LINKÖPING UNIVERSITY

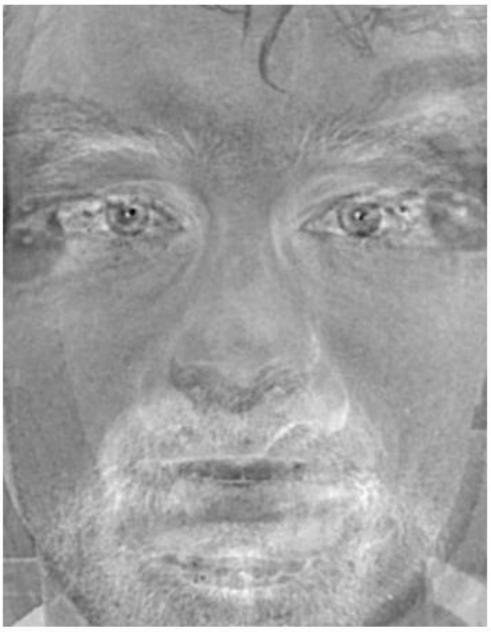
TNM034

ADVANCED IMAGE PROCESSING

Face Recognition

Authors:
Ronja Grosz, RONGR946
Isabell Jansson, ISAJA187
Jens Jakobsson, JENJA698
Christoffer Engelbrektsson, CHREN574

Examinator:
Daniel Nyström



December 11, 2015

Abstract

The paper considers different methods for face detection and face recognition in databases.

For detecting a face in an image, the color has to be corrected. Gray world is used for white balancing the images. The face can be detected by using several algorithms, for example with color space based algorithms or with the Viola-Jones algorithm. The eyes are extracted by eye maps which are used to align the faces. The aligned faces are matched by comparing important features in the faces. The relevant features that have been studied are intensity and phase from the Fourier transform. The intensity based method is called Eigenfaces, and the phase based method is called Local Phase Quantization. For dimension reduction, Principal Component Analysis and Singular Value Decomposition have been compared.

The Eigenface algorithm generally gives the highest matching rate, except from when considering blurred images.

Contents

1 Introduction

1.1 Background

Face recognition is useful in many areas. Such as verifying that the holder of a passport is the right person, identifying people in a criminal database or using face recognition to gain access to personal information or belongings like smartphones.

Facebook is an example of a company which uses face recognition. They have a research group that works on face recognition. The group has written a paper about their face recognition software, *DeepFace* [1]. DeepFace is a software trained on a large data set of varying faces and has an accuracy of 97.35%. The face recognition software is used to tag people in images.

1.2 Purpose

The aim of the project was to develop a program which matches a query image with a database of images. Two methods have been studied, Eigenfaces and Local Phase Quantization together with Principal Component Analysis or Singular Value Decomposition. Both methods extract and compare relevant features from the faces.

1.3 Limitations

The project has been limited to not include a training set of images. No machine learning or other methods - that needs a training set of images - have been used. Instead methods that are either invariant for a specific feature or methods that do not need to *learn* are being used.

The project has been limited to two data sets of images which depicts people from the front and from shoulders and up with background. The second data set is captured under different conditions. The images can be blurred, have different facial expressions, have a cluttered background or different illumination.

The robustness of the program is limited to the following transformations, the program should be able to handle a face rotation of +/- 5 degrees, scaled a maximum of +/- 10 percent and a maximum tone value difference of 30 percent.

2 Related Work

2.1 LogAbout

The paper *Novel Method to Compensate Variety of Illumination In Face Detection* [2] describes a method for balancing the illumination in an image. The method uses a high-pass filter followed by a log transformation. It can give good results under specific conditions. A drawback is that each image must have a specific set of parameter values for the log transformation. This means that it is not efficient for a more general face recognition software.

2.2 Gamma Intensity Correction

In the paper *Illumination Normalization for Robust Face Recognition Against Varying Lighting Conditions* [3] a study was made on different kinds of methods for illumination normalization (balancing). One of the methods presented was Gamma Intensity Correction (GIC). It "correct the overall brightness of the face images to a predefined "canonical" face images ... the GIC is expected to make overall brightness of the input images best fit the pre-defined normal face images". They conclude that the method can "significantly improve the recognition rate compared with the non-preprocessing case since it can eliminate the heavy side lightning effects".

3 Methods

This section describes the methods used in the project. The pipeline can be divided into two main groups, face detection and face recognition. Face detection will describe how the face, eyes and mouth were detected and aligned before the recognition. Face recognition will describe how the recognition can be done by using the methods Eigenfaces or Local Phase Quantization.

3.1 Face Detection

3.1.1 Illumination and Color Balance

Images can have a wide variation in illumination and color. Poor lightning conditions can be a challenge for any face detection software. By balancing the color and illumination a better face detection can be acquired. This section will describe the method Gray World.

3.1.1.1 Gray World

Gray world is a method to white balance an image. It is used when the image is assumed to have an average color of gray [4]. To white balance an image with gray world, the following steps are made [5]:

1. Calculate the average and the total mean of the color channels red, green and blue separately.

$$avgGray = \frac{(avgRed + avgGreen + avgBlue)}{3} \tag{1}$$

2. Calculate the adjustment factors of red, green and blue color channels separately and divide is with average gray from Equation 1.

$$adjustmentfactorColor = \frac{avgColor}{avgGray} \tag{2}$$

3. Multiply the adjustment factors, from Equation 2, by the pixel values for each channel.

$$adjustedColorChannel = adjustmentColorfactor * ColorChannel \tag{3}$$

The result of gray world can be seen in Figure 1. Figure 1a is the original image with a more reddish tone. By applying the gray world method the colors can be balanced and a more uniform color tone can be acquired as seen in figure 1b.

3.1.2 Feature Detection

There are several methods for finding a face, *Detecting skin in face recognition systems: A color spaces study* categorizes them into four groups [6]. One of the categories the paper focuses on is the feature invariant approach by detecting skin color based on color space.

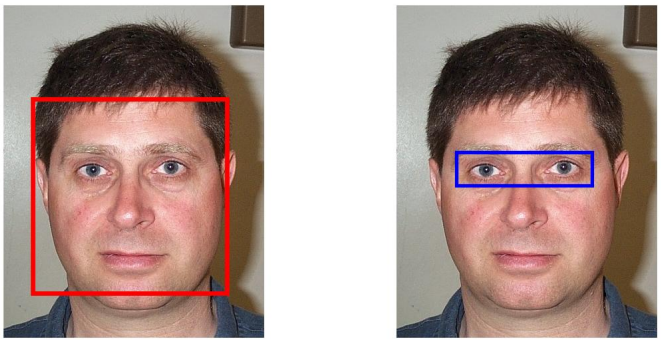
Another method is the Viola-Jones algorithm. Viola-Jones algorithm is included in Matlab and provides a fast and robust object detection framework. This section will briefly present three face detection methods based on Viola-Jones algorithm and color space.

(a) Original (b) Gray World

Figure 1: Before and after using gray world

3.1.2.1 Viola-Jones

Viola-Jones is an algorithm used to detect features in a face. It is an algorithm created by P. Viola and M. Jones, that uses an object detection framework for fast object detection. It has a machine learning approach and is an inbuilt function in *Matlab*. This function can detect objects like face, eyes and mouth that are useful in face detection. For further reading how the algorithm works we recommend the paper *Rapid Object Detection using a Boosted Cascade of Simple Features* [7].



(a) Detection of the face using Viola-Jones (b) Detection of the eyes using Viola-Jones

Figure 2: Found features using Viola-Jones algorithm

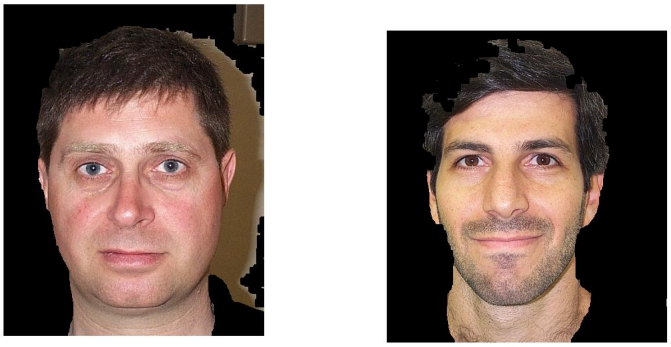
By using the algorithm, regions of interest can be located as seen in Figure 2. These regions can be cropped from the rest of the image to narrow down the search for facial features like eyes. This will further increase the precision of aligning the face based on facial features.

3.1.2.2 Color models and Color Space

Color model and color space is an important part in face detection and can greatly affect the success rate in finding the face or facial features. Detecting skin color can be done by the use of color spaces. This section will briefly talk about two kinds of color spaces that is used in face detection, HSV and YC_bC_r.

HSV stands for Hue, Saturation and Value. An implementation based on the pseudo-code in *Detecting skin in face recognition systems: A color spaces study* [6] gave the following results. Figure

3a illustrates poor segmentation results as parts of the background and hair where included in the segmentation.



(a) Poor results, parts of the background where included

(b) Better segmentation but the hair still got included

Figure 3: Skin segmentation based on HSV method

Figure 3b illustrates a better result but the segmentation still includes large portions of the hair. This implementation was based on the Hue channel and not the whole HSV color space. An implementation based on the whole color space should give an even better result [6]. It should also be noted that a series of dilation and erosion where used to eliminate edges and fill holes.

In YC_bC_r Y represents the luminance component, C_b the blue-difference chrominance component and C_r the red-difference chrominance component [6]. The images given in the project where represented with the RGB channels. Equation 4 explains the conversion between RGB and YC_bC_r. The constants R, G and B are the values in each pixel as represented by the RGB channels.

$$\begin{aligned} Y &= 16 + (65.481R + 128.553G + 24.966B) \\ Cb &= 128 + (-37.797R - 74.203G + 112.0B) \\ Cr &= 128 + (112.0R - 93.786G + 18.213B) \end{aligned} \tag{4}$$

YC_bC_r is widely used in many of the papers regarding face detection and has a high success rate when finding faces [6]. It is also a popular color space when finding face features like eyes and mouth. It can for example be used when calculating the eye map and mouth map [8]. Figure 4 illustrates the results with the YC_bC_r color space.

As can be seen the results were rather poor as the method was unable to fill the holes in some cases. Even though no further attempt was made with the method regarding skin segmentation, it was still useful during the feature extraction process.

3.1.3 Extract eyes and mouth

When the outline of the face has been extracted it is necessary to find the face's features. Features like eyes and mouth can be used to align the face. This section will describe the method used for finding the eyes and mouth.

3.1.3.1 Eye map

The eyes are found by separating the chrominance and the luminance color components and constructing two different eye maps which are combined later on. The chrominance eye map is based on

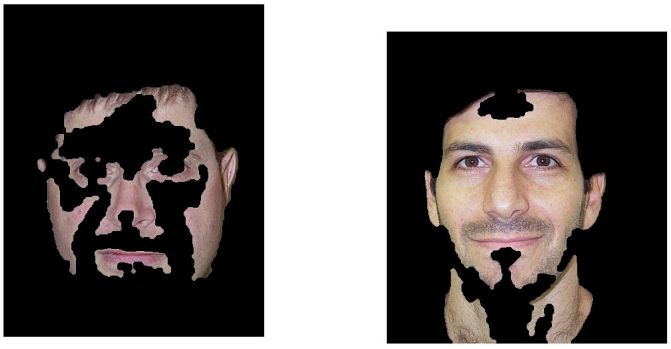


Figure 4: Skin segmentation based on YC_bC_r method

the assumption that C_b is high and C_r is low in the eye regions. In equation 5, the values C_b^2. $(\widetilde{C}_r)^2$ and $\frac{C_b}{C_r}$ are normalized to the range [0, 255] or [0,1]. $\widetilde{C}_r$ is the negative of C_r, 1 - C_r or 255 - C_r [8].

$$EyeMapC = \frac{1}{3}\left\{(C_b^2) + (\widetilde{C}_r)^2 + \frac{C_b}{C_r}\right\} \tag{5}$$

The luminance eye map aims to find the eyes assuming that the eye region contains both bright and dark spots. Morphological operators as erosion $\ominus$ and dilation $\oplus$ is used together with a structure function to create the luminance eye map, Equation 6, C_b^2 [8].

$$eyeMapL = \frac{Y(x,y) \bigoplus g_\sigma(x,y)}{Y(x,y) \ominus g_\sigma(x,y)} \tag{6}$$

(a) Chrominance eye map (b) Chrominance eye map with histogram equalization (c) Luminance eye map

Figure 5: Separate eye maps and the combined

The chrominance eye map is enhanced by a histogram equalization to distributed the intensities of the pixels more and the two maps are combined by multiplication [8]. In Figure 5 the different eye maps are displayed.

The combined eye map is filtered by finding the maximum intensity and multiplying it with a factor, in this case 75%, and removing every pixel-value with a intensity below. The images are then converted to binary images and dilated to make remaining areas expand.

Pixel values outside the eye region, found by the Viola-Jones algorithm, will be set to zero (black). The result can be sees in figure 6.

(a) The combination of eyemapC with histogram equalization and eyemapL

(b) Better segmentation but the hair still gets included

(c) Segmentation Image combined with detected eye region

Figure 6: Before and after filtering the combined eye mask

3.1.3.2 Mouth map

The mouth map can be found by converting the RGB color image to the color model YC_bC_r [9]. The lips contains a stronger red component and weaker blue component. This can be done by using Equation 7 and 8.

$$MouthMap = C_r^2 \cdot (C_r^2 - \eta \cdot C_r/C_b)^2 \tag{7}$$

$$\eta = 0.95 \cdot \frac{\frac{1}{\eta}\sum_{(x,y)\epsilon FG} C_r(x,y)^2}{\frac{1}{\eta}\sum_{(x,y)\epsilon FG} C_r(x,y)/C_b(x,y))} \tag{8}$$

The chrominance component C_b^2 and $\frac{C_b}{C_r}$ are normalized to the range [0, 255] or [0,1] depending on if the image's pixel values are in *uint8* or *doubles*, the parameter η is the ratio between the average C_b^2 and the average $\frac{C_b}{C_r}$,

(a) Original

(b) Mouthmap

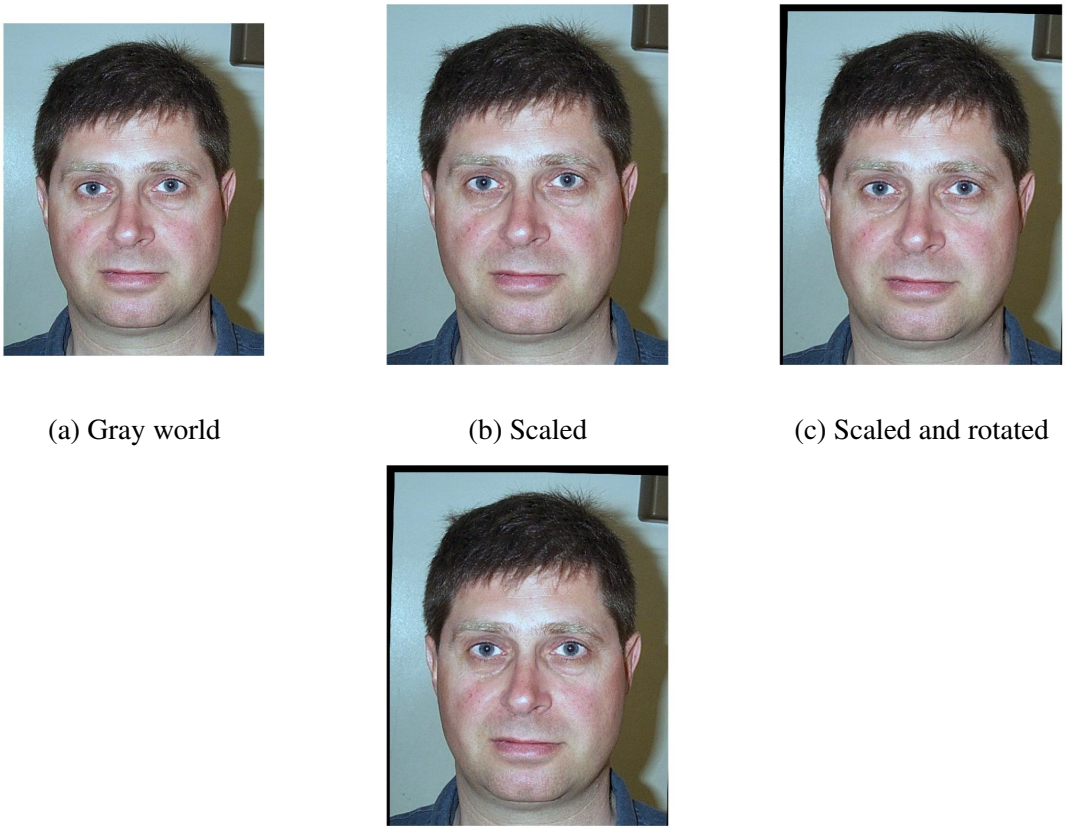
Figure 7: Transformation to align the image

The result of the equation can be seen in Figure 7. The mouth's pixel values became brighter than the rest of the face and is therefore easier to find than the rest of the image when a segmentation is done.

3.1.4 Face alignment

To get a reliable face recognition, the image that is compared to the database, needs to be aligned the same way, or as much as possible, as the corresponding image in the database. It is therefore necessary to align the face by using linear algebra and transformations. The used transformations are scaling, rotation and translation.

The scaling is used to get the same length between the eyes in all the images, rotation is used to align both eyes on the same height and the translation is used to translate the middle-point between the eye to the center of the image.



(a) Gray world (b) Scaled (c) Scaled and rotated

(d) Scaled, rotated and translated

Figure 8: Steps to align the image

Figure 8 illustrates the transformation pipeline from the normal image to the combined transformation of scaled, rotated and translated.

3.2 Face recognition

The recognition is performed by extracting and comparing important features from the images. The features that have been analyzed are intensity and phase from the discrete Fourier transform. The intensity analysis is performed by an algorithm called Eigenfaces, and the phase analysis is performed by Local Phase Quantization. Generally, the features are extracted in every face and then compared to an average face. This average face is calculated through Equation 9, where m is the number of images in the database. The feature difference between each face and the average face is calculated for every image in the database by Equation 10 and stored in a matrix F. Every column in F contains the j^{th} element of x_j and represent how much each face differs from the average.

$$f_a = \frac{1}{m}\sum_{j=1}^{m} f_j \tag{9}$$

$$x_j = \frac{1}{\sqrt{m}} f_j - f_a \tag{10}$$

The eigenvectors for the covariance matrix $C_f = FF^T$ are used to find the best subspace for representing the faces. Due to the large number of data, the dimensions have to be reduced in order to find the eigenvectors. The reduction will also make the procedure more efficient. The reduction can be done with Singular Value Decomposition or Principal Component Analysis.

If the eigenvalues and the eigenvectors are sorted decreasingly, the first eigenvector points in the direction where the faces varies the most. The corresponding eigenvalue is measuring the variance in this direction [10]. The last eigenvector points in the direction where the faces varies the least. Therefore the first eigenvector is the most important one, and the last is the least important one.

3.2.1 Singular Value Decomposition - SVD

Singular value decomposition is a form of factorization of a real or complex matrix, in this case the F matrix. The singular value decomposition of a matrix F with the dimensions $m \times n$ is $F = U\Sigma V^T$, where U is a $m \times m$ matrix and V is a $n \times n$ matrix. U contains the eigenvectors, and V the eigenvalues, of the covariance matrix $C_f = FF^T$. Σ is an $m \times n$ diagonal matrix containing the non negative singular values $\sigma_j, j = 1, ..., min(m, n)$ sorted decreasingly. The values that are the least accurate representations of the image subspace will be the smallest singular values.

3.2.2 Principal Component Analysis - PCA

Principal Component Analysis is a method for finding the principal components of data and is used to find a subspace for the faces. The eigenvectors are found by using $C'_f = F^T F$ as the covariance matrix instead. This matrix has the dimensions $m \times m$ instead of $m \times n$ which decreases the dimensions drastically and enables calculations for the eigenvectors. To find the eigenvectors for the actual covariance matrix $C_f = FF^T$, the eigenvectors and eigenvalues have to be multiplied with F.

3.2.3 Eigenfaces

The main idea behind the Eigenface recognition technique is to extract the intensity as the important feature from the faces and represent it as efficiently as possible, as eigenfaces. The representations are then compared instead of comparing the faces directly [11].

Assume that the images in the database have the dimensions $p \times q$ pixels when the preprocessing is done. All images are resized into arrays with dimension $n \times 1$, where $n = p \times q$. The arrays are stored into a matrix, $m \times n$, where m is the number of images stored in the database. Since the dimensions of the matrix is large, the best subspace for representing the images has to be found. The reduced matrix represents the training set which is used to match input images to the database. This means that all images in the database might not be included in the training set, but they are sufficiently represented by the faces in the training set [11].

In order to create the training set, an average face has to be calculated as described above. The result matrix F spans a subspace with an unknown orthogonal basis [11].

To find the basis vectors the eigenvectors and eigenvalues of the covariance matrix FF^T have to be found. But the covariance matrix will have the dimensions $(n \times m)$ and therefore PCA or SVD has

to be used in order to find the eigenvectors and eigenvalues. The eigenvector matrix is reduced from $n \times m$ to $m \times m$ dimensions, but due to the minor variance of the last eigenvectors, the number of basis vectors could be further decreased. If the eigenvectors are resized into images, they are called eigenfaces.

3.2.4 Local Phase Quatisation - LPQ

The phase from the discrete Fourier transform is a blur invariant property which enables matching blurred faces in the database. The algorithm is insensitive to centrally symmetric blur, for example motion blur and out of focus [12].

A blurred image $g(x)$ can be described by a convolution between the original image $f(x)$ and the unknown Point Spread Function $h(x)$, as described in Equation 11 [13]. The discrete Fourier transform (DFT) of the blurred image is described by Equation 12. The magnitude and the phase of the Fourier transform are separated and only the phase is considered, Equation 13.

$$g(x) = f(x) * h(x) \tag{11}$$

$$G(w) = F(w)H(w) \tag{12}$$

$$\angle G(w) = \angle F(w) + \angle H(w) \tag{13}$$

If the Point Spread Function $h(x)$ is assumed to be centrally symmetric, $h(x) = h(-x)$, the Fourier transform is real-valued and $\angle H(w) = 0$ for all $H(w) \geq 0$. This proves that the phase is a blur invariant property [12] [13].

The discrete Fourier transform is computed over a neighborhood N_x of a pixel position x in the image $f(x)$. The Fourier transform of the small neighborhood is described in Equation 12. In the equation, $\mathbf{u}$ is a vector describing the size of the neighborhood centered in N_x [12] [13].

$$F(\mathbf{u}, \mathbf{x}) = \sum_{y \in N_x} f(\mathbf{x} - \mathbf{y}) e^{-j2\pi \mathbf{u}^T \mathbf{y}} \tag{14}$$

For each neighborhood, the result will be a vector with information about the phase,
$\mathbf{F_x} = [F(\mathbf{u_1}, \mathbf{x}), F(\mathbf{u_2}, \mathbf{x}), F(\mathbf{u_3}, \mathbf{x}), F(\mathbf{u_4}, \mathbf{x})]$. The vector is divided into real and imaginary parts and is then quantized using the quantizer in Equation 15, where $g(x)$ is the j^{th} component of the vector F divided into real and imaginary parts [12] [13].

$$q_j(x) = \begin{cases} 1, & if q_j(x) \geq 0 \\ 0, & otherwise \end{cases} \tag{15}$$

After quantizing the vector, there are eight binary coefficients that are represented as integer values in the range $[0, 255]$. They are used as the blur invariant LPQ labels. A histogram of the integers from all the local neighborhoods is composed and used as the relevant feature [13]. As when using the intensity as a feature, the dimension have to be reduced with PCA or SVD in order to perform the matching.

3.2.5 Match an image with the database

To match an input image with a gallery image the input image has to be processed through either the eigenface method or the local phase quantization method. These methods results in a set of eigenvectors and weights. The eigenvectors are the vectors that represents the subspace of which the gallery images lie in. The subspace will either be intensity variant or phase variant depending on the used method. There are sixteen eigenvectors that describe the subspace. The weights are calculated through multiplying the image in vector form with the eigenvectors. The weights are a result of how close each image is to each of the eigenvectors. So every image will have sixteen weights, which is the number of images in the database. The resulting weights of the input image, ϕ_i, will then be compared with every gallery image's weights, ϕ_g, through nearest neighbor, Equation 16.

$$\epsilon^2 = ||\phi_g - \phi_i|| \tag{16}$$

The smallest euclidean distance between the weights represents a possible match between the gallery image and the input image. To decide whether or not the image with the smallest euclidean distance is a match, or another face with similar face features, the distance is compared with a threshold. If the euclidean distance is above the threshold, the input image does not match any of the gallery images.

4 Results

The result from doing face recognition on the manipulate images from DB1 on the unchanged images in DB1, is shown in Table 1. The methods used are Gray world, Viola-Jones, HSV, eye map and alignment for face detection. For face recognition, both Eigenfaces and LPQ are tested. It has also been tested with and without skin segmentation to see if the result is better with or without it.

The different methods for face recognition has also been tried when comparing images from DB2 with images in DB1 with and without skin segmentation. The results can be found in Table 2 where DB1 has been used as the database and DB2 is match to DB1.

Table 1: Results when matching a manipulated image from DB1 with images in DB1

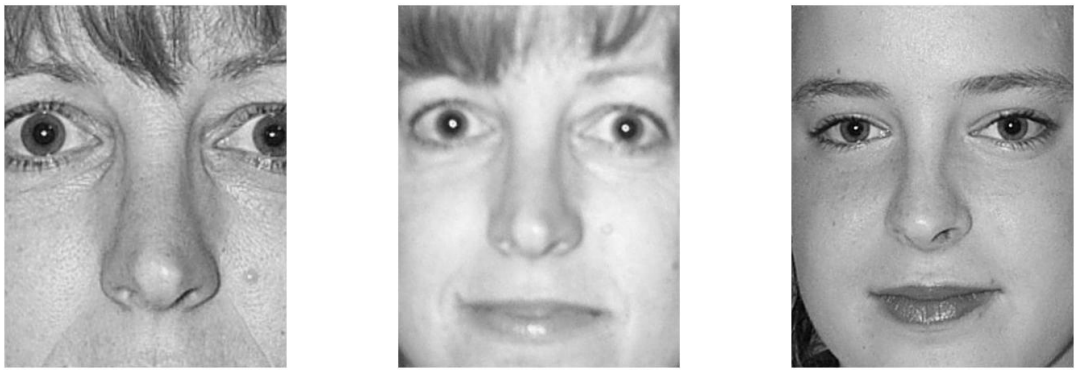
Applied change	Eigenfaces	Eigenfaces + skin segmentation	LPQ	LPQ + skin segmentation	Best case
No change	94%	94%	94%	94%	All
+30% tone difference	31%	-	69%	-	LPQ
-30% tone difference	13%	-	69%	-	LPQ
+5° rotation	94%	-	75%	-	Eigenface
-5° rotation	89%	-	69%	-	Eigenface
+10% scaling	50%	-	44%	-	Eigenface
-10% scaling	44%	-	0%	-	Eigenface

Table 2: Results when matching DB2 to DB1 with different methods

Image type	Eigenfaces	Eigenfaces + skin segmentation	LPQ	LPQ + skin segmentation	Best case
Blurred faces	22%	11%	67%	22%	LPQ
Cluttered background	50%	32%	44%	13%	Eigenface
Variant facial expressions	29%	29%	15%	14%	Eigenface with/without skin segmentation
Variant illumination	33%	0%	0%	0%	Eigenface
Total	34%	18%	32%	12%	Eigenface

5 Discussion

The matching rate could be further increased if the result from the face alignment was improved. The images in Figure 9 have been processed, which means that their faces should be aligned. The aligning works quite good, but after this step the images have to be cropped to equal sizes. The cropping discomposes the alignment in cases where the original image's format is very wide compared to the height, or very high compared to the width. The result of a bad cropping can be seen in Figure 9a. The blurred image, Figure 9b, is closest to another image in DB1, Figure 9c, which gives a false match. A possible solution for improve the aligning results would be to align the found faces instead of aligning the original images. The cropping algorithm could be improved by not let the cropping procedure depend on the image's original width or height.



(a) Original (b) Blurred (c) Matched

Figure 9: False match because of a bad cropping

There is no use of the mouth map in the current state. Possible area of use are combining the eye map and mouth map to translate the image with a different coordinate. The point between the eyes and the mouth is closer to the face center and could generate a more accurate recognition.

When matching blurred faces, LPQ gives the highest matching rate. This was the expected result since LPQ uses the Fourier transform phase as the important feature, which is a blur invariant feature.

With cluttered images, faces are easy to find. The reason for the low match rate is a result from the bad cropping. If the images have a long width, a lot of the background will remain and which makes the match more difficult.

For badly illuminated faces, or for variant facial expressions, there is a problem with detecting both eyes which disables the matching process. It can happen when one of the eyes has an intensity below the calculated threshold. If both eyes are found, the matching is often negative since no solution for variant illuminations has been implemented yet. Another possible solution to this problem would be to implement a better illumination normalization method, like Gamma Intensity Correction. The white balance method used by Gray World is not enough for the more illumination challenging images.

Several thresholds have been studied to find the most suitable one. However it is impossible to set a threshold that works for all images, the aligning accuracy has to be increased in order to increase the possibility to find a good threshold. An image which should not match sometimes has a smaller euclidean distance than an image that should match.

The result from the dimensional reduction using SVD or PCA of either the intensity images or the local phase quantization are sixteen vectors that describes the images' subspace. Some of the sixteen vectors describe the subspace better than others, therefore some of the vectors can be eliminated. This was not done during the project, but could easily be done. For an example when using eigenfaces, the

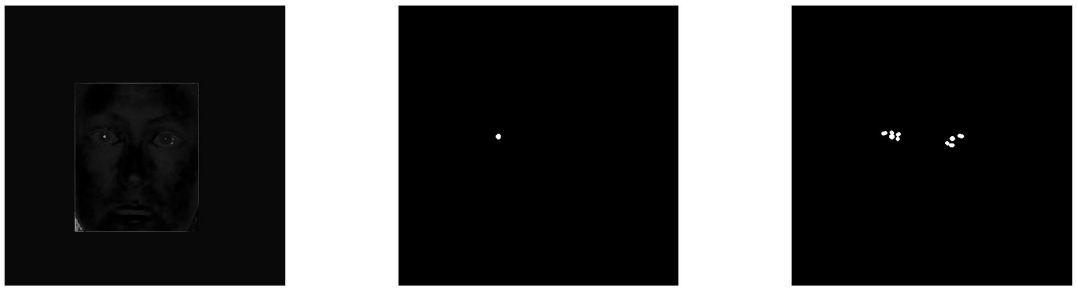
last two eigenvectors describes the image subspace poorly. This can easily be seen when comparing the last eigenface with one of the better eigenfaces, see Figure 10.

(a) A relatively good representation

(b) A bad representation

Figure 10: Eigenfaces that represent the image subspace

For one image, $db1\_11.jpg$, in DB1, the intensity is too low, which makes the eyes harder to register. That is why only 94% of the input images in DB1 is matched to the original one when the input image is unchanged, Table 1. All images in DB2 that correspond to this person will not result in a match. Image $db1\_11.jpg$, in DB1 gets an eye map with such low intensity values that the image is almost completely black, even in the eyes (Figure 11a). There is also a big difference between the eye's intensity. After the map is filtered (Figure 11b), the right eye disappears and it is impossible to align the face. This is also a problem for some of the images in DB0. One possible solution to the problem is to use histogram equalization on the original image. Figure 11b. Another problem with an image in DB0 is that its format is too small. The original image is smaller than the desired size after the cropping procedure.



(a) The right eye have low intensity

(b) One eye left

(c) Histogram equalization on original image

Figure 11: Right eye disappears after filtration and possible solution

Both recognition algorithms, Eigenface and LPQ, have their strengths and the most convenient solution would be to be able to combine them depending on the image type. LPQ decreases the matching rate distinctly when the query image is blurred or with a tone map of $(+/-30\%)$. On the other hand, Eigenfaces is preferred when the input image is rotated, scaled or has a cluttered background. LPQ has a problem with matching the images those cases. The query image is always closer to another image in the database.

Both PCA and SVD have been studied and SVD was chosen in the final project. SVD was considered as the most simplest one to implement. It also sorts the eigenvectors automatically and gave the best

visibly result when the Eigenfaces where studied. The matching rate could be further increased by improving the cropping procedure.

6 Conclusion

Viola-Jones is a reliable algorithm to find face- and eye regions, combined with the eye map a relative reliable face detection and features extraction is acquired. By improving the cropping and illumination normalization the success rate for matching can be increased.

Local phase quantization is a good approach when dealing with blurred imaged or different tone maps. Otherwise, the Eigenface algorithm is preferred. The best case would be to combine them depending on the type of image.

References

[1] Y. Taigman, M. Yang, M.A Ranzato and L Wolf, *DeepFace: Closing the Gap to Human-level Performance in Face Verification*, Facebook AI Research, Menlo Park, CA, USA and Tel Aviv University, Tel Aviv, Israel (2014)

[2] H. Liu , W. Gao, J. Miao and J. Li, *A Novel Method to Compensate Variety of Illumination In Face Detection*, Institute of Computing Technology Chinese Academy of Sciences, Bejing 100080, China (2002)

[3] S. Shan, W. Gao, B. Cao and D. Zhao, *Illumination Normalization for Robust Face Recognition Against Varying Lighting Conditions*, Institute of Computing Technology, Bejin China, Harbin Institute of Computing Technology, Harbin China (2003)

[4] J. Su, *Illuminant Estimation: Gray World*, `http://web.stanford.edu/~sujason/ColorBalancing/grayworld.html`, Accessed: 2015-12-10

[5] B. Wang, C. Liu, X. Chang, *Skin Detection and Segmentation of Human Face in Color Images*, Hebei University of Technology, Tianjin, China and Cangzhou Vocational College of Technology, Cangzhou, China (2011)

[6] J. Chaves-González , M. Vega-Rodíguez, J. Gómez-Pulido and J. Sánchez-Pérez, *Detecting skin in face recognition systems: A color spaces study*, University of Extremadura (2010)

[7] P. Viola and M. Jones, *Rapid Object Detection using a Boosted Cascade of Simple Features*, Compaq Cambridge Research Lab, (2001)

[8] R. Hsu, M Abdel-Mottaleb and A.K. Jain, *Face Detection in Color Images*, IEEE Trans. Pattern Analysis and Machine Intelligence Vol.24, No.5 (2002)

[9] S. Shan, W. Gao and D. Zhao, *Face Detection in Color Images*, IEEE, (2002)

[10] M. Turk and A. Pentland, *Eigenfaces for Recognition*, Massachusetts Institute of Technology (1991)

[11] N. Muller, L. Magaia and B.M Herbst, *Singular Value Deocmposition, Eigenfaces, and 3D Reconstructions*, Department of Applied Mathematics, University of Stellenbosch, Matieland, Stellenbosch (2004)

[12] T. Ahonen, E. Rahtu, V. Ojansivu, J. Heikkilä, *Recognition of Blurred Faces Using Local Phase Quantization*, University of Oulu (2008)

[13] V. Ojansivu and J. Heikkilä, *Blur Insensitive Texture Classification Using Local Phase Quantization*, University of Oulu (2008)