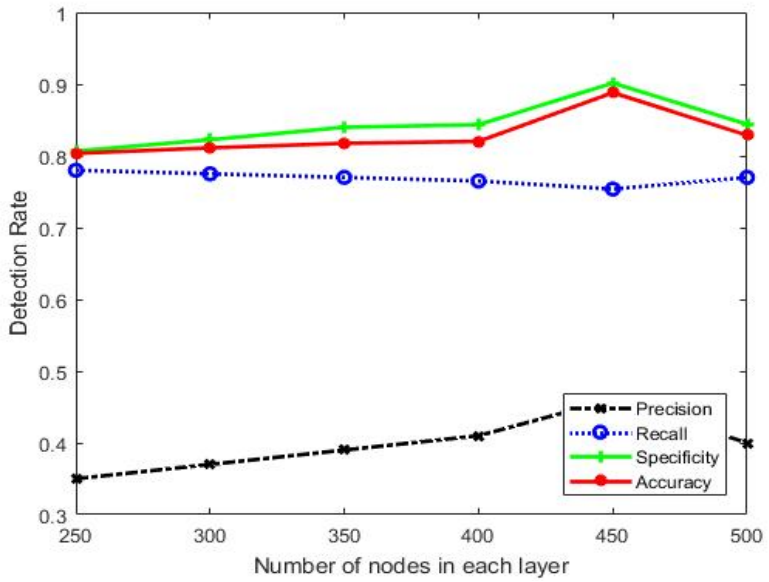


Figure 5-22: The performance of Deep Belief Net using different nodes in each layer on VT-AAST database

As can be seen in Figure 5-21 and Figure 5-22, the best results occur when the number of nodes in the hidden layer is 450. The other observation is the tradeoff between recall with precision, accuracy and specificity. The reason is that for having higher recall which is detecting more skin pixels correctly, then the probability that to detect more non-skin pixels as human skin is higher which will reduce the precision and vice versa.

When detecting faces there may be some parts of the skin inside the skin components that have not been detected as human skin; and also some small pixels from the background that have been considered as human skin. In order to increase the skin detection rate, some morphological operations were applied to the images. The first operation was filling the holes. This operation considers the pixels which are inside a skin component as skin. The other operation that is used is the opening method, which removes the small pixels from the image.

Table 5-19 shows the result of using deep learning (Deep Belief Net) along with the other methods discussed using the UCD database. Skin detection using deep learning shows better results in terms of recall and accuracy.

	Precision	**Recall**	**Specificity**	**Accuracy**
Gaussian	54.96	66.82	81.12	77.46
Chen	63.75	51.13	89.98	80.02
Kovac	62.51	69.09	85.71	81.45
Kong	37.47	14.58	91.61	71.87
Neural Network	**71.30**	60.25	**93.43**	82.36
Deep learning	65.81	**70.17**	88.68	**82.93**

Table 5-19: Accuracy results for other methods on UCD database.

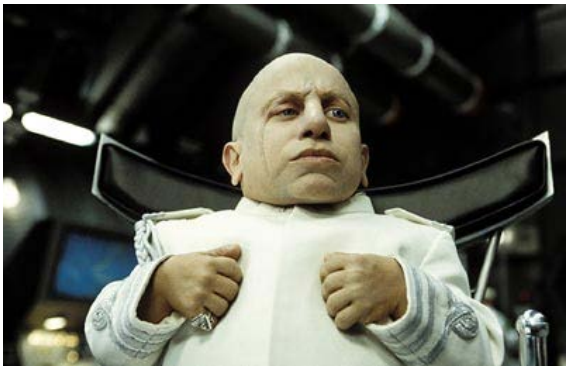
Table 5-20 shows the result of different methods on VT-AAST.

	Precision	**Recall**	**Specificity**	**Accuracy**
Gaussian	30.00	60.37	79.84	77.40
Chen	31.62	54.59	83.10	79.53
Kovac	31.81	65.02	80.05	78.17
Kong	45.97	29.51	**95.03**	86.83
Neural Network	**54.26**	59.07	93.36	88.56
Deep learning	46.05	**75.38**	90.15	**88.81**

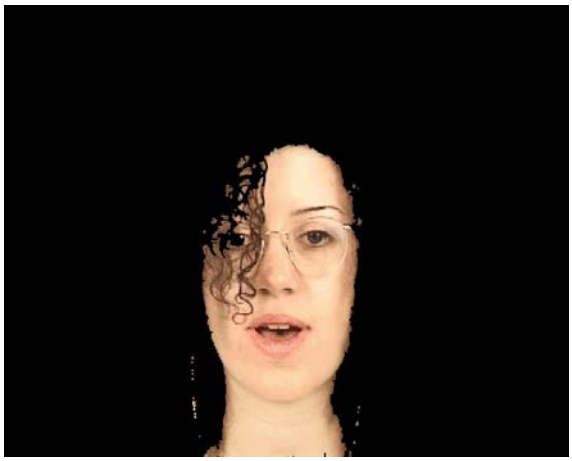
Table 5-20: Accuracy results for other methods on VT-AAST database.

Figure 5-23 illustrates some of the experimental results on images from the UCD database. The first image shows the original image from the database and the second image shows the result

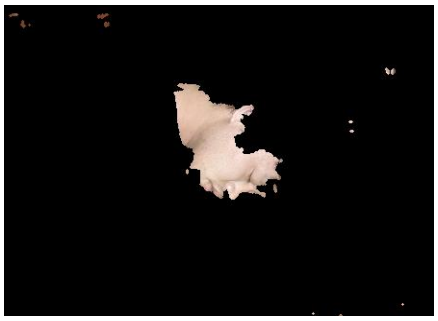
after applying deep learning method for detecting the skins followed by filling the holes and opening methods on the result. Figure 5-24 illustrates some of our result on VT-AAST database.

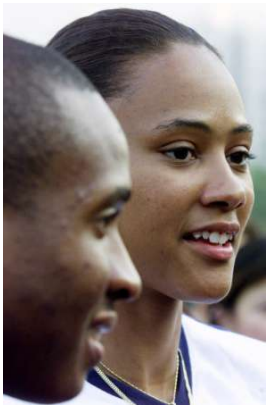










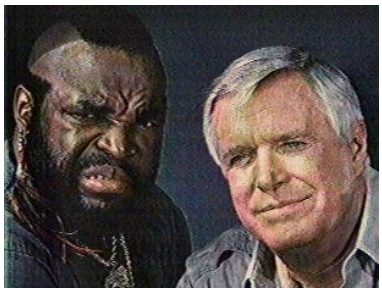
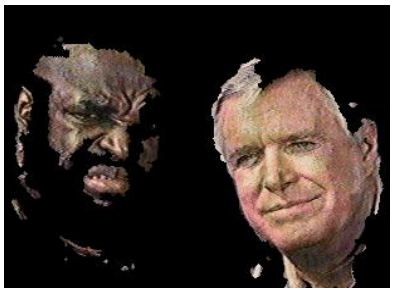











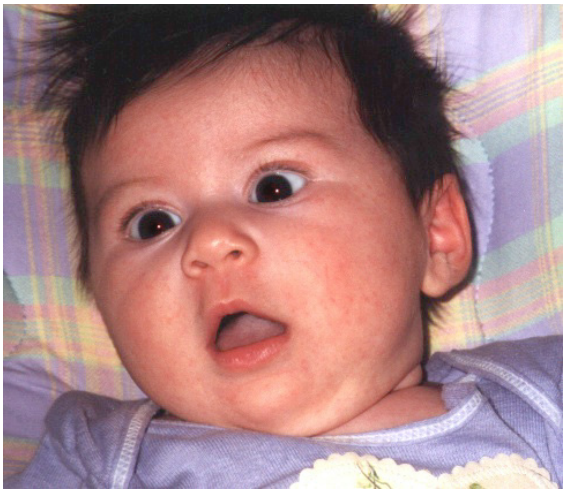


Figure 5-23: Experimental results on UCD database.





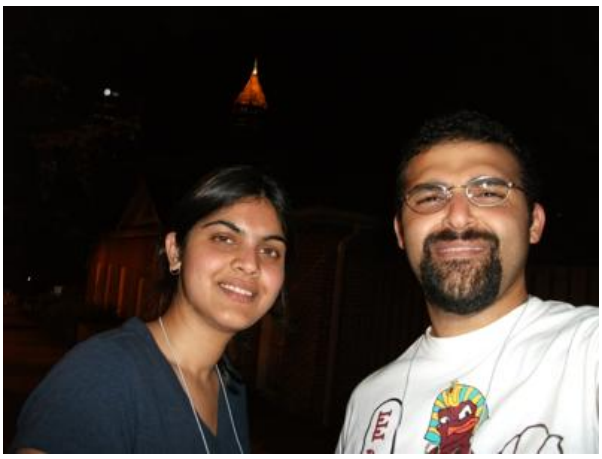
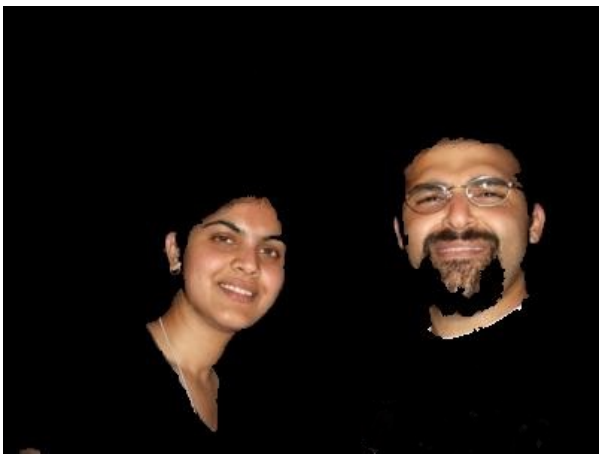



Figure 5-24: Experimental results on VT-AAST database.

5.5 Texture Segmentation

The skin detection algorithm works well when the color of the environment does not affect the color of the face. As mentioned before, the color of some objects and other things in the nature—such as the color of sand, rock, or soil—can be identical to that of the human skin. In such cases,

the skin detection algorithm may not detect the skin pixel correctly. A number of features have been used in order to distinguish between the human skin and other objects. The color of two objects may be similar; however, there are other characteristics that can be used to distinguish between the two objects. One such example is the texture of the two objects. The texture of rock differs from the texture of sand; and we can detect them easily even if they have the same color. One method for texture segmentation is to use the co-occurrence matrix (Shapiro & Stockman, 2000). Co-occurrence matrix *C* is a two dimensional array which shows the number that value *i* co-occurs with value *j*. for example *C(0,1)* looks at the right pixel and counts the occurrences of the pattern in the image (Shapiro & Stockman, 2000).

By calculating the co-occurrences matrix, the properties of the texture are computed. However, this information cannot be used to classify textures. Important information should be extracted from the co-occurrence matrix, including:

$$Energy = \sum_i \sum_j N_d^2\ (i,j)$$

$$Entropy = -\sum_i \sum_j N_d(i,j) log_2\ N_d(i,j)$$

$$Contrast = \sum_i \sum_j (i-j)^2\ N_d(i,j)$$

$$Homogeneity = \sum_i \sum_j \frac{N_d(i,j)}{1+|i-j|}$$

$$Correlation = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_d(i,j)}{\sigma_i \sigma_j}$$

$N_d(i)$ and $N_d(j)$ are defined by:

$$N_d(i) = \sum_j N_d(i,j)$$

$$N_d(j) = \sum_i N_d(i,j)$$

$$N_d(i,j) = C_d(i,j) / \sum_i \sum_j C_d(i,j)$$

In addition to these features, some other features were also selected which correspond to image color. These include the Cb, Cr, H, S components of the YCbCr and HSV color spaces. For training purposes, 20 by 20 patches where collected from different datasets, such as Textures.com and Describable Textures Dataset (DTD) (Cimpoi *et al.*, 2014) and other images from the Web. Figure 5-25 shows a number of patches from the human skin.

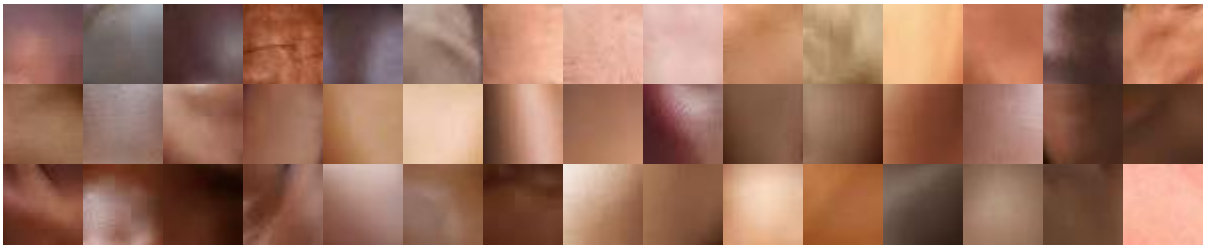


Figure 5-25: Human skin texture.

The color information Cb, Cr, H and S, combined with Energy, Entropy, Contrast, Homogeneity and Correlation from the co-occurrence matrix—built from the grey scale image—were used.

For the training, approximately 10,000 patches were used from human skin, along with 30,000 patches from objects, specifically those whose color were similar to that of human skin color. Support Vector Machine (SVM) was used for training. The classifier managed to classify the objects, as reported in Table 5-21.

For testing, total of 10,000 patches were selected from the Web and other databases; and the results are shown in Table 5-22.

	Precision	**Recall**	**Specificity**	**Accuracy**
Texture (SVM)	98.98%	99.40%	98.98%	99.30%

Table 5-21: Accuracy results for training textures samples.

	Precision	**Recall**	**Specificity**	**Accuracy**
Texture (SVM)	96.46%	95.26%	96.41%	95.83%

Table 5-22: Accuracy results for testing textures samples.

It was determined that texture segmentation technique can be used along with the skin detection. If the system chooses a part as human skin, the texture segmentation can verify the selection, or if the texture differs from human skin, the part can be rejected.

Chapter 6

Methodology – Face Detection

6.1 Methodology

The Rowley and Viola methods both search all areas of the image in order to find faces; however, in our approach, we first divide the image into two parts, the parts that contain human skin and the other parts. After this step, the search for finding human face would be restricted to those areas that just contain human skin. Therefore, face detection using color images can be faster than other approaches. As observed in (Hjelmas and Kee Low, 2001) due to the lack of standardized test, there is not a comprehensive comparative evaluation between different methods, and in case of color images, that is a bigger challenge because there are not many databases with this characteristic. Because of this problem, it is not easy to compare different methods like Viola and Rowley methods with color based methods. But face detection using color based is faster than other methods because unlike Viola and Rowley methods it does not need a window to be moved pixel by pixel on the entire image. Other works such as (Chandrappa *et al.*, 2011) have also mentioned that color based methods are faster compared to other methods.

In color images, we use the idea that we can separate the skin pixels from the other part of the image and by using some information we can recognize the face from other parts of the body. The first step in face detection is region labeling. In this case the binary image, instead of having values 0 or 1, will have value of 0 for the non-skin part and values of 1, 2… for the skin segments which were found in the previous step (Chandrappa *et al.*, 2011).

The next step that can be used is the Euler test (Chandrappa *et al.*, 2011). There are some parts in the image like the eyes, eyebrows, etc. that their colors differ from the skin. By using Euler test one can distinguish face components from other components such as the hands and arms. The Euler test counts the number of holes in each component. One main problem with the Euler test is that there may be some face components which have no holes in them and also some components belonging to hands or other parts of the body with holes in them. So Euler test cannot be a reliable method and we did not use it.

At the next step, the cross correlation between a template face and grayscale image of the original image is calculated. The height, width, orientation and the centroid of each component are calculated. The template face is also resized and rotated. The center of the template is then placed on the center of the component. The cross correlation between these two regions is then calculated. If that value is above a specified threshold, the component would be considered as a face region; otherwise it will be rejected (Chandrappa *et al.*, 2011).

We have modified this algorithm (Hajiarbabi and Agah, 2014). First, we discard the components where the proportion of the height to the width was larger than a threshold, except for some of these components which will be discussed later. In this case we were certain that no arms or legs would be considered as face. Second, the lower one fourth part of image is less probable to have

faces and so we set a higher threshold for that part, as that part of the image most likely belongs to feet and hands.

Third, for the components which contains more than one face, a method has been proposed to find the faces. This method will be covered later.

6.2 Experimental Results

Images included in Figures 6-1 to 6-5 illustrate the experimental results of our method on several images from the UCD database (UCD, 2014). The first image is the original image. The second image is produced after applying the skin detection algorithm. The third image is after filling the holes and applying the morphological operations. The fourth image shows the black and white image, as the background changes to black and the skin pixels change to white. The fifth image displays each component with a color. The sixth image illustrates placing the template on the component which has been considered as a face. The last (seventh) image is the final output of the proposed designed and implemented detection process.

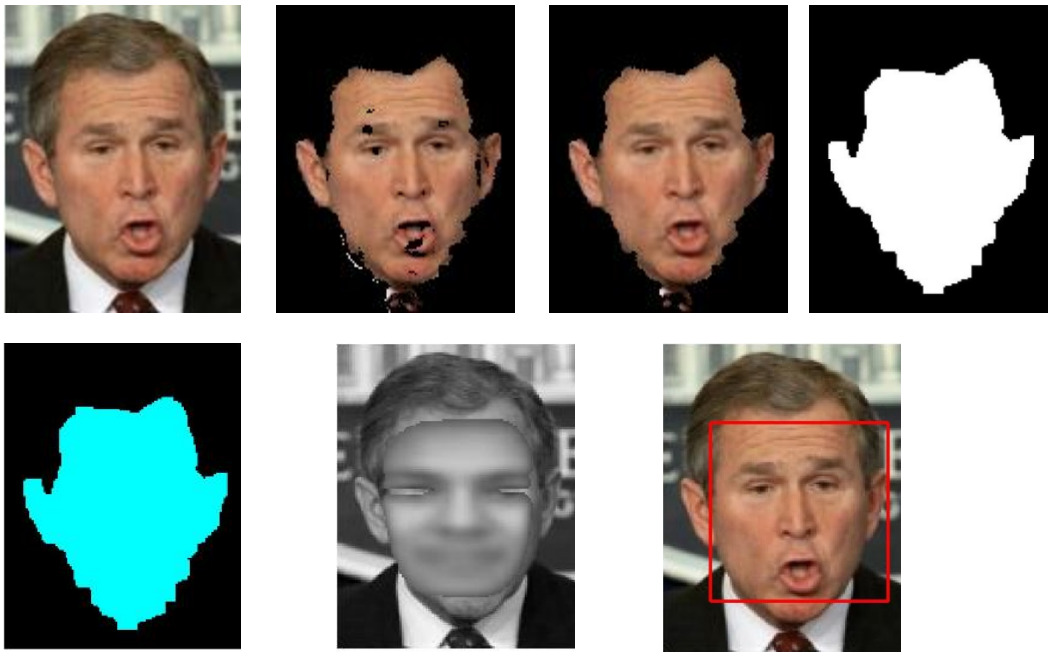


Figure 6-1: Experimental results for face detection, image containing one person (1)

Figure 6-2: Experimental results for face detection, image containing one person (2)

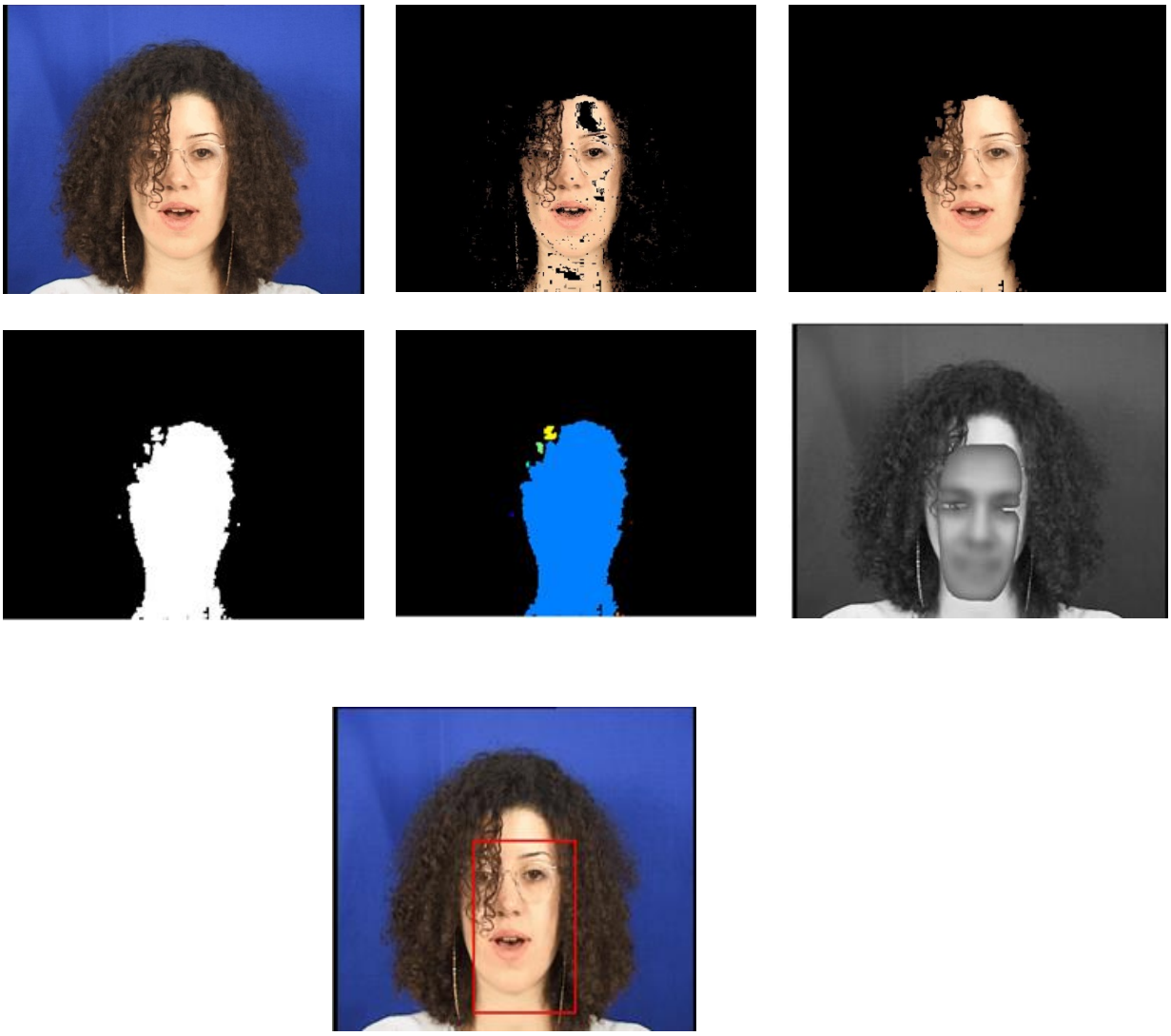


Figure 6-3: Experimental results for face detection, image containing one person (3)

Figure 6-4: Experimental results for face detection, image containing one person (4)

Figure 6-5: Experimental results, two faces with complex background

In some images there may be two or three or more faces which are so close to each other that can become one component.

The method that we have introduced to detect faces in these scenarios is as follows:

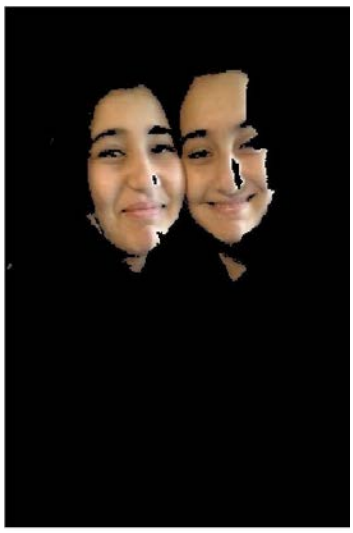
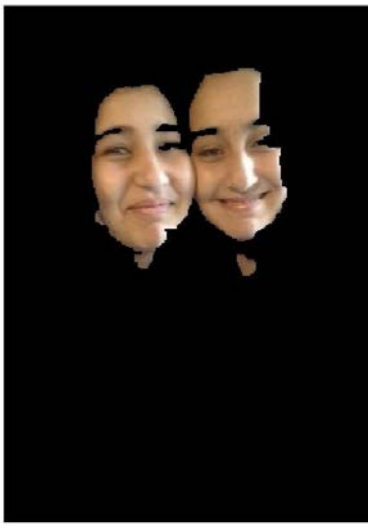
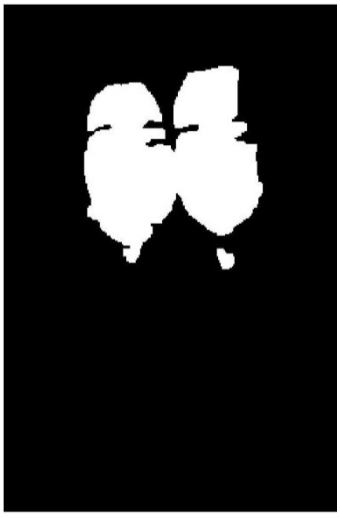
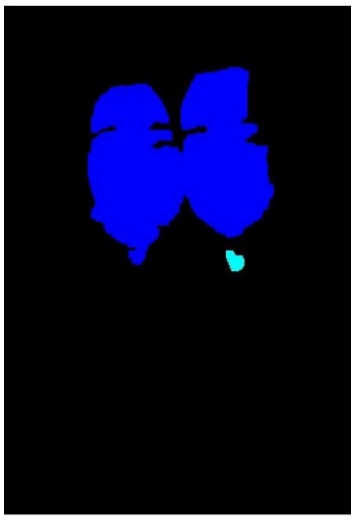
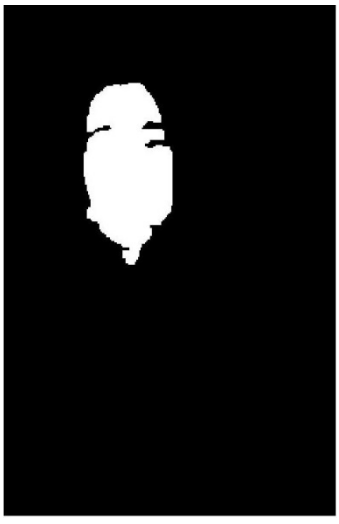
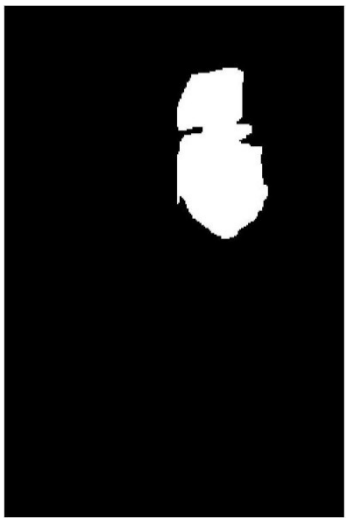
1. Compute the ratio of the height to the width of the component.
2. If the ratio is smaller than a threshold, then it means that the components may belong to more than one face. Due to the value of the ratio this component can consist of three or more faces; however, it is not probable that a component consists of more than three faces.
3. Find the border of the two faces which are combined together. For finding the border, we count the number of pixels on the horizontal axes. The two maximums (if the threshold suggests that there are two faces) are placed on the first and last third of the component, and the minimum on the second third of the component. The minimum is found, and the two new components are now tested to find faces on them. If the correlations of these two new components (or more) are more than the correlation of the single component before splitting, then it means that there were two (or more) faces in this component. This phase is done so that we can differ between two upright faces which are stuck together and a

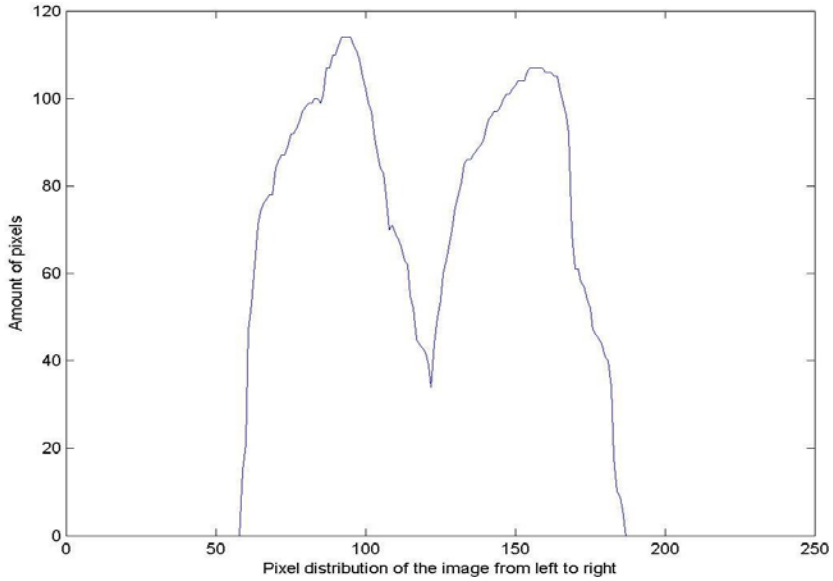
face which is rotated 90 degrees. In this case the correlation of the whole component would be more than the correlation of the two new components, so the component will not be separated.

The human face obeys the golden ratio, so the height to the width of the face is around 1.618. Sometimes with considering the neck, which is mostly visible in images, this ratio will increase to 2.4. Therefore, a height to width ratio between 0.809 and 1.2 shows that the component may belong to two faces. A value smaller than 0.809 shows that the component may consist of more than two faces. These values are calculated along the image orientation.

Figures 6-6 and 6-7 illustrate this case. Image 6-6 is not part of the UCD database. The same technique can be applied when the ratio of the height to the width is higher than a certain threshold. In this case it means that there may be two or more faces which are above each other. The same algorithm can be applied with some modification. Figure 6-8 shows this scenario. This image is not part of the UCD database. For two faces this threshold is between 3.2 and 4.

Finding the border as mentioned in earlier is faster and more accurate than using erosion (using as another method to separate two or more faces). This is because when using erosion, the process may need to be performed several times so that the two faces become separated. In addition, in erosion while separating the two faces, some other parts of the image are also being eroded.

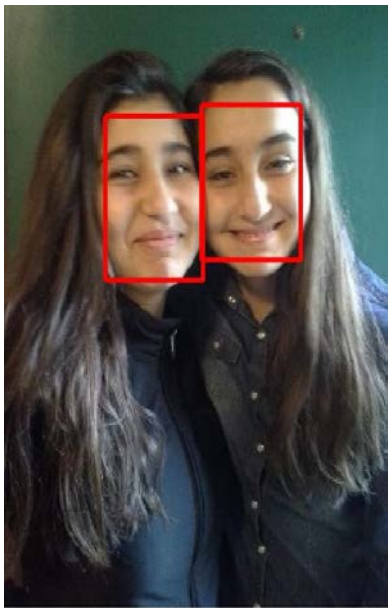
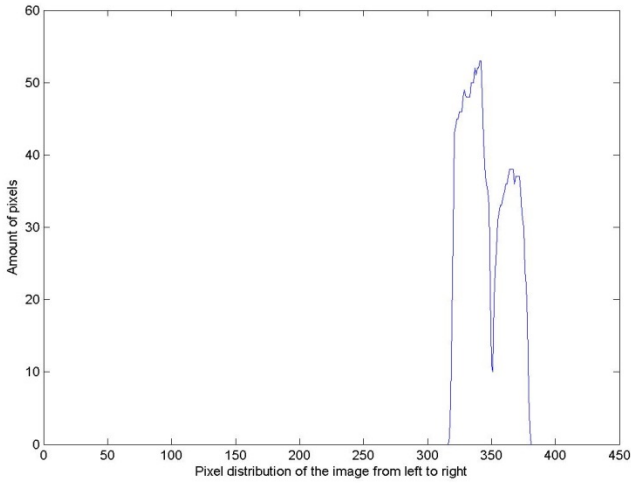


Figure 6-6: Experimental results, when two faces are close to each other


Amount of pixels
Pixel distribution of the image from left to right
0
10
20
30
40
50
60
0
50
100
150
200
250
300
350
400
450

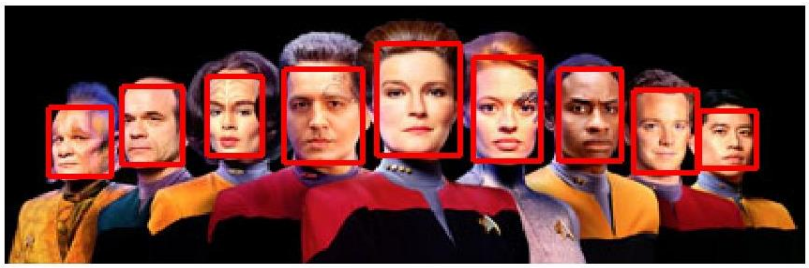
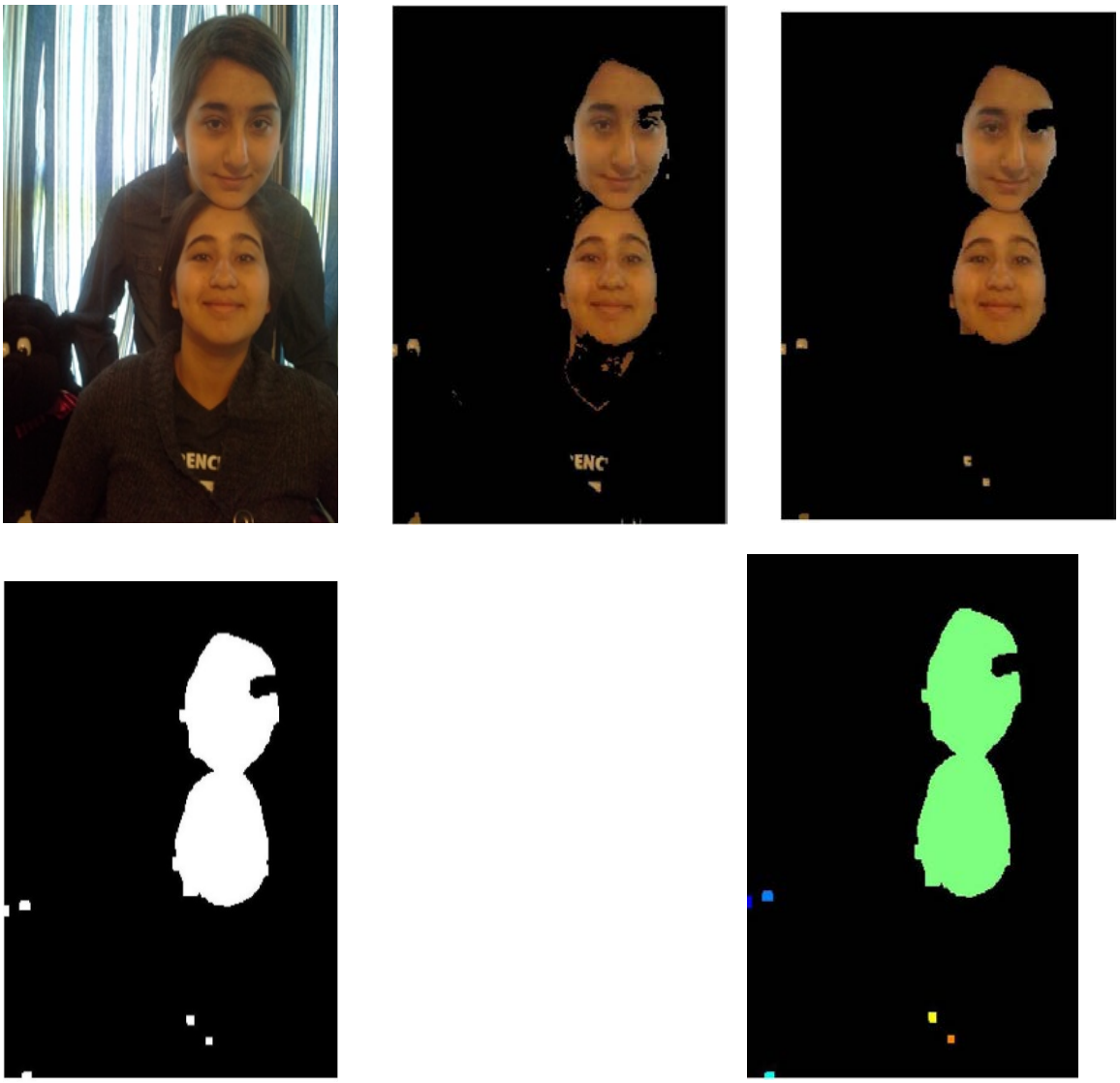
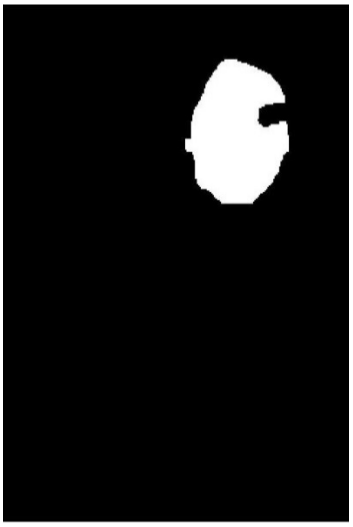
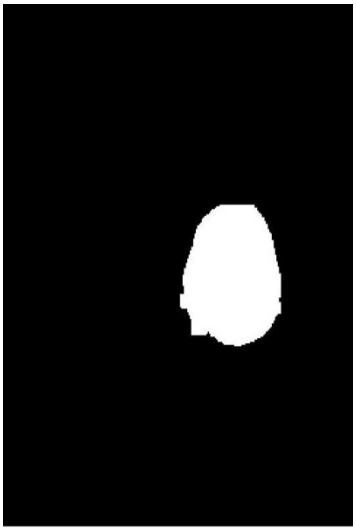
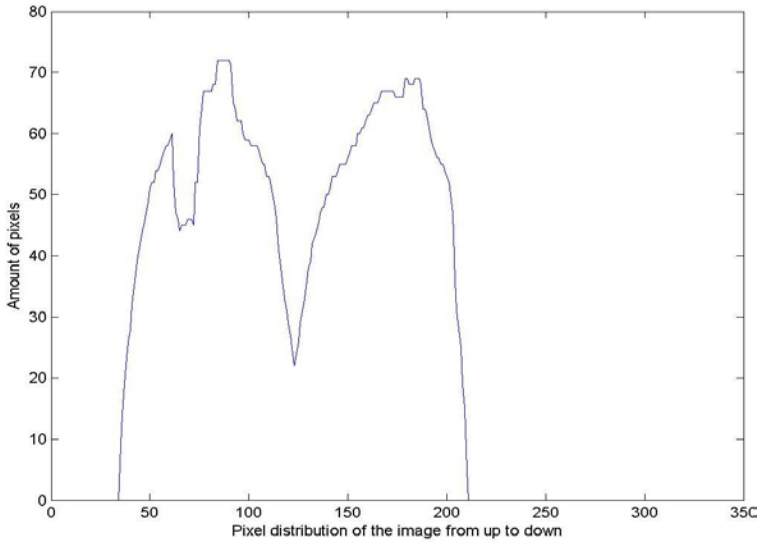


Figure 6-7: Results on UCD database, from top, the image, after applying skin detection, different components, histogram for the red component, the detected faces.








Amount of pixels
Pixel distribution of the image from up to down
0
10
20
30
40
50
60
70
80
0
50
100
150
200
250
300

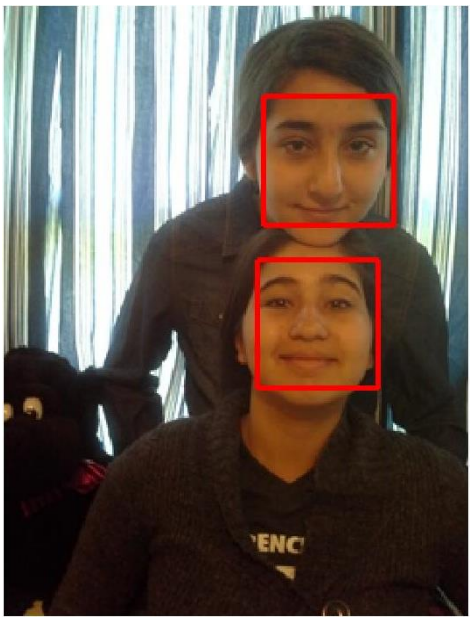
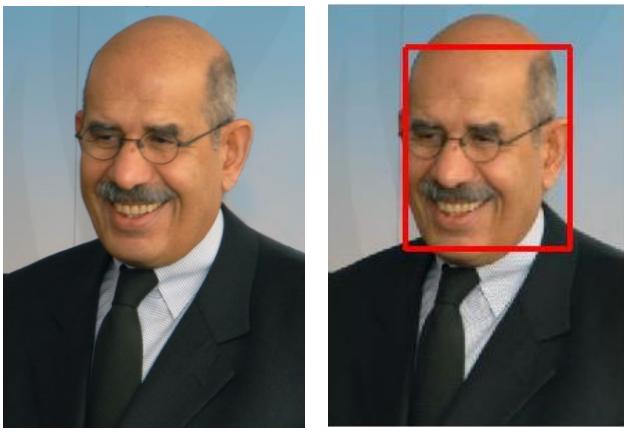


Figure 6-8: Experimental results, when two faces are vertically close to each other

Figure 6-9 shows some other images from UCD database. Figure 6-10 shows images from VT-AAST database.

Figure 6-9: Results on UCD database.



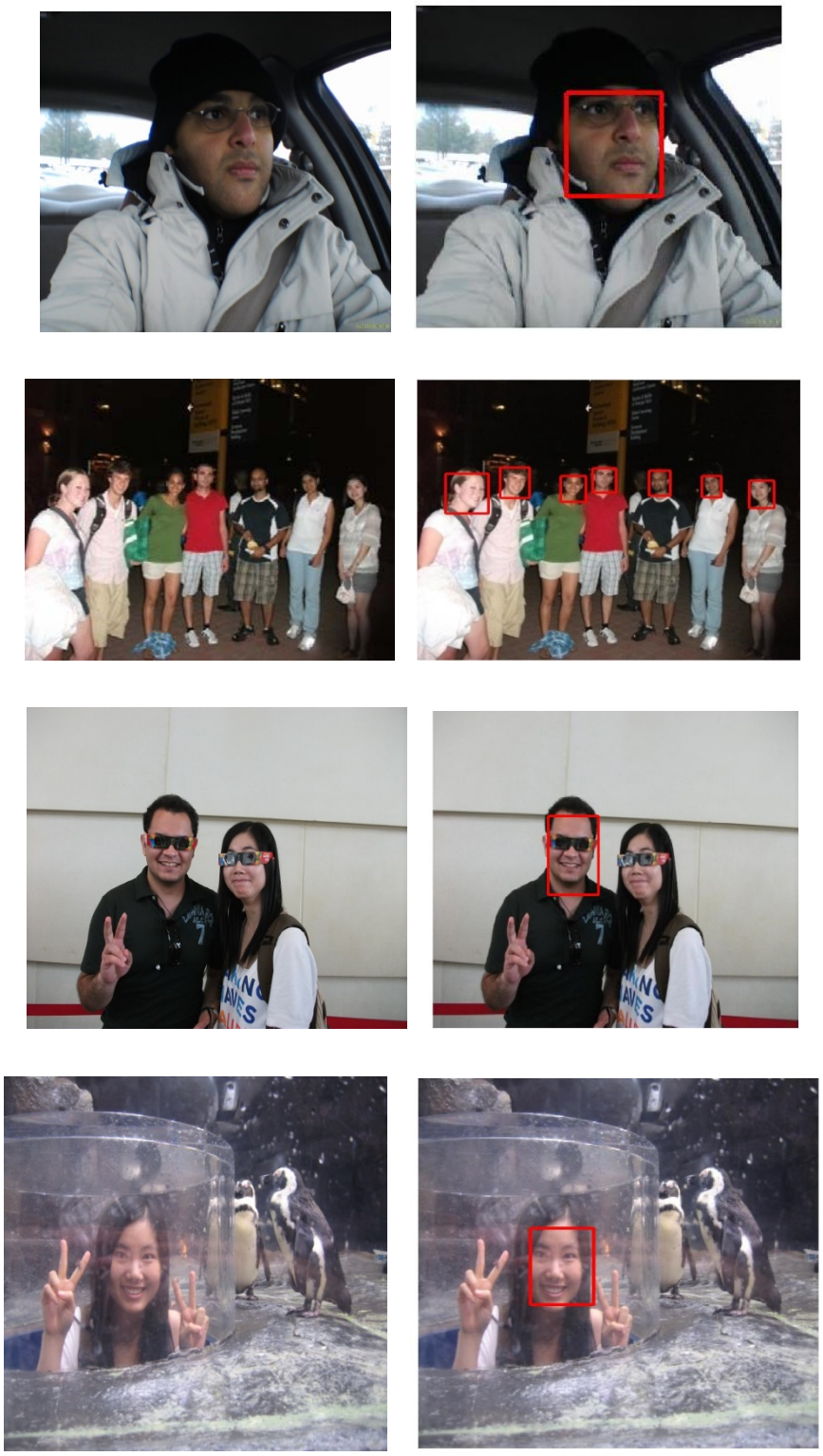


Figure 6-10: Results on the VT-AAST database.

As the results show, the proposed method has found most of the faces. This method can be fast with good results if the images are taken in an environment which is under control. The reason is that the illumination can affect the skin detection part.

As mentioned before, because the results for Rowley and Viola-Jones methods has been published for grey scale images, it is not easy to compare the results of face detection using color images with Rowley and Viola-Jones methods, but with some mathematical expressions it can be shown that face detection using color images is faster based on theoretical operations.

Consider an m*n image, for Rowley method the window size which is used to move over the image is 20*20 (Rowley *et. al.*, 1998), so the number of operations on the pixel will be:

$400*(m-19)*(n-19)$

400 shows the size of the window and $(m-19)*(n-19)$ shows the number of windows which is needed to cover the whole image. Also the image is subsampled several times, each time with the coefficient 1.2 (Rowley *et. al.*, 1998), so the total operations would be:

$400*(m-19)*(n-19) + 400(m/1.2 -19)*(n/1.2 -19) + \ldots$

Each window is applied to a receptive field neural network, which needs lots of computation to decide if the component contains a face (Rowley *et. al.*, 1998). At the worst case each pixel is processed much more than 400 times, 400 times for the original image, 333 times with probability of 83.33% in the first subsampling and etc …

For Viola-Jones method a 24*24 window is moved over the image, around 200 features with different size (s*t which s and t can be around 5 and 10) are applied to the windows (Viola *et al.*, 2001). The operations for Viola-Jones method are:

$200(m\text{-}23)*(n\text{-}23)st$

200 is the number of features which are used to be placed on the image, s*t is the average size of each feature and (*m*-23)*(*n*-23) is the number of the windows used to cover the whole image. Like Rowley method Viola-Jones method also used subsampling in order to find faces which size are less than 24*24, in Viola –Jones method the scaling is 1.25 and is done for 11 times (Viola *et al.*, 2001), so the total operations would be:

$200(m\text{-}23)(n\text{-}23)st + 200(m/1.25\text{-}23)(n/1.25\text{-}23)st + \ldots$

However, Viola-Jones method uses several techniques in order the speed up the operations. Integral image and cascade of classifiers are two techniques that are used in Viola-Jones method (Viola *et al.*, 2001). In (Viola *et al.*, 2001) it has been mentioned that Viola-Jones method is 15 times faster than Rowley method.

In face detection using color images the total operations which is needed for skin detection is 5*mn*. The reason is that each pixel is applied to five different neural networks. These amounts of operations are just for the skin detection part. The time and operation for the face detection part is not so much, because the detection part is just applied on the components. Also for face detection using color images there is no need for subsampling.

A comparison between face detection using color images and Rowley method shows that face detection using color images is at least 80 times faster, and also the neural network used for in Rowley method is much more complicated and time consuming compared to neural network which is used in face detection using color images. By comparing the speed between Viola-Jones and Rowley method it can be seen that face detection using color images is 3 to 4 times faster than Viola-Jones method.

The face detection algorithm can be summarized as follows:

1. Label the regions in the image using region labeling algorithm.

2. Calculate the height, width, orientation, and the centroid of each component (Chandrappa *et al.*, 2011).

    a. Compute the height to the width of the component.
    b. If the ratio is smaller than a threshold, then the component may belong to more than one face. The ratio shows that each component consists of a specific number of faces.
    c. If the component consists of more than one face, then find the border of the two faces which are stuck together. For finding the border count the number of pixels on the horizontal axes. The two maximums are placed on the first and last third of the component, and the minimum on the second third of the component. The minimum is found and the two new components are now tested to find faces in them. If the correlations of these two new components (or more) are larger than

the correlation of the single component before splitting, then it means that there were two (or more) faces in this component.

Face obeys the golden ratio, so the height to the width of the face is around 1.618. The same approach can be applied when the ratio of the height to the width is larger than a certain threshold. In this case it means that there may be two or more faces which are positioned above each other. The same algorithm can be applied with some modification. Also if the component consists of more than two faces, for example *n* faces, then there will be *n* peaks and *n-1* valleys.

3. Calculate the cross correlation between a template face and grayscale image of the original image (Chandrappa *et al.*, 2011).

    a. The template face is resized and rotated (Chandrappa *et al.*, 2011).

    b. The center of the template is then placed on the center of the component (Chandrappa *et al.*, 2011).

    c. The cross correlation between these two regions is then calculated (Chandrappa *et al.*, 2011).

    d. If that is above a specified threshold, the component would be considered as a face region; otherwise it will be rejected (Chandrappa *et al.*, 2011). The lower one fourth part of image is less probable to have faces and so set a higher threshold for that part, as that part of the image most likely belongs to feet and hands.

Increasing the threshold for the lower one forth part decreased the false positive of that part of the images. In some cases, the neck is also a part of the component, in order to not consider the whole neck as a part of the face, set a threshold for the height to the width. Ignore the bottom part of the rectangle that is above the threshold. With this threshold will mostly get a component with just having a face in it.

4. Still there may be some components that may have a face in them but are rejected. Sometimes in some images, the components include the face and some part of the body. In such cases, the described algorithm may not work. For this type of components other methods such as finding the place for eyes and lips can be applied.

6.3 Eye and Lip detection

For eye detection, a new rule based method is proposed (Hajiarbabi and Agah, 2016). Rule based methods are faster compared to other methods. In the eye's area of a face, there is a high contrast. Dark iris is surrounded by white sclera, which is again surrounded by black eyelashes. This fact was used in order to find suitable rules to find eyes in a color image. For this reason, first the portions of the face which are darker are selected. By experiments that was done on more than 300 images, choosing the pixels where $R<100$, $G<100$ and $B<100$ seems to produce the best results. Using these equations most portions of the face are ignored, but all the dark places in the face which do not belong to eye and that may belong to other parts such has eyebrows and eyelashes are selected.

In the next step, the components which remain are those where neighbor pixels have a saturation (S value in HSV color space) value of less than 0.12. These pixels are added to previous components. This results in the white part of the face, which is the sclera, to be added to the previously selected parts. The last step is choosing those components which have the expansion by applying the saturation rule. After this step there will remain just two or three components among the face component. The eyes components are usually larger than the other components and so can be easily detected.

In the detection of the lips, two different conditions should be considered. For those who wear lipstick, their lips are quite distinct from other portions of their face. These lips can be easily detected utilizing the HSV color space by using the equations:

$H>0.94 \ , \ 0.3<S<0.7 \ , V>0.65$

For the cases of without lipstick, the lips can be detected using the equations:

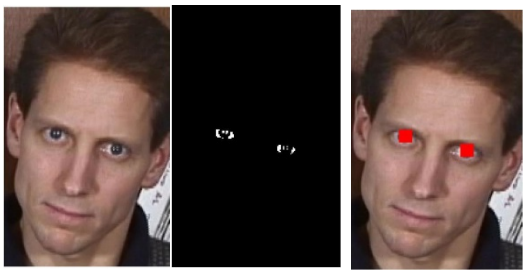
$H<0.03 \ , \ 0.3<S<0.7 \ , \ V>0.65 \ , G<110 \ , \ B<110$

These two equations are combined using the OR operation.

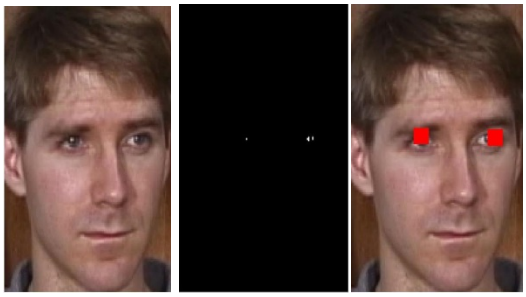
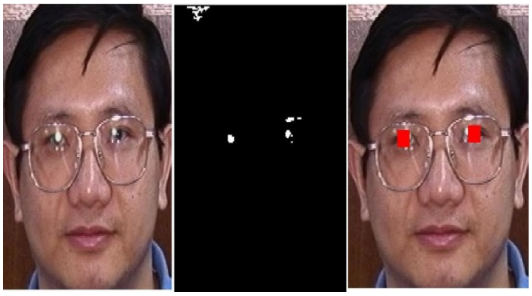
After applying these rules to the skin portion on the image, some components are found, with the largest component belonging to the lips.

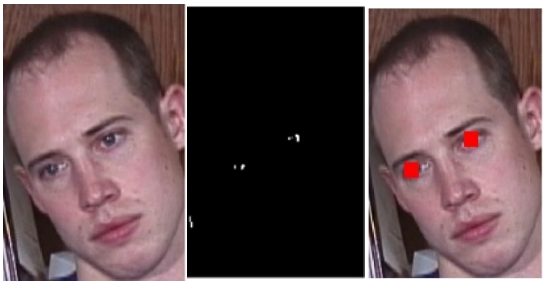
For the testing purposes, selected photos from the Georgia Tech Face database were used (2015). Tests were done on 400 images. The eyes detection rate was 86.5% for finding both eyes, 7% for finding one eye, and 6.5% where no eyes were detected. The major challenge for eyes was mainly due to some problem with skin detection. Additionally, in some images the eyes were

closed. For lip detection, the detection rate was 96.5%. Figure 6-11 shows some samples for eye detection, the first image is the real image, and the second image shows the possible places for eyes in the image. The white fragments are the parts that were added by using saturation < 0.12 to the previously selected parts. The third image shows the detected eyes. Figure 6-12 shows some samples for lip detection. The second image shows the parts that were accepted after applying the rules for finding the location of the lips. In Figures 6-13 and 6-14 some samples are shown that failed to find eyes or lips in the images.







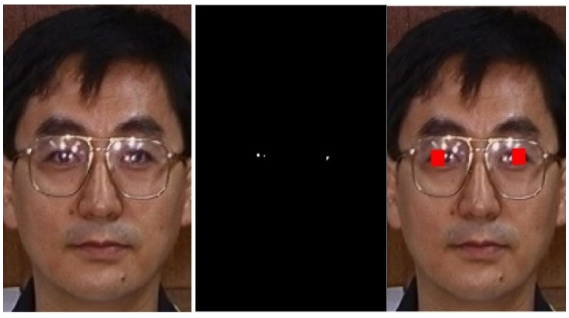
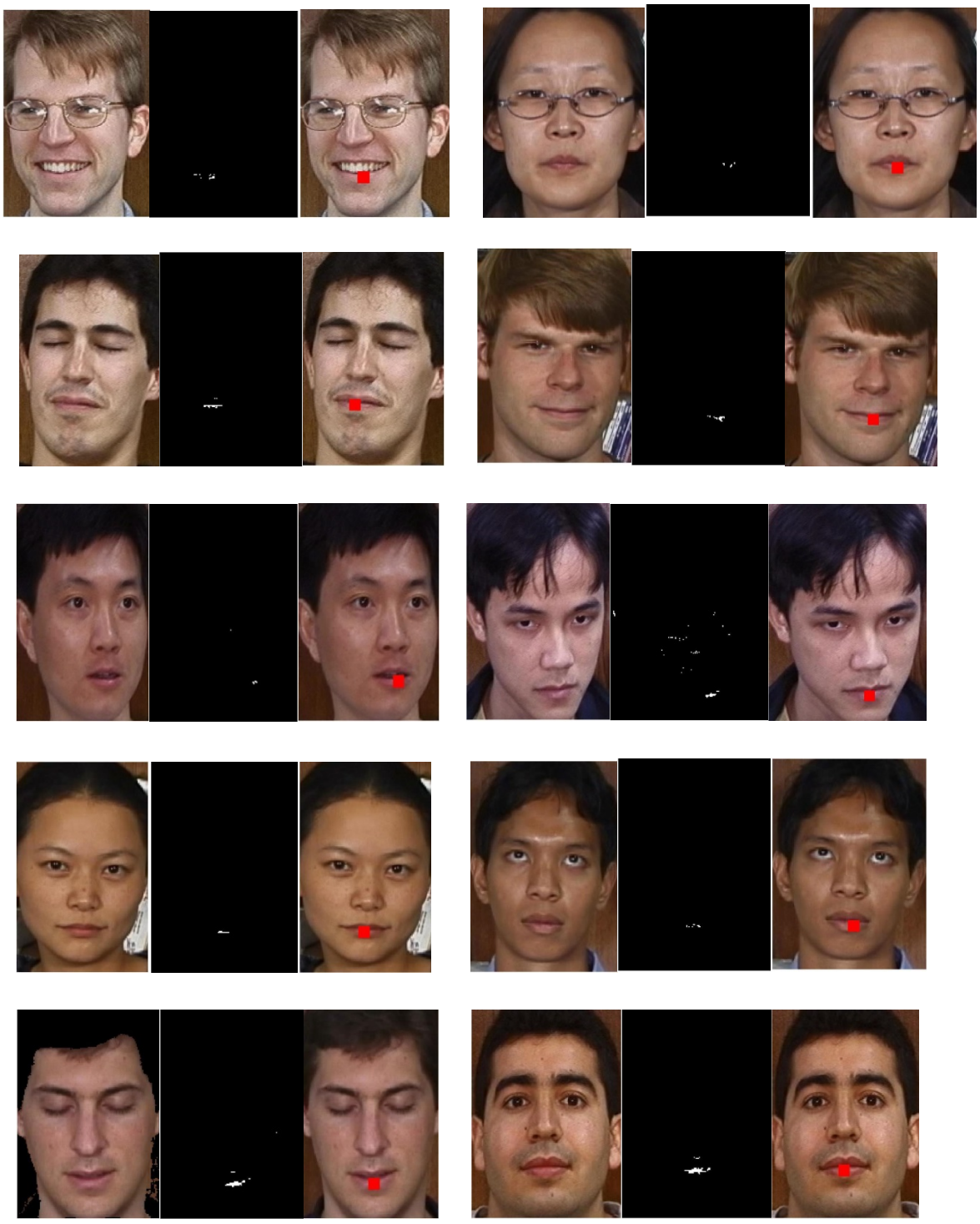


Figure 6-11: Results for eye detection from Georgia Tech database.



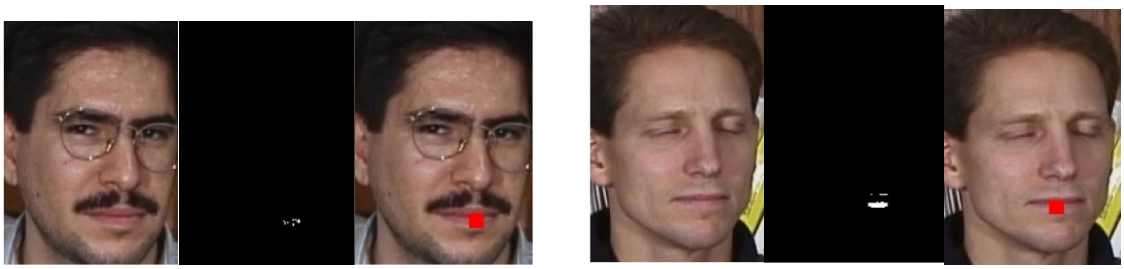


Figure 6-12: Results for lip detection from Georgia Tech database.

Figure 6-13: Images that had problems in eye detection from Georgia Tech database.

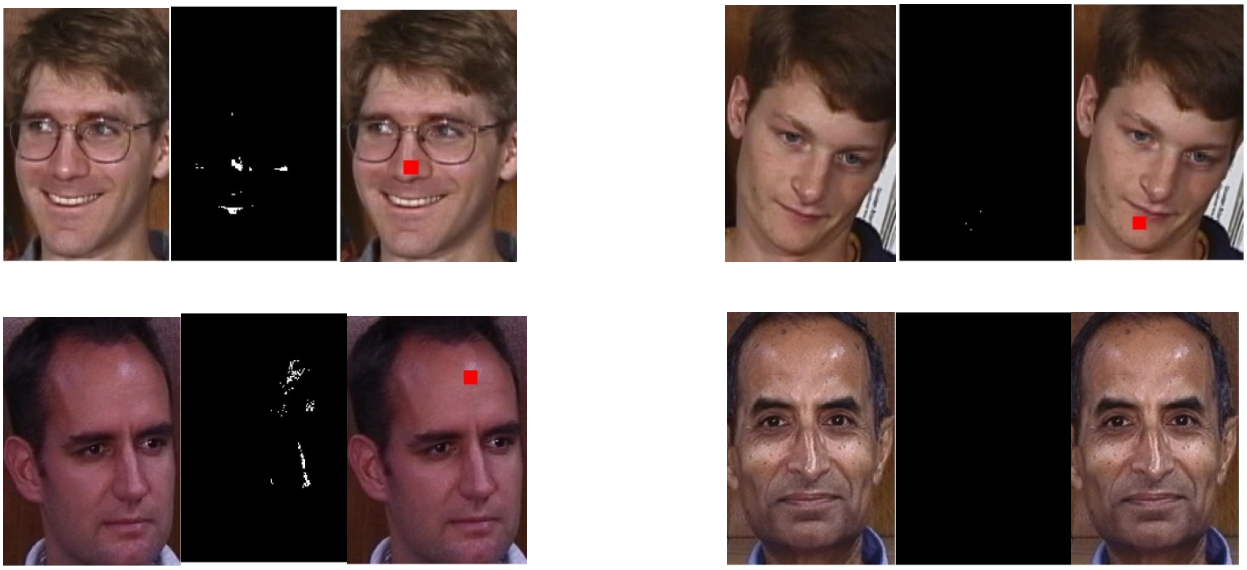
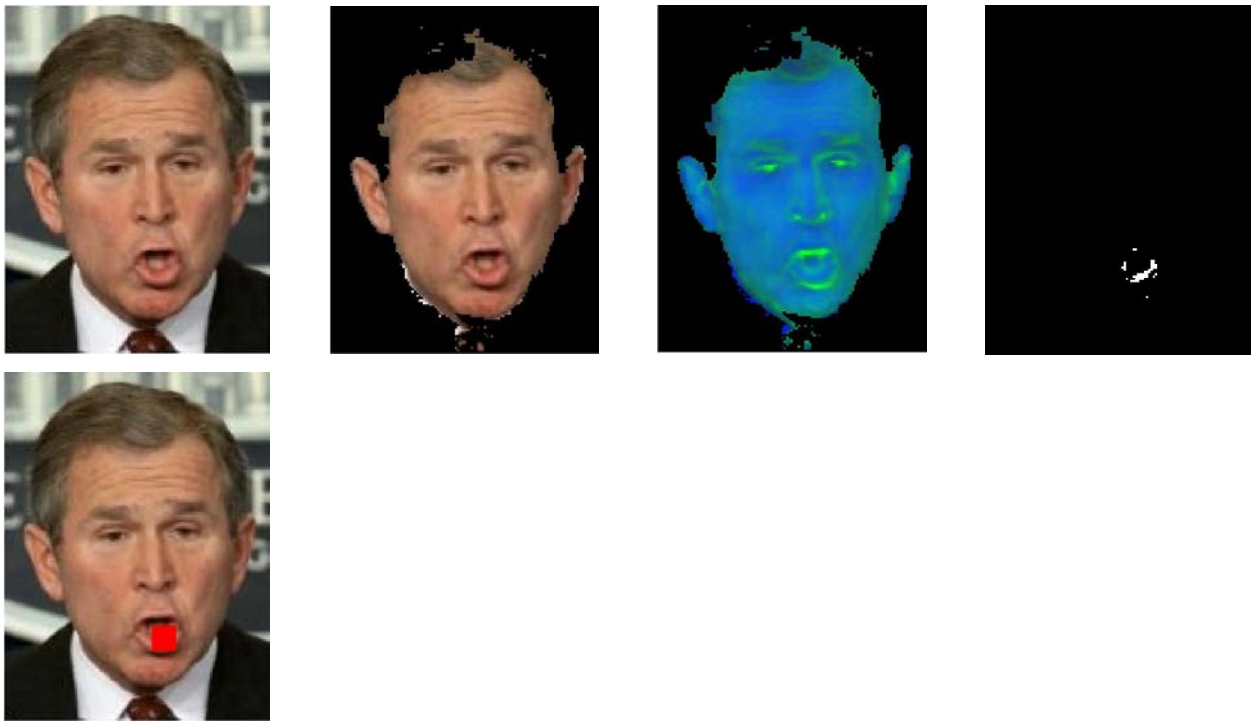


Figure 6-14: Images that had problems in lip detection from Georgia Tech database.

Figure 6-15 shows the results for lip detection using the UCD database. The first image is the real image, the second image is after applying the face detection algorithm, third image shows the detected skin using HSV color space, forth image shows the possible place for lips, and fifth image shows the lips that are detected.



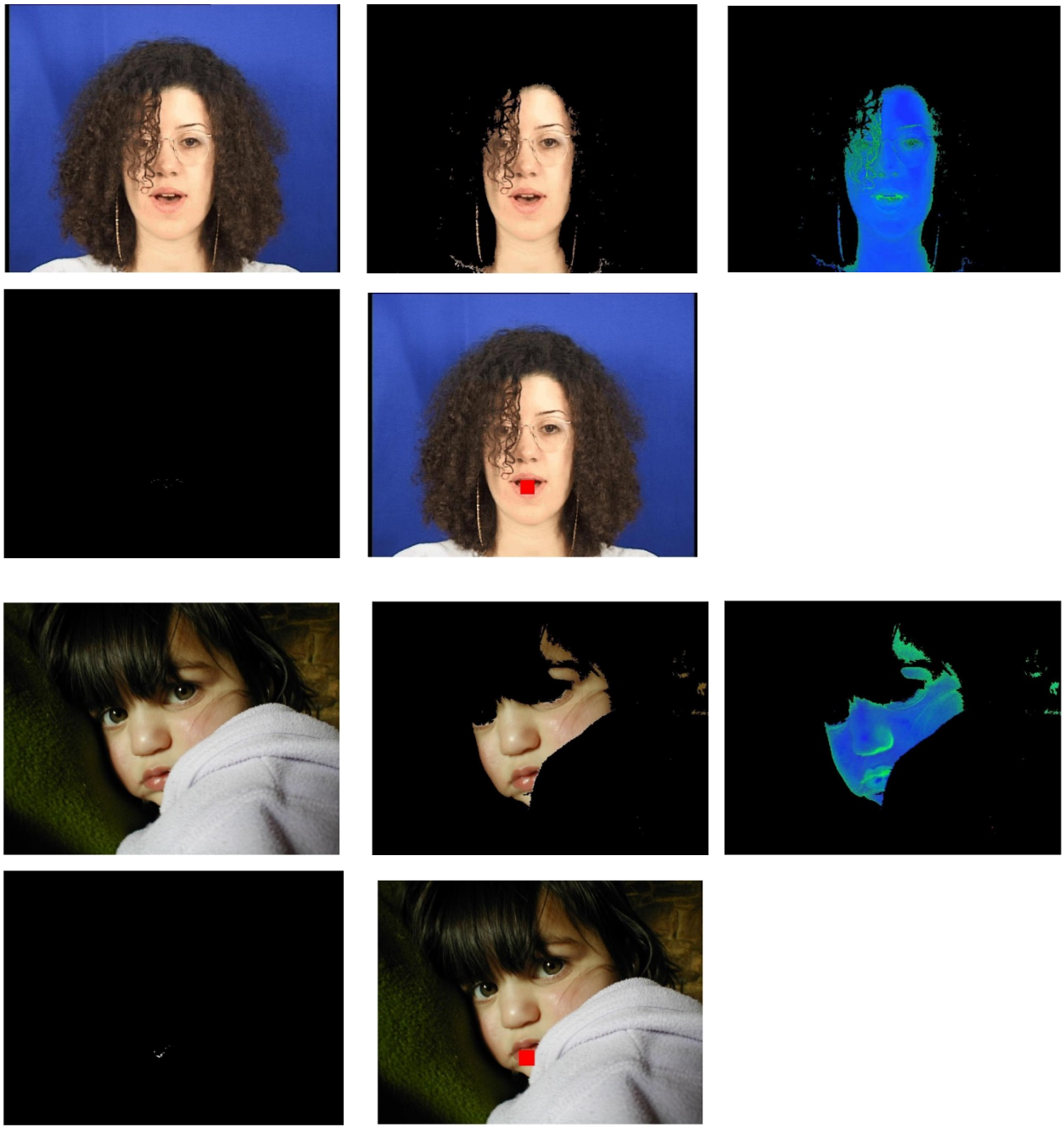
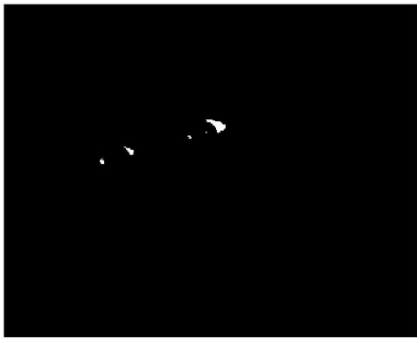


Figure 6-15: Results for lips detection on the UCD database.

Figure 6-16 shows the result of applying the eye detection algorithm to the UCD and VT-AAST databases. The first image shows the real image, the second image is after applying the skin

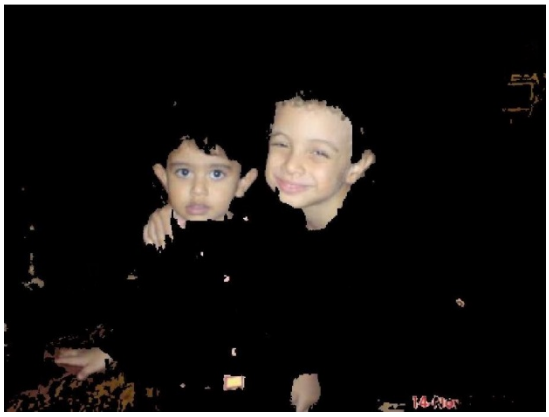
detection algorithm, the third image shows the parts of the skin where R<100, G<100, and B<100, the forth image shows the possible places for the eyes, and the fifth image shows the detected eyes.

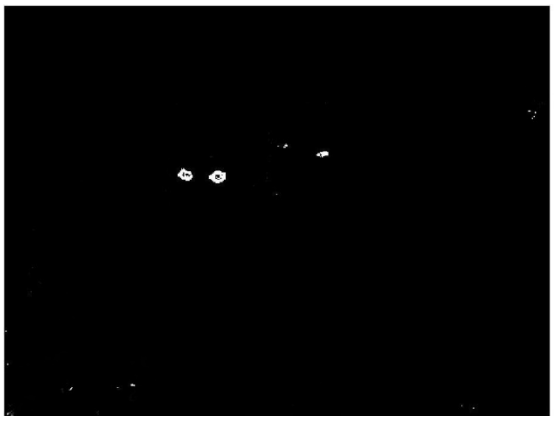


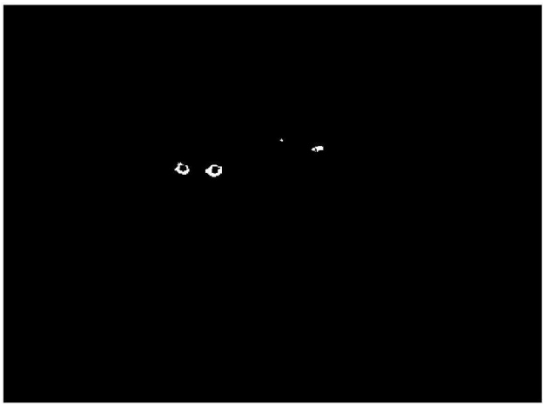






14-Nov-08 4:05 pm







14-Nov-08 4:05 pm

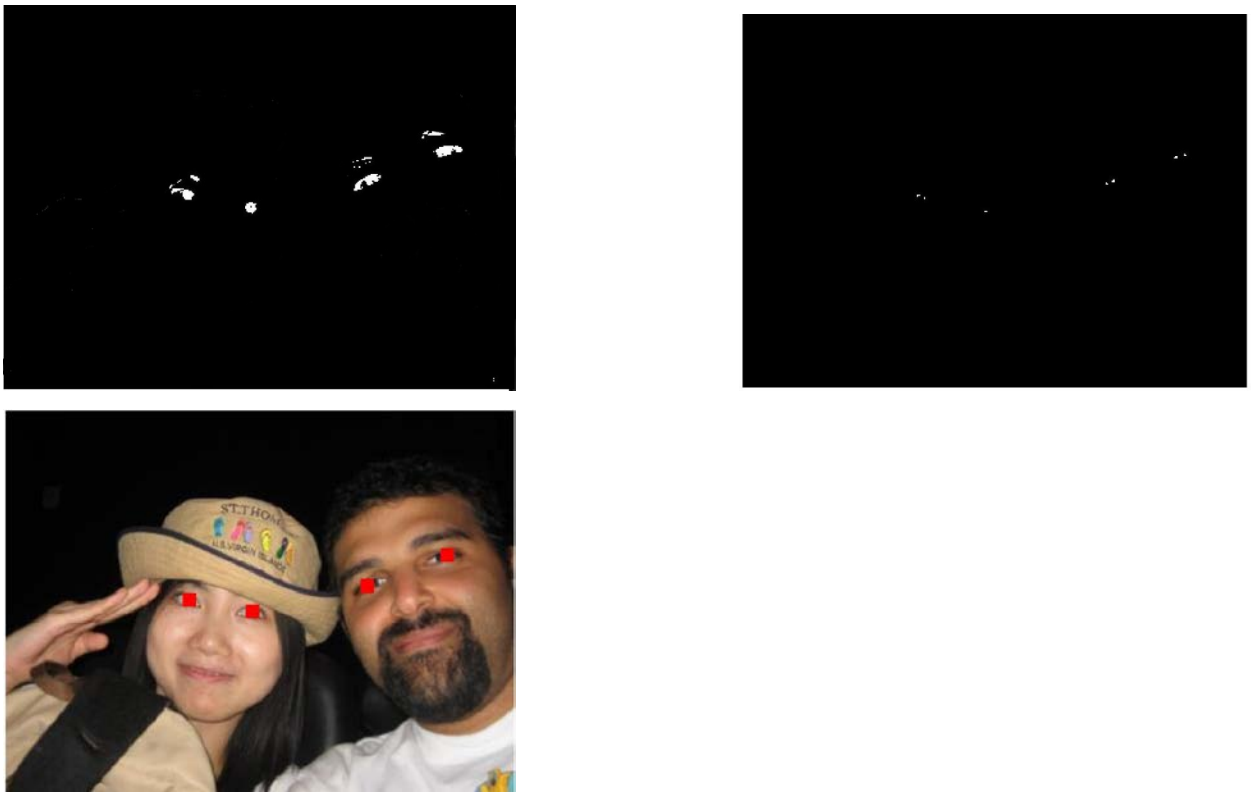


Figure 6-16: Results for lips detection on the UCD and VT-AAST database.

As mentioned previously, another method to find faces in an image is to use the location of the eyes and lips in the image. Figure 6-17 shows an image from the Georgia Tech database. The triangle between the eyes and lips shows the place of the face in the image. In order to draw a rectangle around the face, the distance between the eyes and lips can be used as a measure of the rectangle. The face width and length are usually two and two and half times of this distance, respectively.

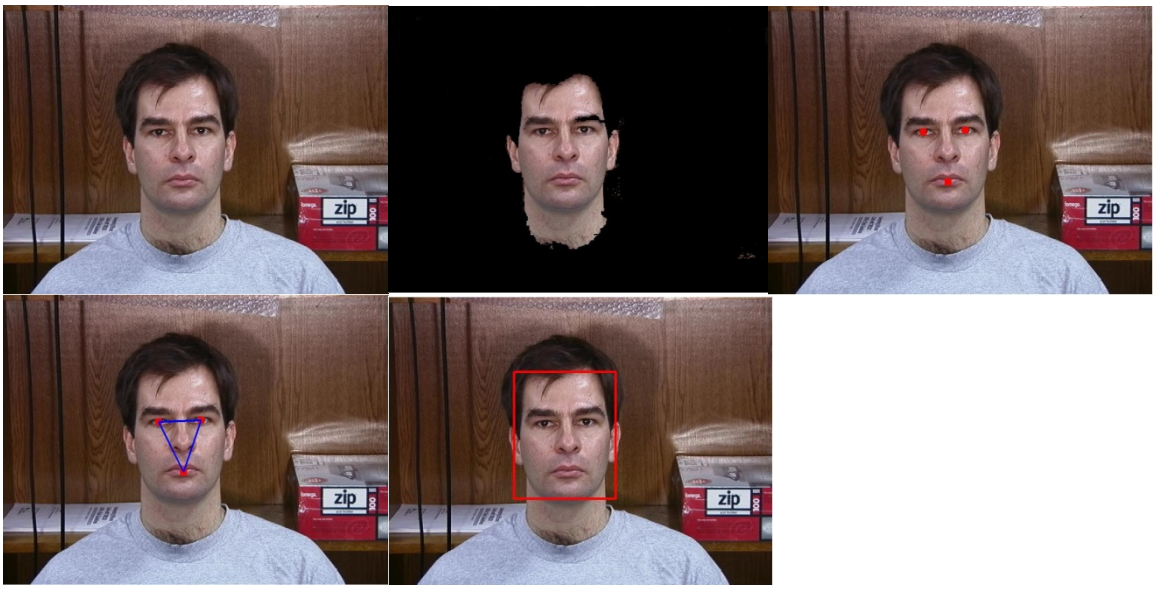


Figure 6-17: Results for face detection on the Georgia Tech database.

Chapter 7

Methodology - Face Recognition

7.1 Databases

In this phase the face recognition algorithms discussed previously are implemented. Two popular databases are used. One of them is the ORL database (ORL, 2014). The images for this database have been taken between April 1992 and April 1994 in the Olivetti Research Laboratory (ORL) in Cambridge University. In this database 400 images from 40 people are included, with 10 images for each person. Some of these images have been taken during different periods of time. There are differences between images due to open and closed eyes, smiling, with and without glasses, and scaling up to 10%. All the images have been taken using a black background and rotation of the head up to 20 degrees. Figure 7-1 shows all the images from the ORL database, and Figure 7-2 show an image of one person. In the implementation phase, five images have been used for training and five have been used for testing.

Another database that has been used is the Sheffield database (Sheffield, 2014). This database, from the University of Manchester, consists of 575 images which belong to 20 people. Images vary from front view to profile. 10 images were chosen for training and the rest for testing. Figure 7-3 shows an image of one person.

Each algorithm was run several times and with using (Er *et al.*, 2002):

$$E_{ave} = \frac{\sum_{i=1}^{m} E_{NM}(i)}{mN_t}$$

The accuracies of the algorithms were calculated. Where m is the number of runs, E_{NM} is the error in classifying, and N_t is the number of the images used for testing in each run.

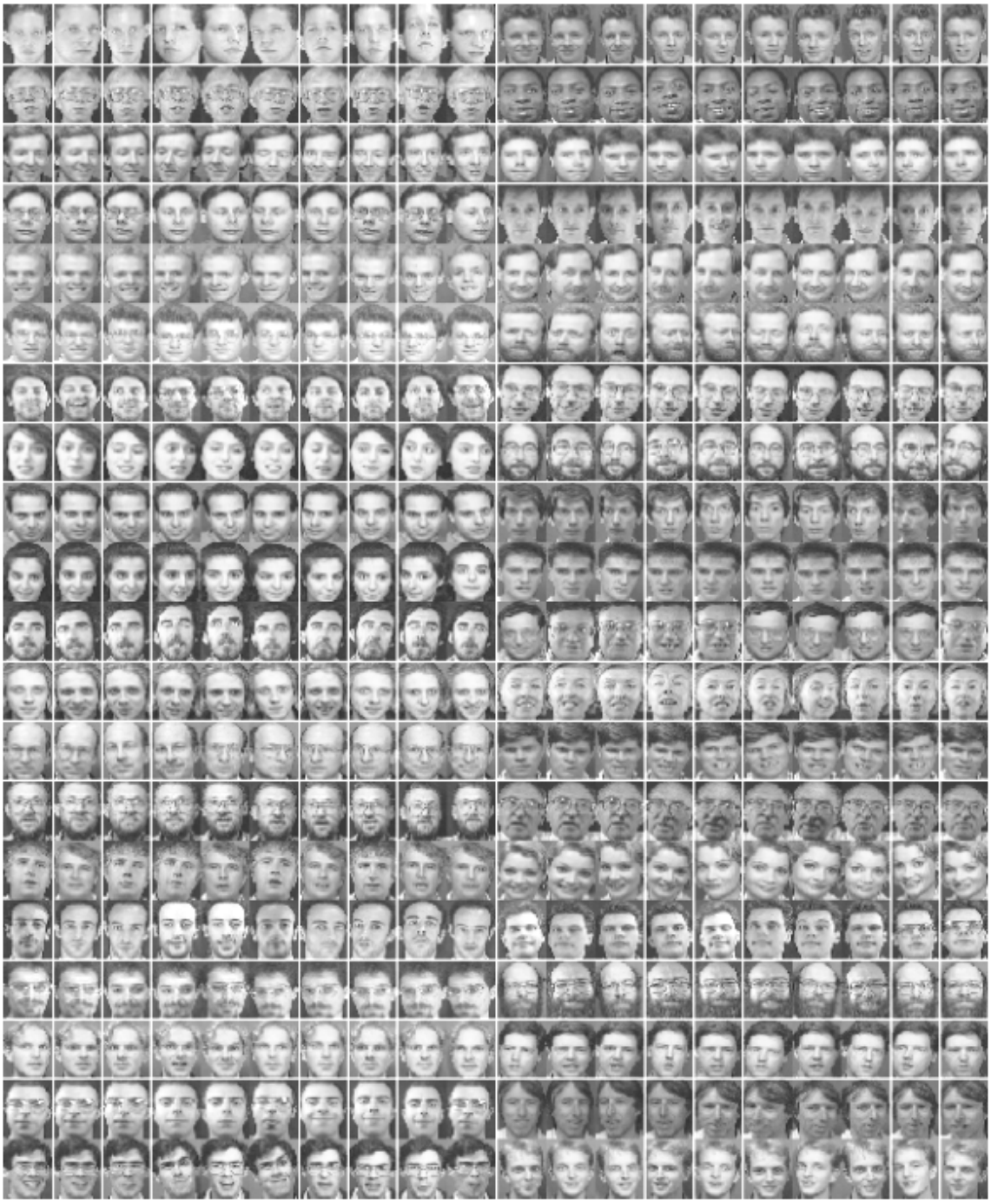


Figure 7-1: ORL database (ORL, 2014)

Figure 7-2: Images belonging to a person from ORL database (ORL, 2014)

Figure 7-3: Images belonging to a person from Sheffield database (Sheffield, 2014)

There are multiple ways for classifying. The distance based methods and the neural network based methods are more common than the other methods. The most important distance based classifiers are city block distance, Euclidean distance and Mahalanobis distance. We used the three nearest neighbor in this case.

For classification, a distance measure and also RBF neural network is used in order to compare their classification performance. As a distance measure Euclidean distance is selected. RBF

neural network is a powerful classification method for pattern recognition problems. It does not have the drawbacks of back propagation neural networks; and the training is much faster.

With $P \in \Re^r$ denoting the input vector and $C_i \in \Re^r \ (1 \le i \le u)$ being the prototype of the input vectors, the output of each RBF units is as follows:

$$R_i(P) = \exp \frac{-\|P - C_i\|^2}{\sigma_i^2}$$

Where σ_i is the width of the $i\,th$ RBF unit. The $j\,th$ output $y_j(P)$ of an RBF neural network is

$$y_i(p) = \sum_{i=1}^{u} R_i(P) * w(j,i) \ + \ w(j,0)$$

Where $R_0 = 1$, $w(j,i)$ is the weight of the $i\,th$ receptive field to the $j\,th$ output. The weights of the first layer are all equal to one. The number of nodes in the second layer at first equals to the number of classes. Whenever two classes have intersection with each other, then a node is added to the second layer and the class is split into two subclasses.

7.2 Linear Algorithms Results

To establish a base line, beginning the linear algorithms are used. Matlab was used for the simulation. For neural network the network inputs are equal to the features vector's dimensions. For output two methods can be used. The first one is the bit method in which the class number is shown by using bits. Each output neuron is equivalent to one bit. For example, 000110 shows class 6 and 001001 shows class 9. The output of an RBF network is a real number between 0 and 1. The other method is considering a neuron for each class. If there are 40 classes, then there are

also 40 nodes in the output layer. The second method produced better results and in all simulations the second method has been used. But if there are a large number of classes, then maybe the first method is better. Also a neuron can be considered for images that do not belong to any classes.

It should be noted that the two other important neural networks classifiers, back propagation neural network and probabilistic neural network, have lower performances than RBF neural networks. Back propagation neural network needs significant time for training compared to the RBF neural network. The memory needed for back propagation neural network is also much larger than the RBF neural network. The experiment also shows that the results using back propagation neural network is of lesser quality than RBF neural network. Probabilistic neural network performance is equivalent to distance based classifiers performance.

For linear methods, principal component analysis, linear discriminant analysis, fuzzy linear discriminant analysis and multiple exemplar linear discriminant analysis have been chosen. Figure 7-4 shows the mean of the training images. Figure 7-5 shows 10 Eigen faces from ORL database and Figure 7-6 shows 10 fisher faces from ORL database. Results are shown based on the number of extracted features.

Figure 7-4: The mean of the ORL images

Figure 7-5: Top from left, first to fifth Eigen faces, second row from left, 10th, 30th, 50th, 70th, 90th Eigen faces from ORL database

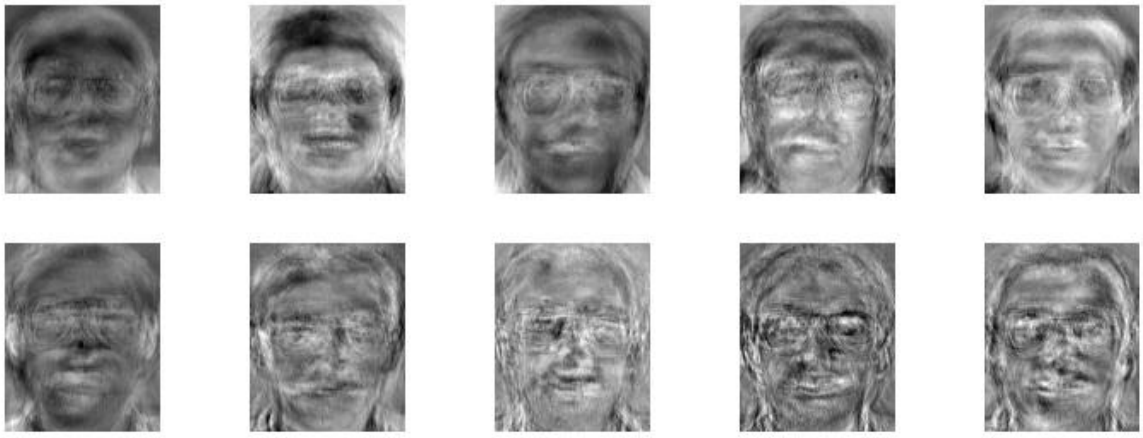


Figure 7-6: Top from left, first to fifth Fisher faces, second row from left, 10th, 30th, 50th, 70th, 90th Fisher faces from ORL database

Figures 7-7 to 7-12 compare the linear algorithms using RBF neural networks and Euclidean, city block distance as a classifier. PCA has been used as a dimension reduction method before all the algorithms. The images show the recognition rate on the numbers of features. As the results illustrate the multiple exemplar discriminant analysis method has better performance compared to other methods. Also RBF neural network shows to be a better classifier compared to other classifiers.

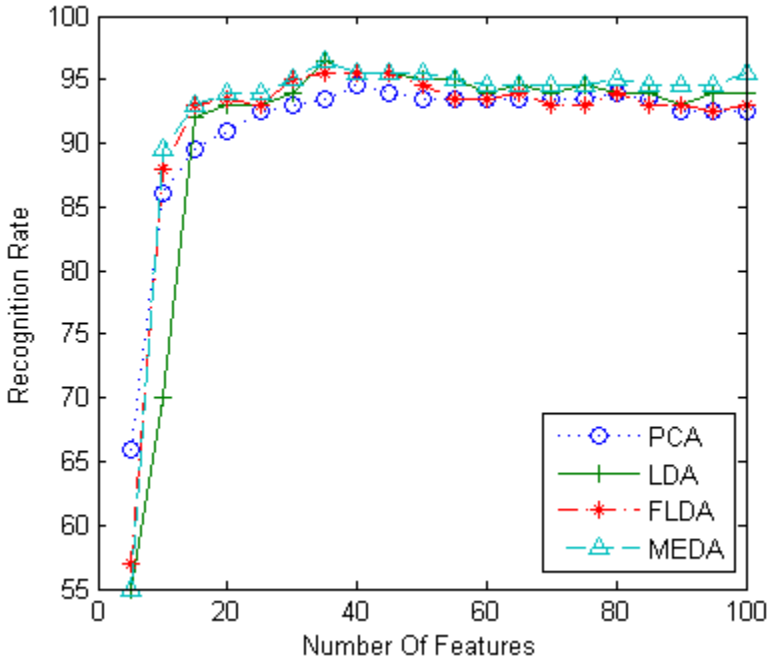


Figure 7-7: The linear algorithms applied on ORL database using RBF neural network as a classifier

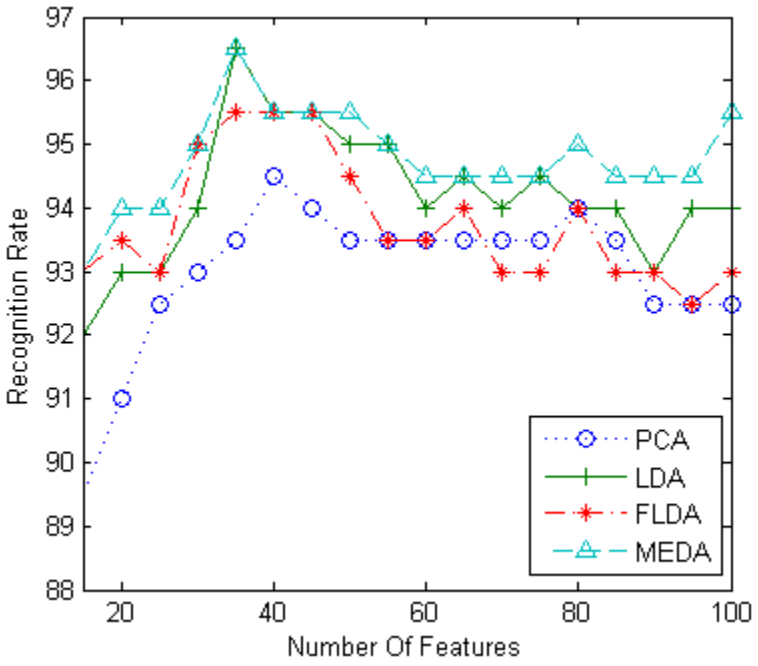


Figure 7-8: The linear algorithms applied on ORL database using RBF neural network as a classifier, figure 7-7 from a closer view

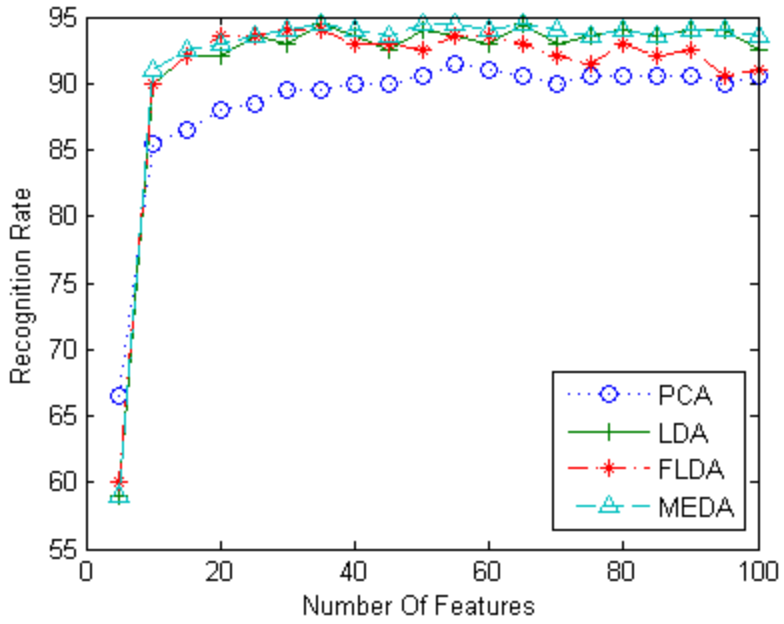


Figure 7-9: The linear algorithms applied on ORL database using Euclidean distance as a classifier

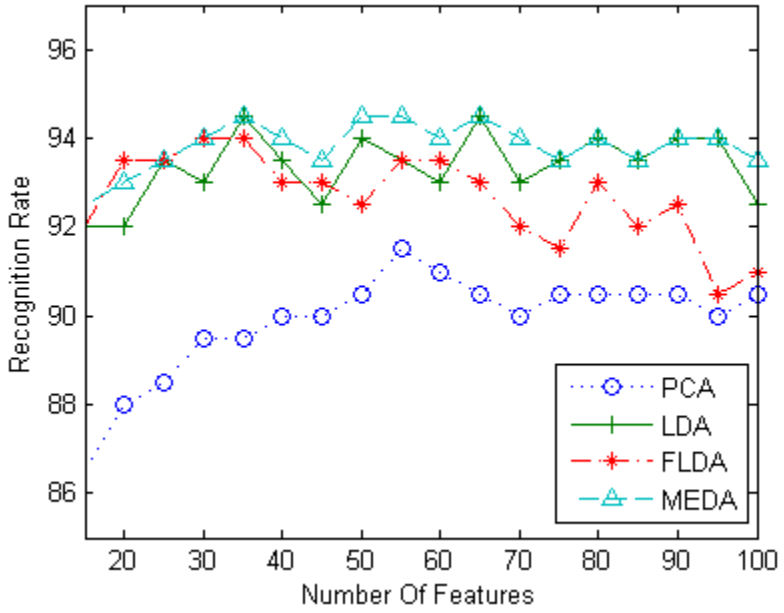


Figure 7-10: The linear algorithms applied on ORL database using Euclidean distance as a classifier, figure 7-9 from a closer view

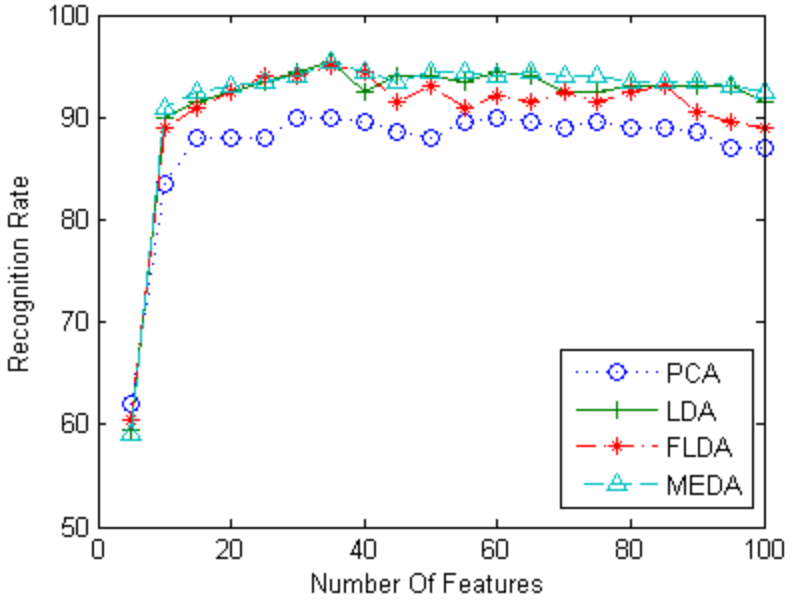


Figure 7-11: The linear algorithms applied on ORL database using city block distance as a classifier

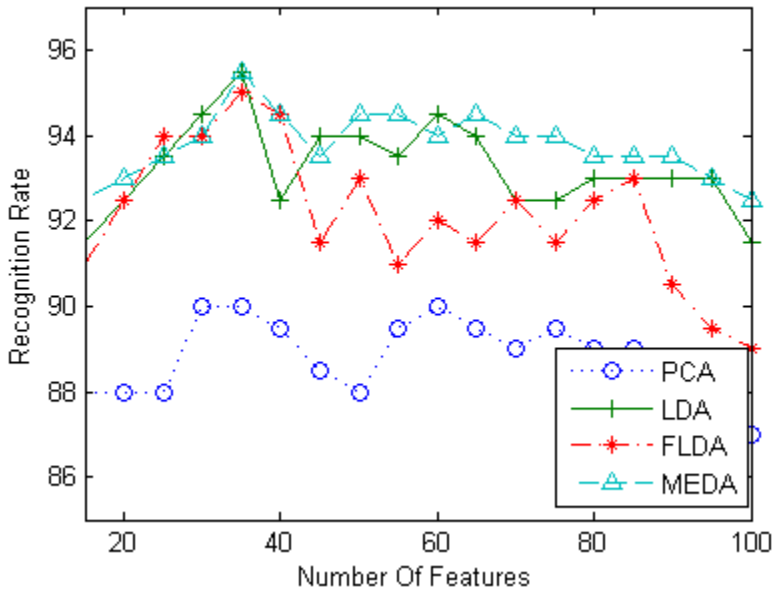


Figure 7-12: The linear algorithms applied on ORL database using city block distance as a classifier, figure 7-11 from a closer view

Tables 7-1 and 7-2 show the run time of algorithms when using 50 features. The training time is the time for loading an image into memory, applying the algorithm to it, and when using RBF neural network as a classifier, the time needed to train the network. The testing time is the time of classifying 200 testing images.

	Training Time (ondssec)	Testing Time (ondssec)	Average Time for recognizing one image (seconds)
Eigen face	37.2536	14.5009	0.0725
Fisher Face	37.7945	16.0130	0.0801
Fuzzy Fisher Face	38.7758	17.4251	0.0871
Multiple Exemplar Discriminat Analysis	38.5554	16.5037	0.0825

Table 7-1: The run time of linear algorithms using RBF neural networks on ORL database

	Training Time (ondssec)	Testing Time (ondssec)	Average Time for recognizing one image (seconds)
Eigen face	25.9273	12.5981	0.0630
Fisher Face	26.0475	14.2805	0.0714
Fuzzy Fisher Face	27.3922	14.6811	0.0734
Multiple Exemplar Discriminat Analysis	26.4065	14.5810	0.0729

Table 7-2: The run time of linear algorithms using Euclidean distance on ORL database

The tables show that although the Euclidean distance is faster than RBF neural network, RBF neural network performs better as a classifier compared to the Euclidean distance.

The same experiments have been performed using the Sheffield database (Sheffield, 2014). Figure 7-13 displays the mean images of the Sheffield database and Figures 7-14 and 7-15 show the Eigen faces and Fisher faces of the Sheffield database.

Figure 7-13: The mean of the Sheffield images

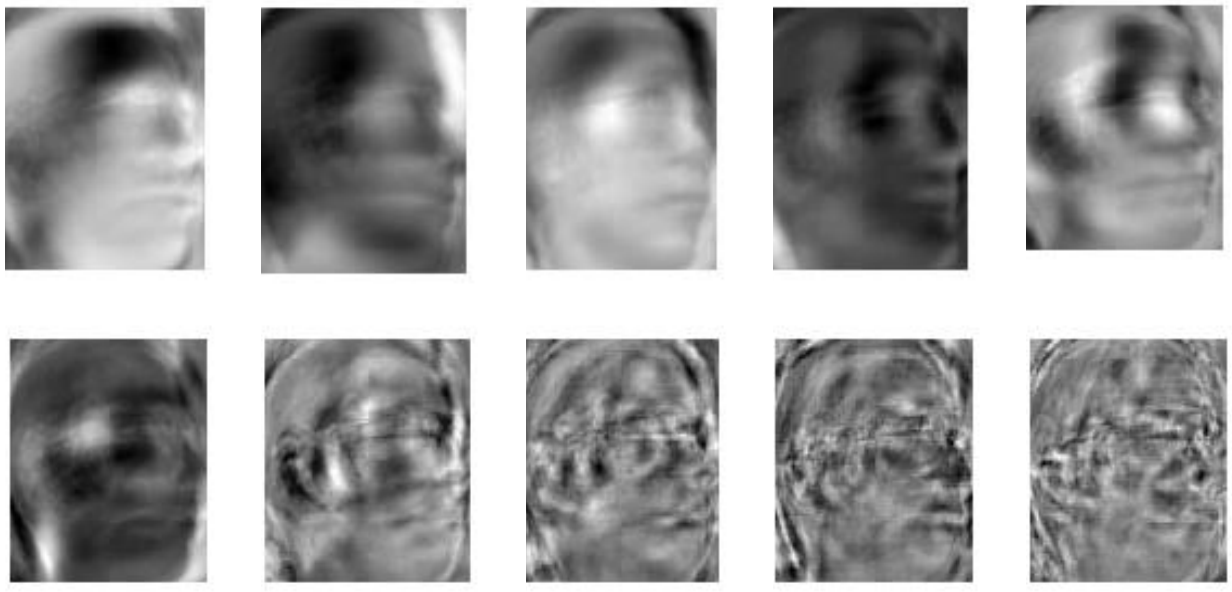


Figure 7-14: Top from left, first to fifth Eigen faces, second row from left, 10^{th}, 30^{th}, 50^{th}, 70^{th}, 90^{th} Eigen faces from Sheffield database

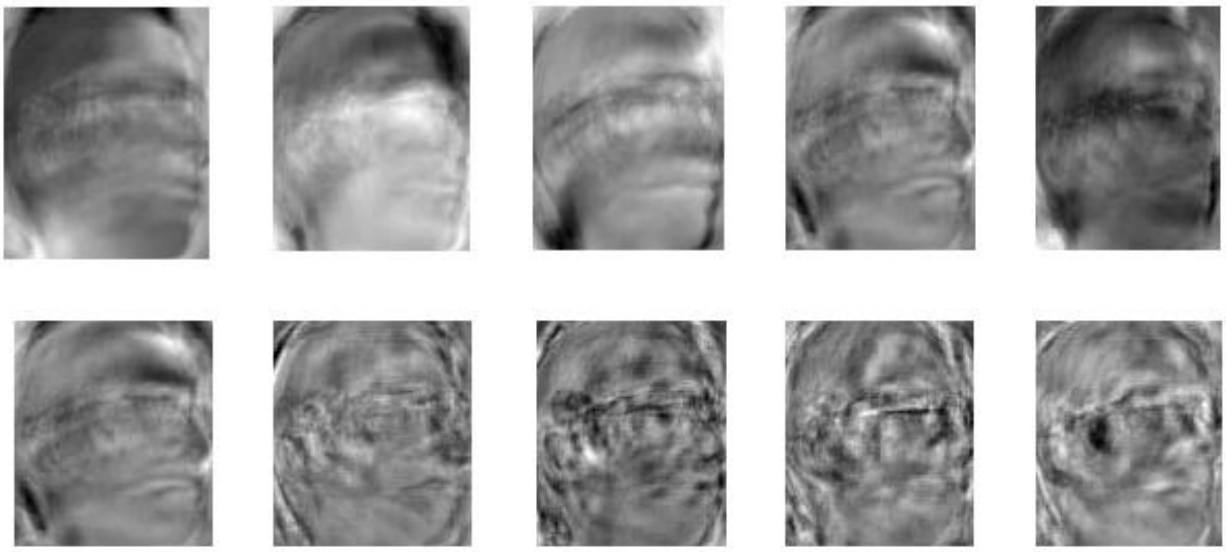


Figure 7-15: Top from left, first to fifth Fisher faces, second row from left, 10^{th}, 30^{th}, 50^{th}, 70^{th}, 90^{th} Eigen faces from Sheffield database

Figures 7-16 and 7-17 compare the linear algorithms on Sheffield database using the RBF neural network and Euclidean distance. On Sheffield database, similar to the ORL database, the method Multiple Exemplar Discriminant Analysis had better results when using RBF neural network as a classifier compared to other results.

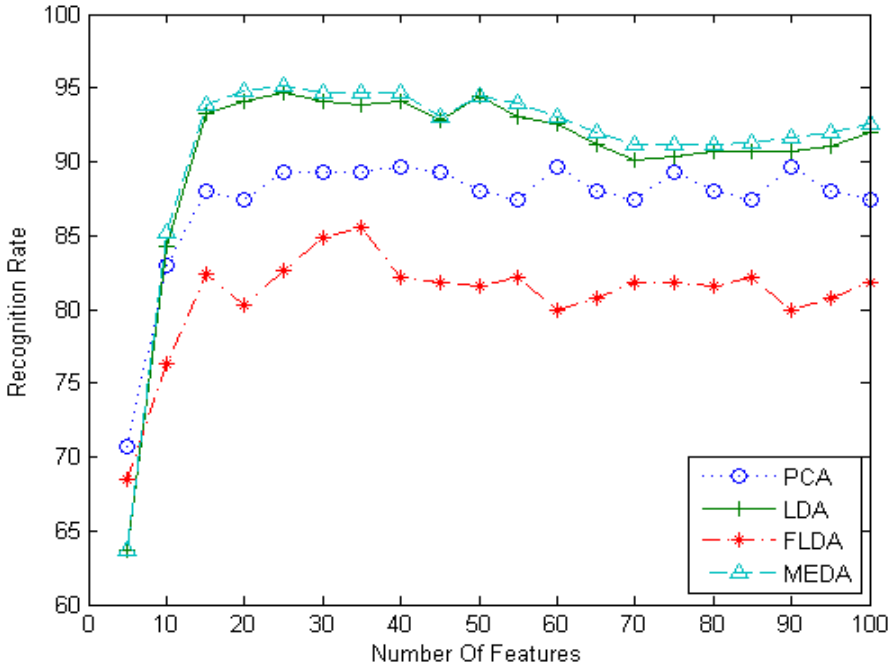


Figure 7-16: The linear algorithms applied on Sheffield database using RBF neural network as a classifier

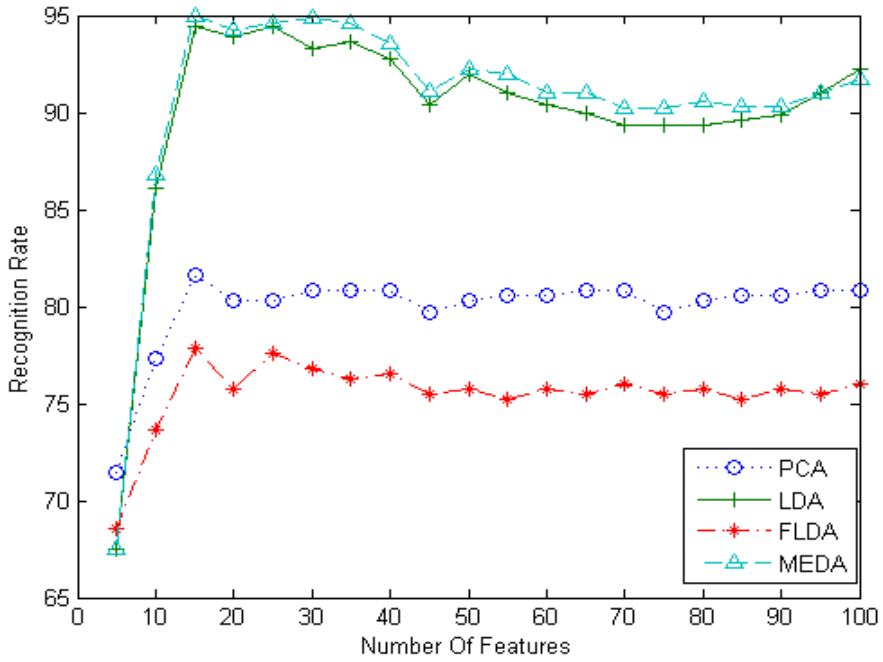


Figure 7-17: The linear algorithms applied on Sheffield database using Euclidean distance as a classifier

The two-dimensional Principal Component Analysis (2DPCA) and two-dimensional Linear Discriminant Analysis (2DLDA) did not produce acceptable performance. One of the problems was that when we have two dimensional data, we cannot apply them to neural network. The matrix can be changed to a vector but the size is too large, and due to small number of training images the training of the neural network was not done very well. Table 7-3 shows the recognition rate of these two methods on ORL database using Euclidean distance as classifier. *E* is the number of the features extracted from the image.

	E=5	E=10	E=15	E=20	E=25
2DPCA	92	93	90	89.5	88.5
2DLDA	93	90.5	85	80.5	77

Table 7-3: Comparing the two dimensional methods on ORL database using Euclidean distance classifier

The Feature Specific Subspace method did not have acceptable performance on face recognition. The reason was the small number of training images for the images in each class. For the ORL database the recognition rate was 88.5% and for Sheffield database it was 78.1% both using Euclidean distance as a classifier.

7.3 Non-linear Algorithm Results

In this experiment the Kernel Principal Component Analysis (KPCA) and Kernel Linear Discriminant Analysis (KLDA) are compared to each other. For KLDA first KPCA is applied to

the images and then KLDA is used. Second order polynomial is used as the kernel function, because in the training part using five-fold cross validation and choosing second order polynomial, third order polynomial and RBF kernel with sigma 1, the second order polynomial performed better. It was chosen to be used as the kernel function. KLDA showed better results compared to KPCA. The results of KLDA are comparable to LDA.

Figures 7-18 to 7-23 show the comparison of the non-linear algorithms using RBF neural network, nearest neighbor using Euclidean distance, and city block distance as a classifier.

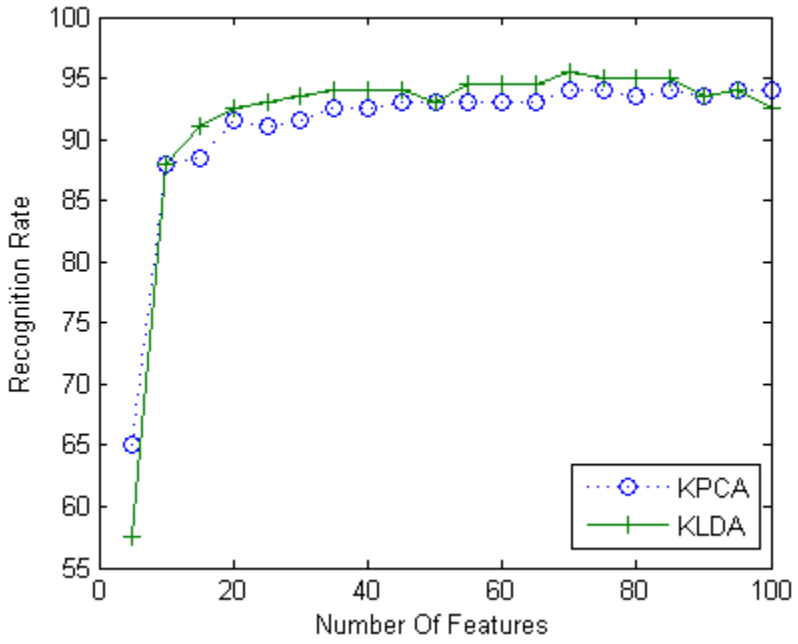


Figure 7-18: The non-linear algorithms applied on ORL database using RBF neural network as a classifier

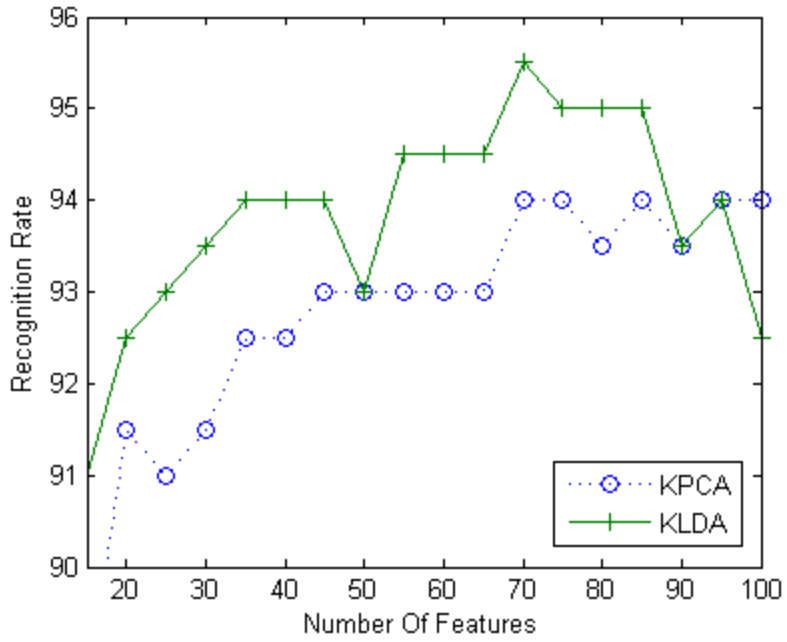


Figure 7-19: The non-linear algorithms applied on ORL database using RBF neural network as a classifier, figure 7-18 from a closer view

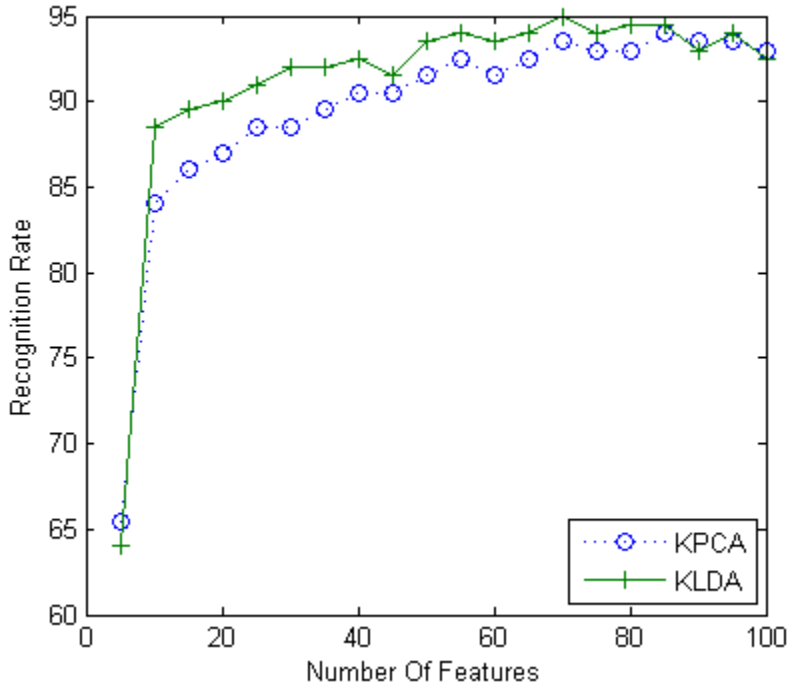


Figure 7-20: The non-linear algorithms applied on ORL database using Euclidean distance as a classifier

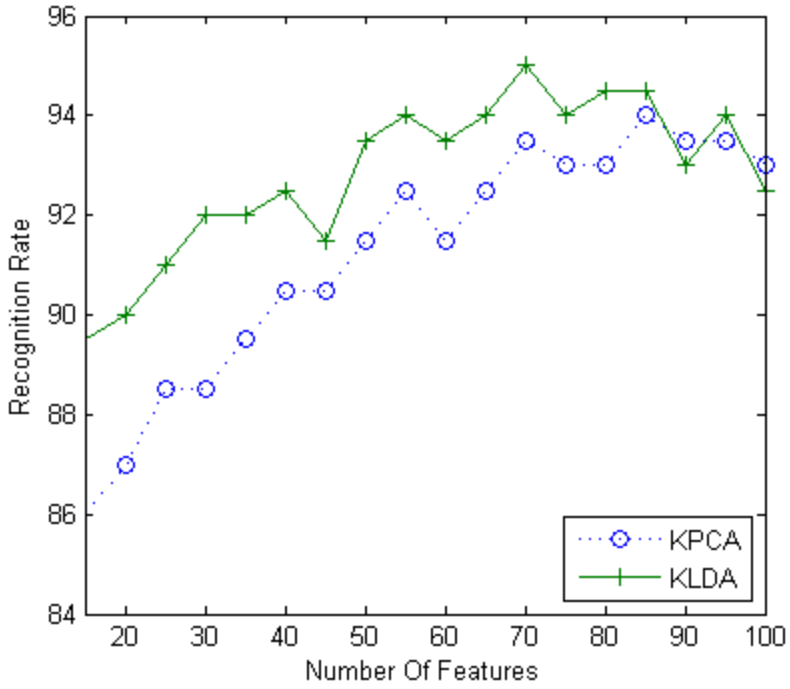


Figure 7-21: The non-linear algorithms applied on ORL database using Euclidean distance as a classifier, figure 7-20 from a closer view

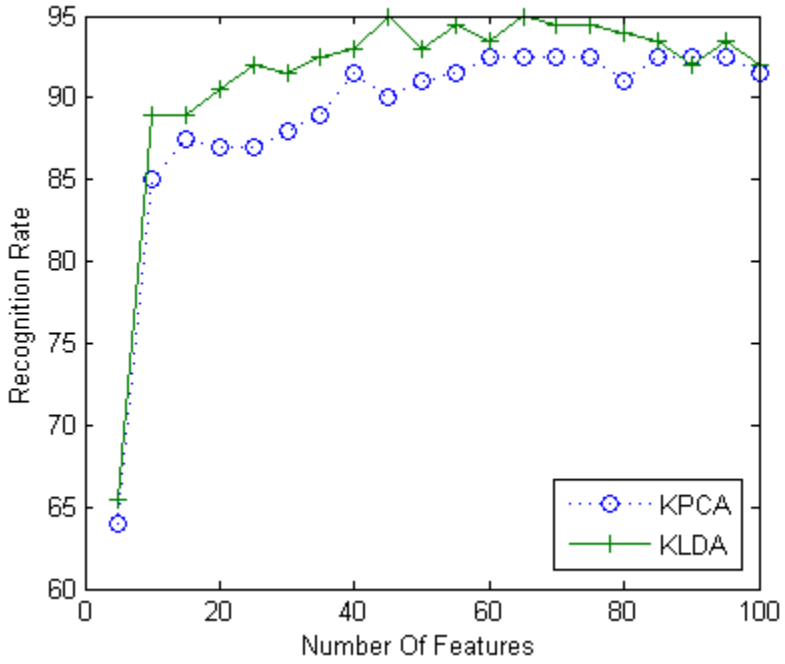


Figure 7-22: The non-linear algorithms applied on ORL database using city block distance as a classifier

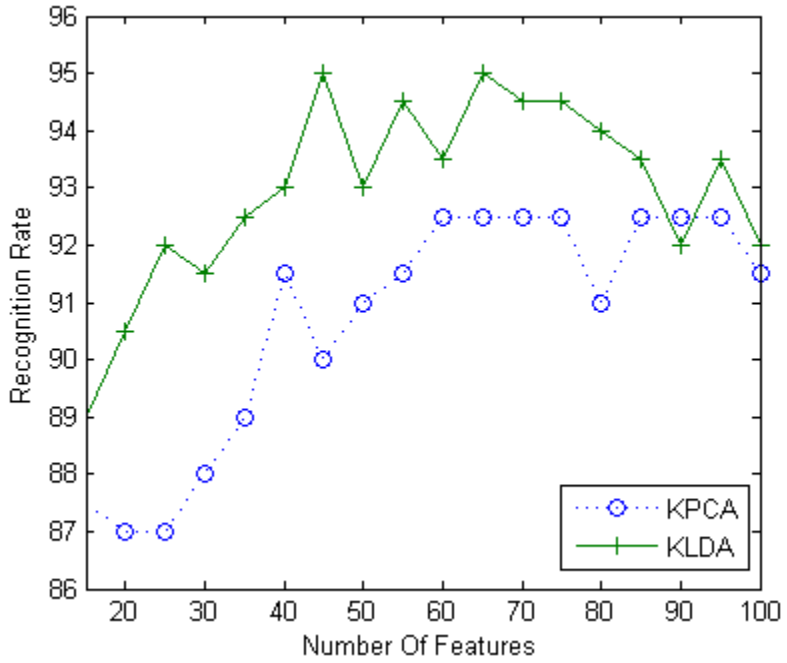


Figure 7-23: The non-linear algorithms applied on ORL database using city block distance as a classifier, figure 7-22 from a closer view

Comparing the results with the linear algorithms shows that the non-linear method is not that much better than the linear methods. The reason can be that the between class distances have not become more when we change the space to a higher dimensional space. Tables 7-4 and 7-5 include the run time for the non-linear algorithms.

	Training time (seconds)	Testing time (seconds)	Average Time for recognizing one image (seconds)
KPCA	37.1534	41.7000	0.3085
KLDA	37.3237	95.2269	0.4761

Table 7-4: The run time of non-linear algorithms using RBF neural networks on ORL database

	Training time (seconds)	Testing time (seconds)	Average Time for recognizing one image (seconds)
KPCA	22.6254	58.5542	0.2928
KLDA	24.2148	87.2154	0.4361

Table 7-5: The run time of non-linear algorithms using Euclidean distance on ORL database

As the results illustrate, the run time of the non-linear methods is much more than the linear methods. Figures 7-24 to 7-26 show the results of non-linear algorithms on Sheffield database using RBF neural network, nearest neighbor with Euclidean distance and city block distance as a classifier. The KLDA shows acceptable results on Sheffield database.

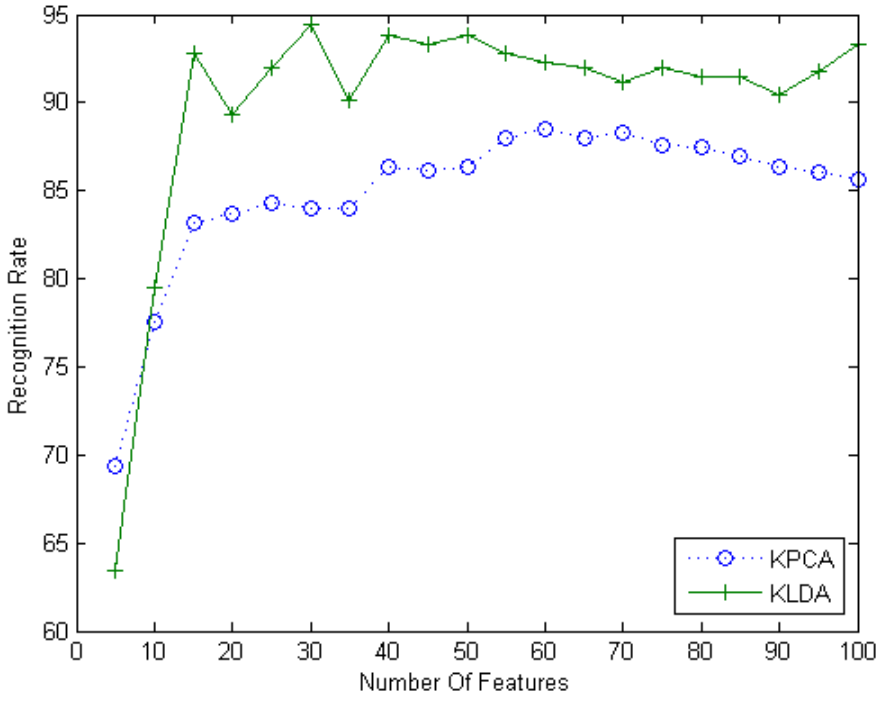


Figure 7-24: The non-linear algorithms applied on Sheffield database using RBF neural network as a classifier

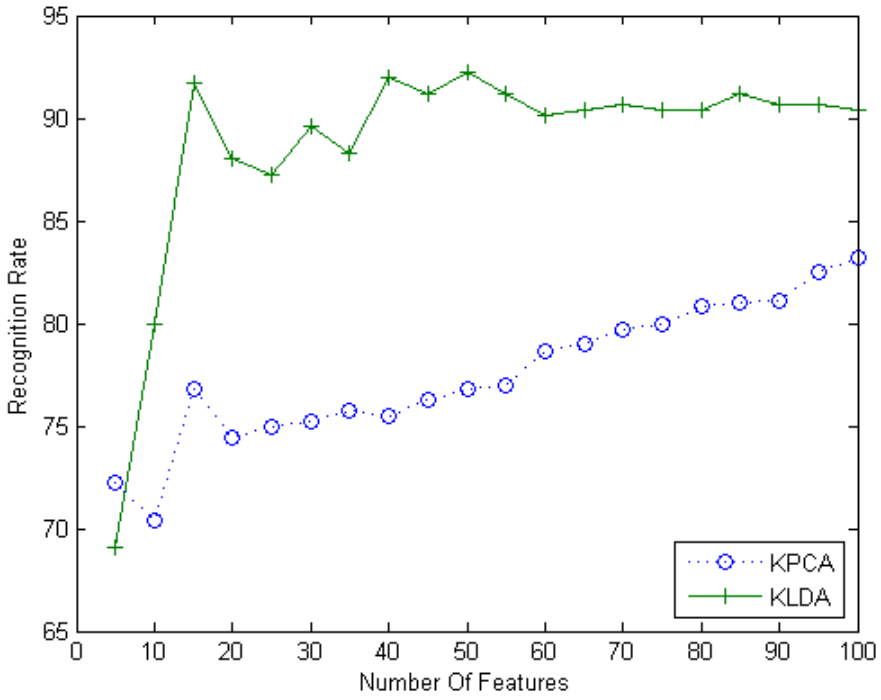


Figure 7-25: The non-linear algorithms applied on Sheffield database using Euclidean distance as a classifier

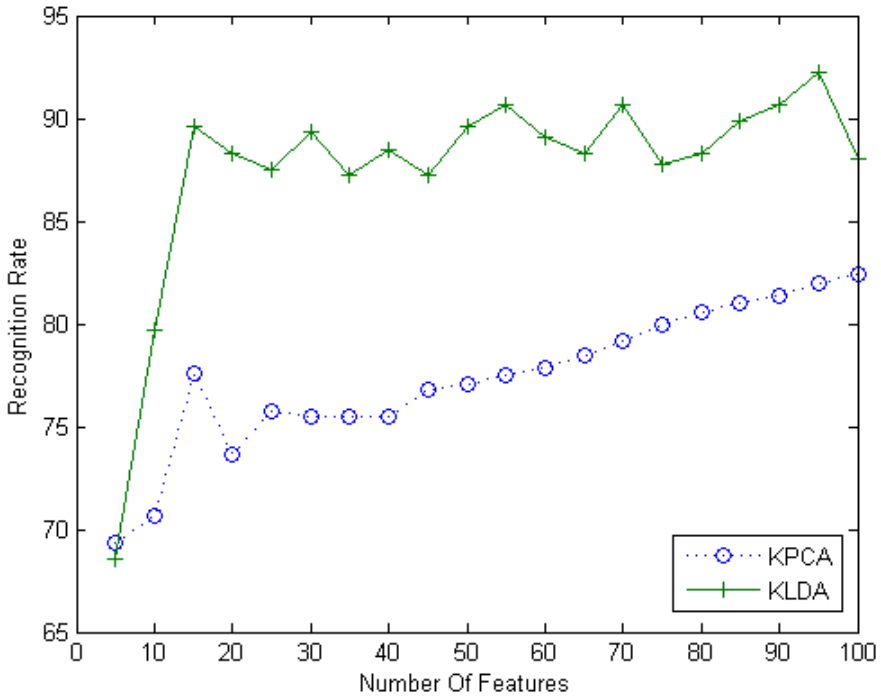


Figure 7-26: The non-linear algorithms applied on Sheffield database using city block distance as a classifier

7.4 Canonical Correlation Analysis (CCA) Results

Combining the information is a powerful technique that is being used in data processing. This combining can be done at three levels of pixel level, feature level, and decision level (Similar to combining the classifiers). CCA combines the information in the feature level.

One of the advantages of combining the features is that the features, vectors which have been calculated using different methods contain different characteristic from the pattern. By combining these two methods not only the useful discriminant information from the vectors are kept, but also the redundant information is omitted.

For this experiment, CCA has been applied to two different feature vectors. The feature vectors have been produced from applying Principal Component Analysis (PCA) and Discrete Cosine Transform (DCT) to images and combining the results using CCA.

The DCT changes the image information to coefficients that show the frequency information. DCT can also be used for compressing information. The DCT is defined as (Ahmed *et al.*, 1974):

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\frac{\pi(2x+1)u}{2N}\cos\frac{\pi(2y+1)v}{2N}$$

$$u,v = 0,1,2,...,N-1$$

$$\alpha(u) = \begin{cases} \sqrt{1/N} & u = 0 \\ \sqrt{2/N} & u \neq 0 \end{cases}$$

For an $N \times N$ image, the image is processed from the top left pixel in a zigzag manner and the pixel's values are put in a vector. As a feature extraction method, the DCT will change the high dimensional image to a low dimensional one in which the important features of the face, such as line of hair, place of eyes, nose and etc. are kept.

Figures 7-27 to 7-29 show the result of applying CCA algorithm using DCT and PCA to ORL and Sheffield databases. Table 7-6 shows the run time for CCA algorithm using 50 features.

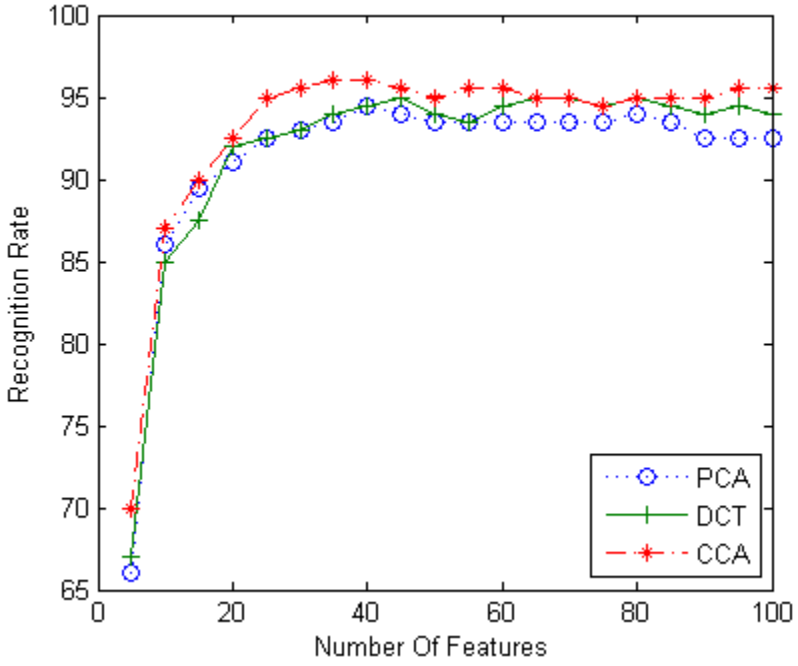


Figure 7-27: The CCA algorithm using PCA and DCT applied on ORL database using RBF neural network as a classifier

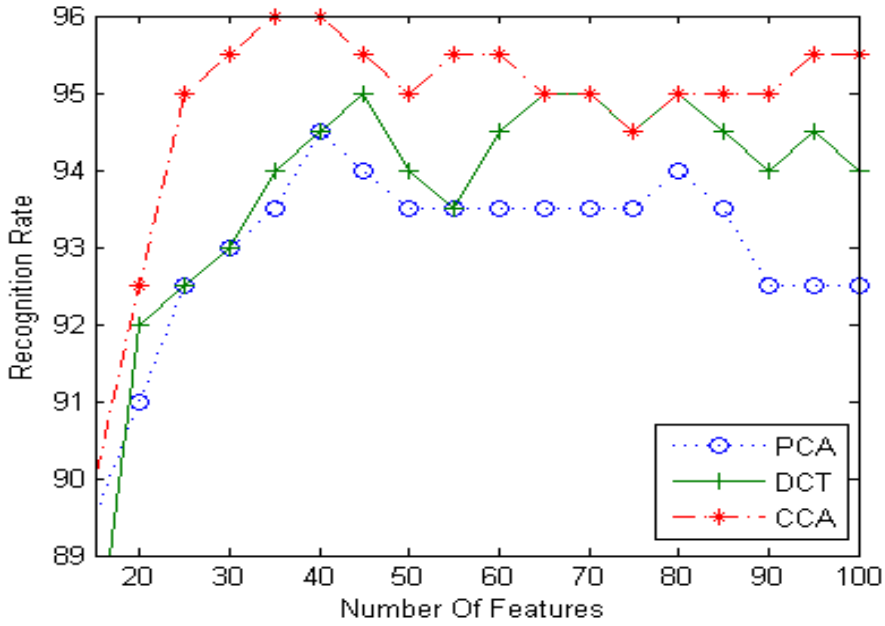


Figure 7-28: The CCA algorithm using PCA and DCT applied on ORL database using RBF neural network as a classifier, figure 7-27 from a closer view

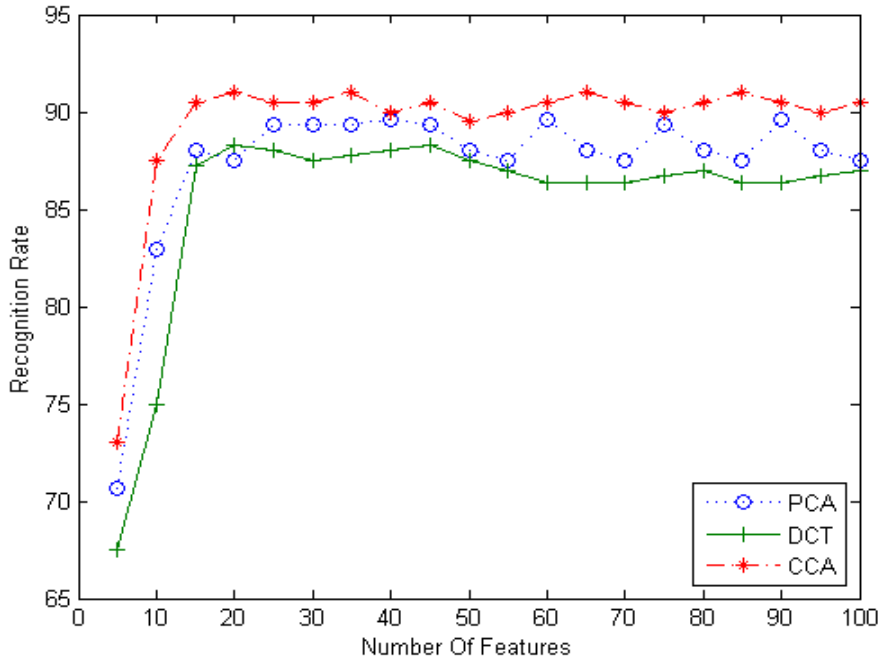


Figure 7-29: The CCA algorithm using PCA and DCT applied on Sheffield database using RBF neural network as a classifier

	Training time (seconds)	Testing time (seconds)	Average Time for recognizing one image (seconds)
DCT	16.2634	2.9743	0.0149
CCA	51.9238	16.9944	0.0850

Table 7-6: The run time of CCA and DCT algorithms using RBF neural networks on ORL database

The results show that CCA has increased the recognition rate. Also it shows that DCT has good performance in face recognition and also it is fast.

We also did another experiment using CCA. We decided to apply CCA on stronger features. A method has been introduced based on DCT that extract features that have better capability to discriminate faces (Dabbaghchian *et al.*, 2010). As mentioned before in conventional DCT the coefficients are chosen using a zigzag manner, where some of the low frequency coefficients are discarded because they contain the illumination information. The low frequency coefficients are in the upper left part of the image. Some of the coefficients have more discrimination power compared to other coefficients, and therefore by extracting these features a higher true recognition rate can be achieved. So, instead of choosing the coefficients in a zigzag manner Dabbaghchian *et al.* (2010) searched for coefficients which have more power to discriminate between images. Unlike other methods such as PCA and LDA which use between and within class scatter matrices and try to maximize the discrimination in the transformed domain, DPA searches for the best discrimination features in the original domain.

The DPA algorithm is as follows (Dabbaghchian *et al.*, 2010):

Considering DCT has been applied to an image and coefficients are *X:*

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ \dots & \dots & \dots & \dots \\ x_{M1} & x_{M2} & \dots & x_{MN} \end{bmatrix}_{M \times N}$$

Where the number of people in the database is C (The number of classes), and for each person there are S images (Training images). There are total C*S training images. The following algorithm shows how to calculate the DPA of each coefficient x_{ij}:

1. Construct a large matrix containing all the DCT from the training images.

$$A_{ij} = \begin{bmatrix} x_{ij}(1,1) & x_{ij}(1,2) & \dots & x_{ij}(1,C) \\ x_{ij}(2,1) & x_{ij}(2,2) & \dots & x_{ij}(2,C) \\ \dots & \dots & \dots & \dots \\ x_{ij}(S,1) & x_{ij}(S,2) & \dots & x_{ij}(S,C) \end{bmatrix}_{S \times C}$$

2. Calculate the mean and variance of each class:

$$M_{ij}^{c} = \frac{1}{S}\sum_{s=1}^{S} A_{ij}(s,c)$$

$$V_{ij}^{c} = \sum_{s=1}^{S}\left(A_{ij}(s,c) - M_{ij}^{c}\right)^2 \qquad c = 1,2,\dots,C$$

3. Calculate variance of all classes:

$$V_{ij}^{W} = \frac{1}{C}\sum_{c=1}^{C} V_{ij}^{c}$$

4. Calculate the mean and variance of all training samples:

$$M_{ij}^{c} = \frac{1}{S \times C}\sum_{c=1}^{C}\sum_{s=1}^{S} A_{ij}(s,c)$$

$$V_{ij}^{B} = \sum_{c=1}^{C}\sum_{s=1}^{S}\left(A_{ij}(s,c) - M_{ij}\right)^2$$

5. For location *(i, j)* calculate the DP:

$$D(i,j)=\frac{V_{ij}^{B}}{V_{ij}^{W}} \quad 1\le i\le M,\ 1\le j\le N$$

The higher value in *D* shows the higher discrimination ability it has. The procedure for recognizing faces are as follow:

1. Compute the DCT of the training images, and normalize the results.
2. Use a mask, to discard some of the low and high frequencies.
3. Calculate DPA for the coefficients inside the mask.
4. The *n* largest coefficients are found and marked. Set the remaining coefficients to zero. The resulting matrix is an *M*N* matrix having *n* elements that are not zero.
5. Multiply the DCT coefficients by the matrix which was calculated in the previous step. Convert the resulting matrix into a vector.
6. Train a classifier using the training vectors. Apply the same process for the test images.

Figures 7-30 and 7-31 show the comparison between FMEDA, DPA and CCA. The results illustrate that applying CCA to the features can increase the recognition rate for human faces.

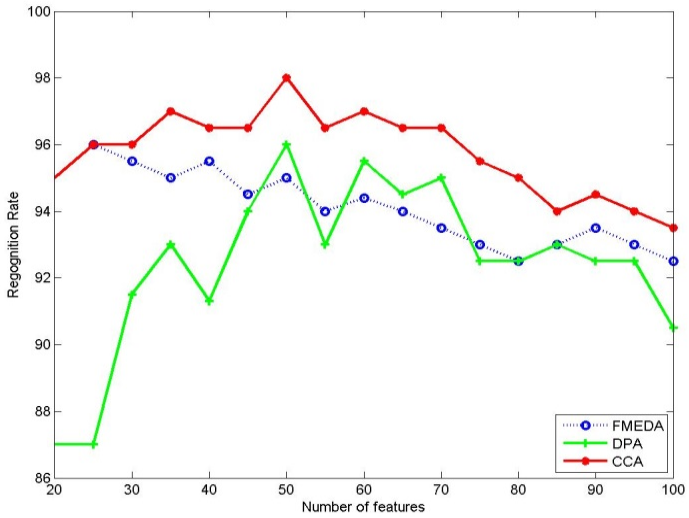


Figure 7-30: Comparing CCA with FMEDA and DPA using RBF classifier on ORL database

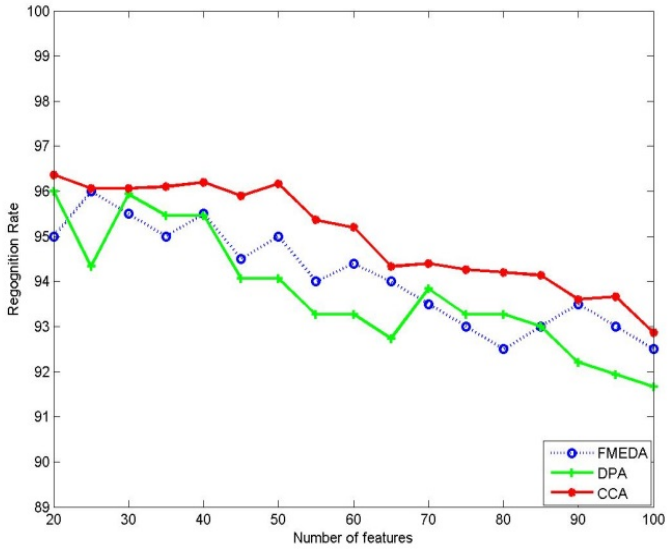


Figure 7-31: Comparing CCA with FMEDA and DPA using RBF classifier on Sheffield database

7.5 Fractional Step Dimensionality Reduction Results

The fractional step dimensionality reduction has been rarely used on face recognition and has not been applied to Multi Exemplar Discriminant Analysis. In this experiment we have combined these two methods with each other and Figures 7-32 till 7-35 show the results. In the implementation we used $r = 35$ and the dimension of the space was reduced 10 times. The weight function d^{-4} had better results compared to other weights function. The new method is called Fractional Multi Exemplar Analysis (FMEDA). Table 7-7 shows the run time of the algorithm.

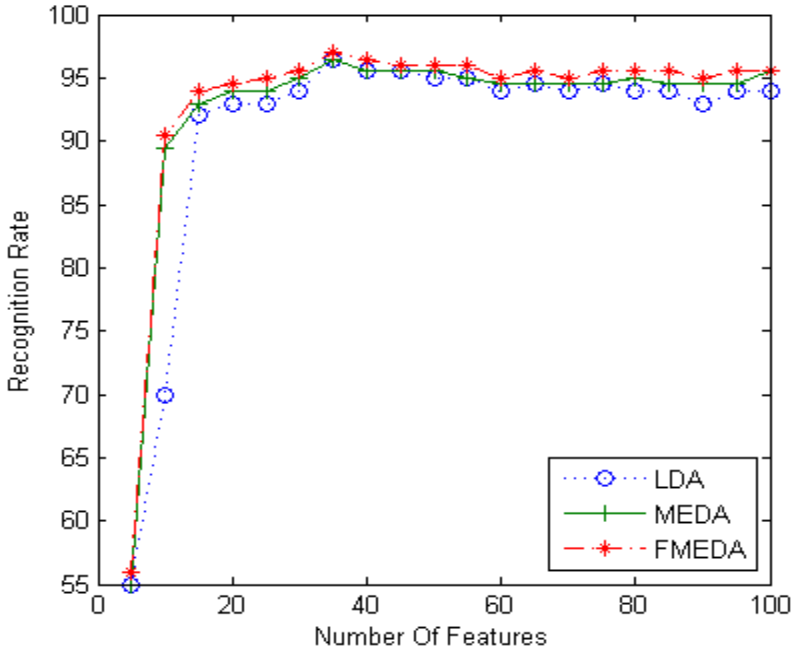


Figure 7-32: The FMEDA algorithm comparing with LDA and MEDA applied on ORL database using RBF neural network as a classifier

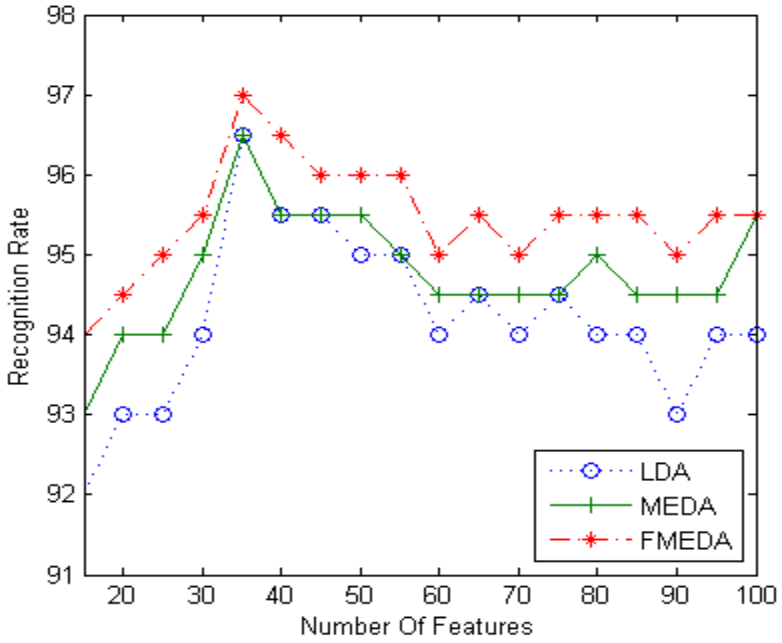


Figure 7-33: The FMEDA algorithm comparing with LDA and MEDA applied on ORL database using RBF neural network as a classifier, figure 7-32 from a closer view

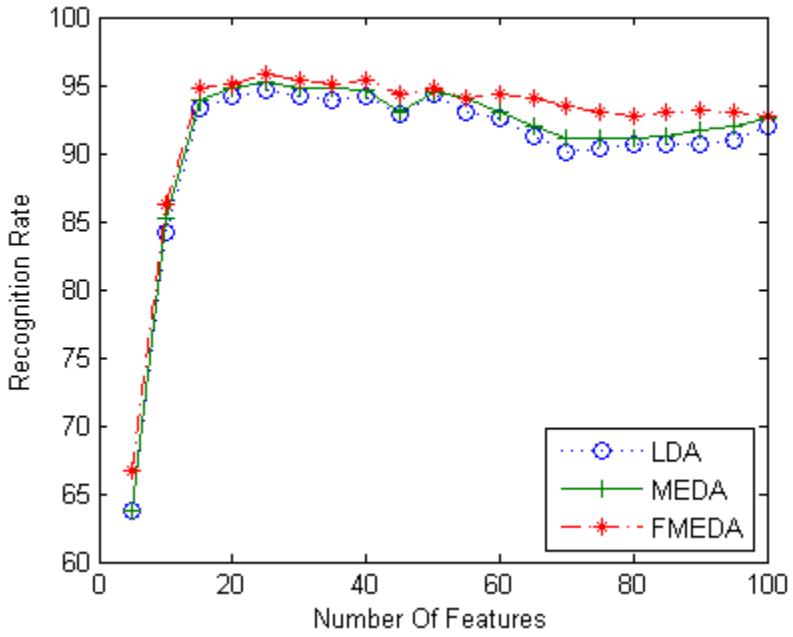


Figure 7-34: The FMEDA algorithm comparing with LDA and MEDA applied on Sheffield database using RBF neural network as a classifier

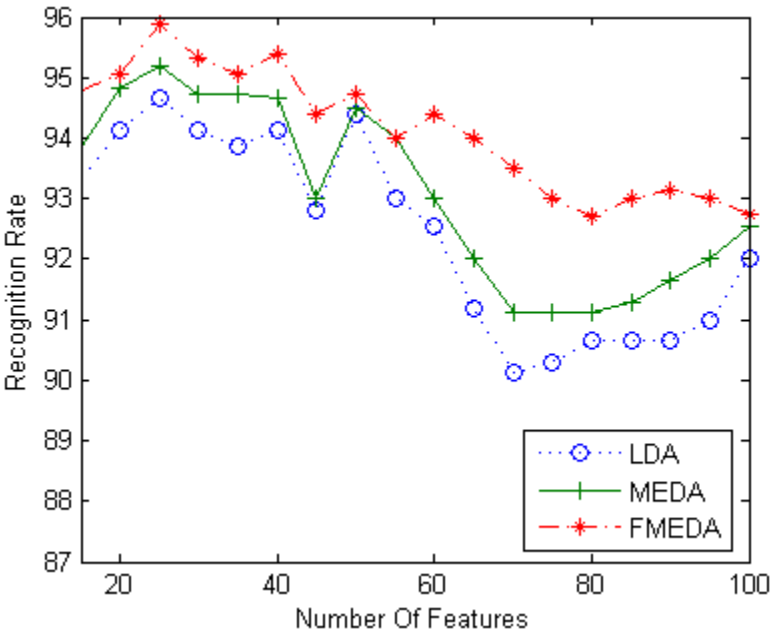


Figure 7-35: The FMEDA algorithm comparing with LDA and MEDA applied on Sheffield database using RBF neural network as a classifier, figure 7-34 from a closer view

	Training time (seconds)	Testing time (seconds)	Average Time for recognizing one image (seconds)
FMEDA	37.9746	20.3893	0.1019

Table 7-7: The run time of FMEDA algorithm using RBF neural networks on ORL database

Chapter 8

Conclusion

8.1 Summary

In this dissertation a face detection and recognition system was designed, developed and tested for color images. In order to detect the face using the color information, first the location of skin should be detected in an image. This process divides the image into two parts, the skin and non-skin components. Neural networks with boosting and deep learning were used for detecting human skin in color images, using a variety of color spaces. The results show that neural network and deep learning have acceptable performance on detecting human skin in color images compared to other methods. Reaching 100% in detection is not possible due to the fact that there are many pixels in color images that are common between the human skin and other objects.

In the face detection phase, a novel methodology to detect faces on color images was proposed, with certain modifications in order to improve the performance of the algorithm. In some images

when the faces are so close to each other, they cannot be separated after skin detection, a method was introduced for separating the face components. Also methods for eye and lip detection were presented using color images. By detecting the eyes and lips in an image or in a face component the face can be easily detected.

In the face recognition phase, several methods in appearance based models were implemented. The experimental results illustrate that MEDA, and also LDA and FMEDA, had better results compared to other techniques. Also by using CCA, the recognition rate was improved. These algorithms were used in the recognition phase of the face recognition system.

8.2 Contributions

The contributions of this dissertation include:

1. Using Neural network with bootstrapping method for skin detection. Different Neural networks has been used in parallel, each trained with different sets of data, using information from different color spaces, also bootstrapping has been used to increase the detection rate.

2. Using Deep Learning method for skin detection. The results show that deep learning has better results compared to other methods in this field.

3. Modifying a template based method for detecting faces in color images after applying the skin detection phase. The methods detects a face in components. If there are more than one face in each component then, the algorithm can detect it.

4. Proposing a rule based method for eyes and lips detection. Detecting the face based on the location of the eyes and lips in the component.

5. Using Canonical Correlation analysis for improving the recognition rate. Testing and comparing different algorithms in appearance based methods on ORL and Sheffield databases.

8.3 Limitation and Future Work

The limitation and future work of this dissertation include:

1. Applying other machine learning and deep learning methods to the skin detection phase.

2. Designing algorithms that reduce the effect of illumination on the image. The illumination causes changes on the skin. Using and designing algorithms that reduce the effect of illumination can increase the detection rate of skin detection phase which can improve the detection rate in the face detection phase.

3. Applying deep learning for detecting faces in the components where the designed methods failed. Sometimes faces are missed in some components. This is the situation when more than one face is in the component and the designed template method fails. In this case deep learning method can be used in order to search the component. A window will be moved on the component and the window is applied to an Deep Belief Net in order to find the faces.

4. Building a database for face detection and face recognition. The databases which are available, most of them are black and white. There are some good databases in face detection and some databases in the face recognition, but there is not a database which contains images that are suitable for both face detection and face recognition. In this case, around 40-50 people should be chosen from different ages and races. Around 10 images should be taken individually, these images will be used for the training phase of the face recognition system. Around 100 images from combination of different people in the database will be used for the detection and recognition in the testing phase.

5. Testing the result of applying the designed algorithms on the database.

REFERENCES

[1] Abdallah, A.S., Abbott, A.L. and El-Nasr, M.A. A New Face Detection Technique using 2D DCT and Self Organizing Feature Map, *Proceedings of World Academy of Science, Engineering and Technology 21*, 15-19.

[2] Abdallah, A.S., El-Nasr, M.A. and Abbott, A.C. A New Colour Image Database for Benchmarking of Automatic Face Detection and Human Skin Segmentation Techniques, *Proceedings of World Academy of Science, Engineering and Technology 20*, 353-357.

[3] Abdel-Mottaleb, M. and Elgammal, A. Face detection in complex environments from color Images. *Proceedings of the International Conference on Image Processing (ICIP)*, 622–626, 1999.

[4] Ahmed, N., Natarajan, T., and Rao, K. Discrete Cosine Transform, *IEEE Transactions on Computers, 23*. 90-93, 1974.

[5] Al-Mohair, H., Saleh, J., and Suandi, S. Human skin color detection: A review on neural network perspective, *International Journal of Innovative computing, information and control, 8*(12). 8115-8131, 2012.

[6] Alshehri S. Neural Networks Performance for Skin Detection, *Journal of Emerging Trends in Computing and Information Sciences, 3*(12). 1582-1585, 2012.

[7] Bakry, E., Zhao, H.M., and Zhao, Q. Fast Neural implementation of PCA for face detection, *International Joint Conference on Neural Networks,* 806-811, 2006.

[8] Bengio, Y., Lamblin, P., Popovici, D. and Larochelle, H. Greedy layer-wise training of deep networks, *Advances in Neural Information Processing Systems (NIPS'06), 19* , 153-160, 2007.

[9] Berbar, M.A., Kelash, H.M., and Kandeel, A.A. Faces and Facial features detection in color images, *GMAI06, London, UK, ISBN/NN: 0-7695-2604-7, IEEE computer society*, 160-168, 2006.

[10] Blanz, V.S., and Vetter, T. Face identification across different poses and illuminations with a 3D morphable model, *IEEE International Conference on Automatic Face and Gesture Recognition*, 202–207, 2002.

[11] Borga, M. Learning multidimensional signal processing, *Department of Electrical Engineering, Linköping University, Linköping Studies in Science and Technology Dissertations, No.531*, 1998.

[12] Boser, B.E., Guyon, I.M., and Vapnik, V.N. A training algorithm for optimal margin classifiers. *In D. Haussler, editor, Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, 144-152, 1992.

[13] Brancati, N., De Pietro, G., Frucci, M., Gallo, L. Human Skin Detection through Correlation Rules between the YCb and YCr Subspaces based on Dynamic Color Clustering, *Computer Vision and Image Understanding, 155*, 33-42, 2016.

[14] Brunelli, R., and Poggio, T. Face recognition: Features versus templates *IEEE Transactions on Pattern Analysis and Machine Intelligence, 15*. 1042–1053, 1993.

[15] Chai, D., and Ngan, K.N. Locating facial region of a head and shoulders color image, *ICFGR98, In Proceedings of the 3rd. International conference on face and gesture recognition*, 124-129, 1998.

[16] Chandrappa, D.N., Ravishankar, M., and RameshBabu, D.R. Face detection in color images using skin color model algorithm based on skin color information, *2011 3rd International Conference on Electronics Computer Technology (ICECT)*, 254-258, 2011.

[17] Chen, H., Huang, C., and Fu, C. Hybrid-boost learning for multi-pose face detection and facial expression recognition, *Pattern Recognition society, Elsevier, 41*. 1173-1185, 2008.

[18] Chiang, C., Tai, W., Yang, M., Huang, Y. and Huang, C., A novel method for detecting lips, eyes and faces in real time, *Real time imaging, 9*, 277-287, 2003.

[19] Chin, S., Seng, K., Ang, L. and Lim, K., New lips detection and tracking system, *Proceedings of the international multiconference of engineers and computer scientists, 1*, 2009.

[20] Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., and Vedaldi, A. *Describing Textures in the Wild*, Computer Vision and Pattern Recognition (CVPR), 2014.

[21] Cootes, T.F., Edwards, G.J., and Taylor, C.J. Active appearance models, *IEEE Transanctions Pattern Analysis and Machine Intelligence, 23*(6). 681–685, 2001.

[22] Dabaghchian, S., Ghaemmaghami, M. and Aghagolzadeh, A. Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology, *Pattern Recognition, 43.* 1431-1440, 2010.

[23] Doukim, C.A., Dargham, J.A., Chekima, A. and Omatu, S. Combining neural networks for skin detection, *Signal & Image Processing: An International Journal (SIPIJ), 1*(2). 1-11, 2010.

[24] Elgammal, A., Muang, C. and Hu, D., Skin Detection – a Short Tutorial, *Encyclopedia of Biometrics by Springer-Verlag Berlin Heidelberg,* 2009.

[25] Er, M.J., Wu, S., Lu, J., and Toh, H.L. Face recognition with radial basis function (RBF) neural networks, *IEEE Trans. on neural networks , 13*(3). 2002.

[26] Fink, B., Grammer, G. and Matts, P., J. Visible skin color distribution plays a role in the perception of age, attractiveness, and health in female faces. *Evolution and Human Behavior 27*(6), 433-442, 2006.

[27] Fleck, M.M., Forsyth, D.A. and Bregler, C. Finding naked people. *Proceedings of the European Conference on Computer Vision (ECCV)*, 593–602, 1996.

[28] Freund, Y. and Schapire, R. A Short Introduction to Boosting. *Journal of Japanese Socity for Artificial Intelligence, 14*(5). 771-780, 1999.

[29] Fukunaga, K. Introduction to statistical pattern recognition, *2nd ed. San Diego, CA: Academic Press,* 445-450, 1990.

[30] Georgia tech face database: http://www.anefian.com/research/face_reco.htm , 2015.

[31] Gomez, G., and Morales, E.F. Automatic feature construction and simple rule induction algorithm for skin detection, *Proceedings Of the ICML workshop on machine learning in computer vision,* A. Sowmya, T. Zrimec(eds), 2002.

[32] Hajiarbabi M., and Agah A. Human Skin Color Detection using Neural Networks, *Journal of Intelligent Systems, 24*(4). 425-436, 2014.

[33] Hajiarbabi M. and Agah A. Face Detection in color images using skin segmentation, *Journal of Automation, Mobile Robotics and Intelligent Systems, 8*(3). 41-51, 2014.

[34] Hajiarbabi M. and Agah A. Face recognition using canonical correlation, discrimination power and fractional multiple exemplar discriminant analyses, *Journal of Automation, Mobile Robotics and Intelligent Systems*, 9(4), 18-27, 2015.

[35] Hajiarbabi M. and Agah A. Techniques for skin, face, eye and lip detection using skin segmentation in color images, *International Journal of computer vision and image processing*, *5*(2), 2016.

[36] Hajiarbabi M. and Agah A. Human skin detection in color images using deep learning, , *International Journal of computer vision and image processing*, *5*(2), 2016.

[37] Hassaballah, M., Murakami, K. and Ido., S. An automative eye detection method for gray intensity facial images, *International journal of computer science issues, 8*(2), 272-282, 2011.

[38] Hietmeyer, R. Biometric identification promises fast and secure processing of airline passengers, *The International Civil Aviation Organization Journal, 55*(9). 10-11, 2000.

[39] Hinton, G.E., Osindero, S. and The, Y. A fast learning algorithm for deep belief nets. *Neural Computation, 18*, 1527-1554, 2006.

[40] Hinton, G. and Salakhutdinov, R. Reducing the dimensionality of data with neural networks, *Science, 313*(5786), 504-507, 2006.

[41] Hjelmas, E., and Kee Low, B. Face Detection: A Survey, *Computer Vision and Image Understanding (CVIU), 83*(3), 236-274, 2001.

[42] Hotelling, H. Relations between two sets of variates, *Biometrika, 8*, 321–377, 1936.

[43] Huang, J., Heisele, B., and Blanz, V. Component-based Face Recognition with 3D Morphable Models, *In Proceedings of the Fourth International Conference on Audio- and Video-based Biometric Person Authentication*, Surrey, UK, 2003.

[44] Jeng, S., Liao, H., Liu, Y., and Chern, M. An efficient approach for facial feature detection using Geometrical Face Model, *1996 Proceedings Of 13th International Conference on pattern recognition, 3*. 426-430, 1996.

[45] Kalbkhani, H. and Amirani, M. An efficient algorithm for lip segmentation in color face images based on local information, *Journal of world's electrical engineering and technology*, *1*(1), 12-16, 2012.

[46] Khan, I., Abdullah, H. and Bin Zainal, M. Efficient eyes and mouth detection algorithm using combination of Viola Jones and skin color pixel detection, *International Journal of Enginerring and Applied Sciences*, *3*(4), 51-60, 2013.

[47] Kong, W., and Zhe, S. Multi-face detection based on down sampling and modified subtractive clustering for color images, *Journal of Zhejiang University*, *8*(1). 72-78, 2007.

[48] Kovac, J., Peer, P., and Solina, F. Human Skin Color Clustering for Face Detection, *EUROCON 2003. Computer as a Tool. The IEEE Region 8. 2*. 144-148, 2003.

[49] Kukharev, G., and Novosielski, A. Visitor identification elaborating real time face recognition system, *In Proceedings of the 12th Winter School on Computer Graphics (WSCG), Plzen, Czech Republic*, 157-164, 2004.

[50] Kwak, K.C., and Pedrycz, W. Face recognition using a fuzzy Fisher face classifier, *Pattern Recognition, 38*. 1717 – 1732, 2005.

[51] Lin, S.H., Kung, S.Y. and Lin, L. J. Face recognition/detection by probabilistic decision-based neural network, *IEEE Transactions On Neural Networks, 8*. 114-132, 1997.

[52] Lotlikar, R., and Kothari, R. Fractional-step dimensionality reduction, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*. 623–627, 2000.

[53] Math Works: www.mathworks.com 2015.

[54] Maryaz, G. and Hinton, G.E. Recognizing hand-written digits using hierarchical products of experts. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*, 180-197, 2001.

[55] ORL Database: http://www.camorl.co.uk 2015.

[56] Peng, K., Chen, L., Ruan, S. and Kukharev, G. A robust algorithm for eye detection on gray intensity face without spectacles, *Journal of computer science and technology*, *5*(3), 127-132, 2005.

[57] Rajpathak, T., Kumar, R. and Schwartz, E. Eye detection using morphological and color image processing, *Florida conference on recent advances in robotics*, 2009.

[58] Rowley, H., Baluja, S., and Kanade, T. Neural network-based face detection, *IEEE Pattern Analysis and Machine Intelligence*, *20*. 22-38, 1998.

[59] Schapire, R., Freund, Y., Bartlett, P., and Lee, W. Boosting the margin: A new explanation for the effectiveness of voting methods. *Machine Learning: In Proceedings of the Fourteenth International Conference, 26*(5). 1651-1686, 1998.

[60] Scholkopf, B. Statistical learning and kernel methods, *Microsoft Research Limited, February 29*, 2000.

[61] Scholkopf, B., Smola, A. and Muller, K. R. Non-linear component analysis as a kernel eigenvalue problem, *Neural Computation. 10* (5). 1299–1319, 1998.

[62] Seow, M, Valaparla, D., and Asari, V. Neural network based skin color model for face detection. *In Proceedings of the 32nd Applied Imagery Pattern Recognition Workshop (AIPR'03)*, 141-145, 2003.

[63] Shan, Sh., Gao, W., and Zhao, D. Face Identification based On face-specific subspace, *International Journal of Image and System Technology, Special issue on face processing, analysis and synthesis*, *13*(1), 23-32, 2003.

[64] Shavers, C., Li, R., and Lebby, G. An SVM-based approach to face detection, *In 2006 Proceedings Of Thirty-Eighth Southeastern Symposium On system Theory,* 362-366, 2006.

[65] Sheffield Database: https://www.sheffield.ac.uk/eee/research/iel/research/face, 2014.

[66] Sidbe, D., Montesinos, P. and Janaqi, S. A simple and efficient eye detection method in color images, *International conference image and vision computing,* 385-390, 2006.

[67] Singh, S., Chauhan, D.S., Vatsa, M., and Singh, R. A Robust Skin Color Based Face Detection Algorithm, *Tamkang Journal of Science and Engineering, 6*(4). 227-234, 2003.

[68] Soetedjo, A. Eye detection based on color and shape features, *International journal of advanced computer sceience and applications, 3*(5), 17-22, 2011.

[69] Sun, Q.S., Zeng, S.G., Liu, Y., Heng, P.A., and Xia, D.S. A new method of feature fusion and its application in image recognition, *Pattern Recognition,* 2005.

[70] Sung, K. Learning and example selection for object and pattern detection, *PhD Thesis, MIT AI Lab,* 1996.

[71] Tesauro, G., Practical issues in temporal difference learning. *Machine Learning*, *8*, 257-277, 1992.

[72] Turk M. A Random Walk through Eigen space, *IEICE transactions on Information and syst*em, *84*(12) 2001.

[73] Turk, M., and Pentland, A. Eigen faces for recognition, *Journal of Cognitive Neuroscience.*, *3*, 71–86, 1991.

[74] UCD Database: http://ee.ucd.ie/~prag/, 2014.

[75] Viola, P., and Jones, M. J. Robust real-time object detection, *In Proceedings of IEEE Workshop on Statistical and Computational Theories of Vision*, 2001.

[76] Webb, A.R. *Statistical Pattern Recognition*, John Wiley & Sons, New York, 2002.

[77] Wiskott, L., Fellous, J.M., Kruger, N., and Malsburg, C. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7). 775-779, 1997.

[78] Wu, Y., and Ai, X. Face detection in color images using Adaboost algorithm based on skin color information, *In Proceedings of 2008 Workshop on Knowledge Discovery and Data Mining*, 339-342, 2008.

[79] Yang, G., Li, H., Zhang, L., and Cao, Y. Research on a skin color detection algorithm based on self-adaptive skin color model, *International Conference on Communications and Intelligence Information Security*, 266-270, 2010.

[80] Yang, J., Jin, Z., Yang, J., Zhang, D., and Frangi, F. Essence of kernel Fisher discriminant: KPCA plus LDA, *Elsevier Pattern Recognition, 37*, 2097 – 2100, 2004.

[81] Yang, K., Zhu, H., and Pan, Y.J. Human Face detection based on SOFM neural network, *2006 IEEE Proceedings Of International Conference on Information Acquisition*, 1253-1257, 2006.

[82] Yang, M.H., Kriegman, D.J., and Ahuja, N. Detecting Faces in Images: A Survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24* (1). 34-58, 2002.

[83] Yu, H., and Yang, J. A Direct LDA algorithm for high dimensional data with application to face recognition, *Pattern Recognition, 34*. 2067-2070, 2001.

[84] Zhou, Sh. K., and Chellappa, R. Multiple-Exemplar discriminant analysis for face recognition *Center for Automation Research and Department of Electrical and Computer Engineering University of Maryland*, College Park, MD 20742, 2003.

[85] Zuo, H., Fan, H., Blasch, E., and Ling H. Combining Convolutional and Recurrent Neural Networks for Human Skin Detection, *IEEE Signal Processing Letters, 24*(3). 289-293, 2017.