Face Recognition in Color Images using Matlab
TNM034, Linköpings University

Viktor Nilsson, vikni067@student.liu.se
Amaru Ubillus, amaub859@student.liu.se
Mirza Talovic, mirta980@student.liu.se

December 2, 2012
Abstract

The report presented will explain an approach for implementing face recognition using the software MATLAB. The developed algorithm can be divided into two main parts. Detection and recognition of faces. Input images are light compensated and converted to $YC_bC_r$ color space in order to efficiently isolate faces. By using thresholding and morphological operations, a person's eyes can be localized. When the eyes are detected, rotation and scaling can be applied in order to prepare the image for recognition. Classification and recognition of each image can finally be performed by an implemented eigenface method.
Chapter 1

Introduction

Explaining the process of face recognition to a computer is a complex task. Many years of research has resulted in numerous of methods for solving this particular problem, despite this, there are two main categories for face recognition:

- **Holistic based methods** treat the face and its different properties as one unit. Certain face properties are then extracted in order to distinguish faces.

- **Feature based methods** are based on facial characteristics and it divides the face into smaller areas, which later on are analysed. This method has several advantages when it comes to handling varying light conditions and different angles compared with the holistic approach. On the other hand this method has downsides when dealing with identification of closed eyes or mouth.

The presented algorithm is based on a holistic approach, since it already was known what kind of images that were supposed to be used, and which limitations the algorithm was allowed to have. Thanks to this information a generalization could be made, which led to the conclusion that the holistic method was the most suitable for these circumstances.

Some properties of the images used for the implemented method were known in advance. This knowledge can be very useful, since the algorithm can be adapted for these properties. The input images in this algorithm have simple background and homogeneous lightning. Also there is only one face per image, and it is always oriented towards the camera.
1.1 Purpose

Face recognition is a technology that can be applied and implemented in many parts of today’s society. Some areas of applications are [1]:

- **Biometrics:** Using a face as a biometric is proved to be a successful approach since it is the way humans recognize each other.

- **Identification Systems:** A face could be used to examine if a person exist or not in a list of individuals. Based on that information the system can allow respectively deny access.

- **Law Enforcement:** Face recognition technology can be used to increase performance of surveillance and law enforcement.

There have been great achievements and progress in this particular field of study, but there are many challenges left to overcome. Still today, low accuracy is one of the main drawbacks of face recognition. Also considering that this technology can be applied in several important areas is making it an appropriate technology to develop.

The purpose of this project is to create an face recognition algorithm that can recognize faces in manipulated images.
1.2 Background

This project is a part of the course TNM034 - Avancerad bildbehandling och bildanalys at Linköpings University.

The task was to write an algorithm that reads images from two different databases, images from database 1 should be allowed access and images from database 2 should be denied access. Images that do not exist in either database 1 or 2 should also be denied access. Each image has the same dimensions 256*256 pixels and they have all similar backgrounds and relatively homogeneous lightning.

The algorithm should meet the following requirements:

- Return ID for persons having their image in database 1.
- Return 0 for persons having their image in database 2.
- Return 0 for images that do not exist in the either database 1 or 2.
- Identify and recognize a face in a modified image. Modifications allowed:
  - Rotation (± 5°)
  - Scaling (± 10%)
  - Translation (± 20 pixels in x and y -direction)
  - Tone values (±10%)
  - Combinations of each modification

1.3 Structure of report

The implemented face recognition algorithm presented in this report can be divided into two parts. In chapter 2 the method for detection is discussed, this will cover an explanation of how a face is detected from a 2D image. In chapter 3 we clarify how recognition of a face is performed. After presenting the main structure of the algorithm, the performance of the implemented methods will be analysed. Finally, a conclusion can be drawn followed by a summary of the work presented.
1.4 Method

Our program is designed to detect and recognize faces in given images with limited conditions. Figure 1.1 and 1.2 briefly covers the most important steps of the implemented algorithm.

Figure 1.1: Face detection work flow.

Figure 1.2: Face recognition work flow.
Chapter 2

Detection

2.1 Light compensation

According to the given task the implemented method should be able to handle modifications in tone values. And since varying illumination can degrade the performance of the algorithm significantly, a light compensation function is implemented. The function is designed to compensate the light in the input image according to [2]. 5% of the brightest pixels in the image are set to as reference white, then all pixels are linearly scaled to 255.

2.2 YCbCr color space

One challenge that arrises when dealing with skin-detection is which color space is most suitable to use. Some color spaces have advantages over others but when it comes to real world applications, where we typically have dynamic scenes it is preferred to choose a color space which easily allows one to separate the luminance from the chrominance (since a human face have certain important color features).

The color space is changed in order to take advantage of the skin color properties of a human face. The two chrominance channels in figure 2.1 can later be used to isolate a persons skin and create a skin-mask as one solid object representing the skin. Similarly eye regions can be extracted by using knowledge regarding the color properties of human eyes. The benefit of using skin color filters is that it significantly reduces the number of pixels to search through in order to find pixels corresponding to eyes.
The luminance information is stored in the Y component and the chrominance is separated into the \( C_b \) and \( C_r \) components. Where \( C_b \) is the blue chrominance and \( C_r \) is the red chrominance as shown in figure 2.1. Skin regions and eyes can be extracted using information in the chrominance components \( C_b \) and \( C_r \). High levels of \( C_b \) values and low levels of \( C_r \) values are especially found in eye regions.

To convert an image from RGB colorspace to \( YC_bC_r \) equation 2.1 is used.

\[
\begin{bmatrix}
Y \\
C_b \\
C_r
\end{bmatrix} = \begin{bmatrix}
65.481 & 128.553 & 24.966 \\
-37.797 & -74.203 & 112 \\
112 & -93.786 & -18.214
\end{bmatrix}
\begin{bmatrix}
R \\
G \\
B
\end{bmatrix} + \begin{bmatrix}
16 \\
128 \\
128
\end{bmatrix}
\]  

(2.1)

Converting the image to \( YC_bC_r \) color space ensures that the algorithm still maintains robust for both dark and light skin-colors. The reason for this is that the skin-tone color depends on the luminance. For varying illumination, a better performance is obtained by using \( YC_bC_r \) compared with the RGB color-space [3].

### 2.3 Face mask

To detect the face a good skin segmentation is an important first step in many intelligent systems like face recognition. The basis of skin detection is that human skin color can be clustered in limited regions. By analysing the color information in the histogram of the three RGB channels they are not very well separated. By converting to \( YC_bC_r \) the luminance can be separated from the chrominance. Earlier studies [3] indicate that the chrominance is independent from the luminance, which can be used to separate the skin color. To localize the outline of a face the chrominance components are thresholded as follows:

\[0.31 < C_b < 0.75\]  

(2.2)

\[0.56 < C_r < 0.70\]  

(2.3)

Equation 2.2 and 2.3 indicates where the face pixels can be found and are used to separate them from non-skin pixels. Morphological operations are used to extract skin-regions and eliminate noise. The result is a facemask which can be used to remove everything except skin-regions. This is done by applying the
facemask to the original image and set the pixels inside the interval as showed in equation 2.2 and 2.3 to white and the rest of the pixels to 0 as showed in figure 2.2.

![Facemask applied to the image](image)

**Figure 2.2: The facemask applied to the image**

### 2.4 Eye localization

#### 2.4.1 EyeMapC

To localize the position of the eyes EyeMapC [2] is used. The main idea behind EyeMapC is based on the characteristics of human eyes in the $YC_bC_r$ color space. The eye map is built from the chrominance components and is based on the fact that higher $C_b$ values exist around the eyes in comparison to the $C_r$ values. The equation for the EyeMapC is shown in equation 2.4

$$EyeMapC = \frac{1}{3}((C_b^2) + (1 - C_r)^2 + (\frac{C_b}{C_r}))$$ (2.4)

The formula is designed to brighten pixels with low $C_r$ value and high $C_b$ value. The result from EyeMapC is shown in figure 2.3 and was finally dilated, eroded and masked. A threshold value was set to separate the brightest eye pixels from the rest.
2.4.2 EyeMapL

Human eyes usually contain both bright and dark pixels in the luminance component. These pixels can be emphasized by using grayscale morphological operations, such as dilation and erosion. The equation used for creating EyeMapL is presented in equation 2.5:

$$EyeMapL = \frac{Y(x, y) \oplus g_\sigma(x, y)}{Y(x, y) \ominus g_\sigma(x, y) + 1}$$  (2.5)

Where the luma component of the image is $Y(x,y)$. $g_\sigma$ is a structuring element and $\oplus$ and $\ominus$ correspond to the morphological operations dilation and erosion. The result from EyeMapL can be seen in figure 2.4.
2.4.3 EyeMapCB

According to Ali Atharzfard and Sedigheh Ghoani [5], histogram processing has a good effect on localizing eyes. By preforming histogram equalization of the $C_b$ color component strong values are found especially in eye regions, which can be seen in figure 2.5.

![Figure 2.5: EyeMapCB](image)

2.4.4 Combinations of eyemaps

To achieve a greater performance EyeMapC and EyeMapL are combined to form EyeMapCL as shown in figure 2.6. The reason for this is mainly to reduce noise around the eyes and use strength from both of the methods.

![Figure 2.6: EyeMapCL](image)
2.5 Thresholding

A result where the eyes contain the overall brightest pixels as seen in figure 2.8, is very useful. By applying a threshold, everything except the eyes can be eliminated. The threshold is set where the difference between two samples in the histogram (figure 2.7) is sufficiently large. Since each image have a unique histogram, the threshold value will differ from image to image.

![Modified histogram plot](image)

Figure 2.7: Modified histogram plot for one of the eyemaps. From 0.9 (brightest) to 0.1 (darkest).

After finding a suitable threshold value, the image is thresholded and the result can be seen in figure 2.8 Only the pixels above the current threshold value are remaining.

![EyeMapC combined with facemask](image)

Figure 2.8: EyeMapC combined with the facemask after applying the threshold value.
2.6 Eyemask

To minimize the risk of identifying false eyepixels a subarea is selected from the image, in which eyepixel-candidates are searched for. Modifications that could effect the image have been considered when deciding the size of the subarea in figure 2.9 which proves that the eyes will always be located within the green area.

![Figure 2.9: The green rectangle corresponds to the subarea in which the eyes can be found.](image)

To increase the performance of the eye localization the maximum and minimum distance between the eyes in all images have been measured. These distances form a interval where the eyes can be found. When two eyes are found, a check is performed to ensure that the found eyes are positioned within the measured interval. If two pixels outside the interval is found, they are removed and the search for new eye-candidates continues.
2.7 Hybrid method

The eyemaps discussed in section 2.3 will sometimes fail to locate the eyes for all of the pictures. Inspired by a report from Muhammad Shafi and Paul W. H. Chung [4] a hybrid method is used. To overcome the weakness of each individual eyemap they are combined as illustrated in figure 2.10.

![Figure 2.10: EyeMapC, EyeMapCL and EyeMapCB are combined to make our final eyemap.](image)

As can be seen in figure 2.10 unwanted connected regions which are detected by one of the eyemaps but not by others are removed due to the AND operation. The next step is to combine the three new images using an OR operation. By using this method the combined eyemap become more stable and the probability to lose an eye due to rotation, scaling or translation become much smaller.
2.8 Rotate and crop the face.

The location of both eyes are necessary to correct rotation and align the eyes so that they both lie on a horizontal line, figure 2.11. Since the position of each eye is known, it is possible to calculate the angle by using equation 2.6:

\[ \theta = \sin^{-1} \frac{a}{c} \]  

(2.6)

Where \( a \) is the vertical distance between the eyes, and \( c \) is the horizontal distance, calculated using Pythagoras theorem.

![Figure 2.11: Aligning the eyes to a horizontal line by applying rotation.](image1)

To crop a face only one eye location is necessary, from that it is possible to use the distance between the eyes as a measurement to crop the image. The image is cropped as in figure 2.12.

![Figure 2.12: The measurement used for cropping a image.](image2)
Chapter 3

Recognition

3.1 Eigenface method

At this stage the faces can be detected and generalized. To be able to recognize a face, the mathematical process Principal Component Analysis (PCA) is implemented in a method called Eigenface method. PCA is a statistical method that decreases the dimensionality of the data, but at the same time maintains the variation.

Eigenface method is a holistic matching method based on the fact that information regarding a face is stored in so-called eigenfaces. These are then compared with other eigenfaces to find a matching face.

3.2 Requirements

To efficiently use the eigenface method the following requirements are needed to be fulfilled:

- Image must be sufficiently illuminated.
- The persons should be facing the camera.
- The face can’t be covered.

Varying lightning conditions be a problem since it can cause shadows on the face, making it harder to extract important face characteristics and therefore harder to recognize the person.

These problems can be avoided or reduced by ensuring that the faces in input images are directed towards the camera, have the same dimension and having similar lightning conditions. If these requirements can not be satisfied, the eigenface method is probably not suitable since it efficiency decreases significantly. Another method should therefore be considered.
3.3 Eigenface algorithm [6]

The eigenface algorithm is used to recognize a face once it has been detected. The images have the same size (100*150 pixels) and the eyes in the approximately same positions after rotation and cropping have been applied, see figure 3.1.

![Figure 3.1: Out image from the face detection part.](image)

For simplification we put all the pixels in a one-dimension vector instead of the image matrix, creating $\Gamma$. We do this for all images $I_i$ which give us $\Gamma_i$.

An average face is then created as seen in equation 3.1, where $M$ is the number of images.

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i \hspace{1cm} (3.1)$$

![Figure 3.2: Average face of database 1, $\Psi$.](image)
Now that we have the average face we can find the features that is unique for each face relative to the image given. To extract the features $\Phi$ in a face we have to subtract the average face $\Psi$ from all the images as seen in equation 3.2.

$$\Phi_i = \Gamma_i - \Psi$$  \hspace{1cm} (3.2)

Figure 3.3: The unique features $\Phi$.

To be able to calculate the eigenvectors $v_i$ the matrix $L$ has to be created, as seen in equation 3.3 with dimensions $(M*M)$.

$$L = \Phi_i'\Phi_i$$  \hspace{1cm} (3.3)

The highest eigenvalues $u_i$ are then calculated as seen in equation 3.4 to 3.6.

$$Lv_i = \lambda v_i$$  \hspace{1cm} (3.4)

$$\Phi_iLv_i = \lambda \Phi_i v_i$$  \hspace{1cm} (3.5)

$$u_i = \Phi_i v_i$$  \hspace{1cm} (3.6)

Now the $k$ number of weights $w_k$ can be created, as seen in equation 3.7.

$$w_k = u_i'\Phi_i$$  \hspace{1cm} (3.7)

When implementing this for all images a face space $\Omega_{i,k}$ is generated as seen in equation 3.8.

$$\Omega_{i,k} = \begin{bmatrix} w_1^1 & w_1^2 & \cdots & w_1^i \\ w_2^1 & w_2^2 & \cdots & w_2^i \\ \vdots & \vdots & \ddots & \vdots \\ w_k^1 & w_k^2 & \cdots & w_k^i \end{bmatrix}, i = 1, 2, \ldots, M$$  \hspace{1cm} (3.8)
Now that the facespace of eigenfaces from the database is created the identification of a face can be made. Repeating the presented process but instead with a single image and with the same average face $\Psi$ will result in the eigenface $\Omega^I$.

To find what face the input image is closest to an euclidian distance function is used, equation 3.9.

$$e_r = \min ||\Omega - \Omega^I||$$  

(3.9)

The closer the weights $w$, are to one each other, the more likely it is that it is the same face on both images as can be seen in figure 3.4. If the degree of similarity $e_r$, is below a certain threshold, the face on image is considered being identified and classified as a known face. On the other hand if $e_r$ ends up above the threshold, it is not identified and therefor classified as a unknown face.

Figure 3.4: Comparison of $\Omega^I$ and the first five eigenfaces of $\Omega$. 

17
Chapter 4

Results

To analyze the performance of the implemented method, some benchmarks have been runned. These are presented in the table 4.1.

Table 4.1: Performance of the implemented algorithm for different modifications.

<table>
<thead>
<tr>
<th>Modification</th>
<th>Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>100.0 %</td>
</tr>
<tr>
<td>±5° degrees</td>
<td>88.5 %</td>
</tr>
<tr>
<td>±10% tone value</td>
<td>86.5 %</td>
</tr>
<tr>
<td>±10% size</td>
<td>90.0 %</td>
</tr>
<tr>
<td>±5° degrees, ±10% tone value</td>
<td>80.8 %</td>
</tr>
</tbody>
</table>

The tests have been done in ± for all modifications as well as a combination of rotation and tone modification which consists of four separated tests. The tests are done on all images from database 1 and all images in database 2. With the eigenface database created from database 1. For database 1 a correct id returnation is considered a success whereas for database 2 a returnation of 0 is considered a success.

Some images that neither exists in database 1 nor 2 have been tested with the above modifications with 100% success rate. One of the images tested is shown in figure 4.1.

Figure 4.1: An image excluded in both databases.
Chapter 5

Discussion

We have seen that there exist components that are crucial to good performance. Ensuring the use of qualitative images in the database is one parameter that facilitates better performance. Other parameters that is harder to control is that the eyes are not closed and that the orientation of the face is always directed towards the camera.

When it comes to real world applications, this type of algorithm would be more suitable to use in controlled environments. Thus, it would be more suitable to use for personal identification applications such as drivers licenses and passports rather than outdoor surveillance. The main reason would be due to the inherent limitations of the algorithm itself.

A distinct benefit of using the eigenface method is speed and computational power, which makes it a good choice when requiring a fast identification process. On the other hand, the method has several limitations. In some cases, there can be a larger difference between two images of the same person, than between two different individuals. We've also discussed in section (lightcompensation) that lightning conditions have a large impact of the performance of the algorithm. Additionally lightning conditions is often hard to control in all environments, this indicated that the eigenface method best suits face recognition performed in controlled and limited environments.

The reason for choosing the eigenface method for this particular algorithm is the fact that we in advance had important information regarding input images. Assumptions en generalizations where therefore possible to make in a early stage of the project.

The implemented algorithm has a reasonably good performance according to the task given, and we are satisfied with the final result. The project has given all group members insight into the world of digital image processing and face recognition.
Chapter 6

Future work

Future work would be concentrated on improving light compensation, morphological operations and optimizing the threshold for the eigenface method. Thus, a greater precision in both detection and recognition of faces would be obtained. An interesting approach would be to combine our holistic method with a feature based method to achieve higher performance.
Bibliography


