IMPLEMENTATION AND COMPARISON OF FACE RECOGNITION ALGORITHMS

Submitted by Tania Pouli
for the degree of
BSc (Hons) Computer Software Theory
of the University of Bath
2007
IMPLEMENTATION AND COMPARISON OF FACE RECOGNITION ALGORITHMS

submitted by Tania Pouli

COPYRIGHT

Attention is drawn to the fact that copyright of this dissertation rests with its author. The Intellectual Property Rights of the products produced as part of the project belong to the University of Bath (see http://www.bath.ac.uk/ordinances/#intelprop). This copy of the dissertation has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the dissertation and no information derived from it may be published without the prior written consent of the author.

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of Bachelor of Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Signed:

This dissertation may be made available for consultation within the University Library and may be photocopied or lent to other libraries for the purposes of consultation.

Signed:
Acknowledgements

Firstly, I would like to thank my project supervisor Dr. Peiyi Shen for his valuable guidance and advice that made this project possible.

I would also like to give thanks Dr. John Collomosse and Dr. Peter Hall for answering several of my random questions throughout this project as well as Dr. Marina De Vos for all of her support.

Finally, I would like to thank my parents for always being there for me and all of my friends for putting up with me and keeping me sane this year.
Abstract

Face recognition is a part of Computer Vision that has attracted the attention of researchers across various disciplines for more than 30 years. A number of methods have been developed, using a wide variety of techniques. The aim of this project is to investigate these different methods, giving a detailed analysis of the process and problems behind face recognition. Three representative methods are implemented and evaluated with respect to their relative performance and the quality of their results under different circumstances. These methods are the popular Eigenfaces technique, a geometric-based, feature-oriented method and a simple template matching technique. In addition, a detailed account is given on face detection methods. Finally, possible improvements to the system are considered and future directions of face recognition research and applications are discussed.
Contents

List of Figures 5

List of Tables 8

1 Introduction 9

1.1 The Face Recognition Problem 9

1.2 Applications 10

1.3 Dissertation Outline 10

2 Literature Review 12

2.1 The Face Recognition Problem 12

2.2 Applications of Face Recognition 13

2.3 Psychophysical Background 14

2.4 Overview of Face Recognition Methods 16

2.4.1 Method Categorisation 16

2.4.2 Face Detection and Segmentation 18

2.4.3 Feature Extraction 18

2.4.4 Template Matching 20

2.4.5 Statistical Methods 21

2.5 Evaluation Methods 22
3 Project Planning and Requirement Analysis

3.1 Project Planning .................................................. 25
3.2 Risk Analysis .................................................... 26
3.3 Resources ......................................................... 26
3.4 Requirements Elicitation ........................................ 27
   3.4.1 Existing Applications ...................................... 28
   3.4.2 Evaluation Protocols ...................................... 29
   3.4.3 Use Cases .................................................... 30
3.5 Requirements ..................................................... 30
   3.5.1 Functional Requirements ................................. 30
   3.5.2 Non-Functional Requirements ............................ 32
   3.5.3 Project Requirements .................................... 33
   3.5.4 Requirements Overview .................................. 34

4 Face Detection ....................................................... 35

4.1 Locating the Face ............................................... 35
4.2 Face Detection Using Haar-like Features ..................... 37
4.3 Detailed Detection Using ASM ................................ 39
   4.3.1 Training ..................................................... 39
   4.3.2 Searching .................................................. 40
   4.3.3 Conclusions ............................................... 41

5 Normalization .......................................................... 43

5.1 Intensity Normalisation ...................................... 43
5.2 Size Invariance .................................................. 44
6 Face Recognition

6.1 Eigenfaces

6.2 Geometric Features

6.3 Template Matching

7 The Application

7.1 Design

7.1.1 Face Detection and Feature Extraction

7.1.2 Face Recognition

7.2 Implementation

7.2.1 Face Detection

7.2.2 Eigenfaces

7.2.3 Template Matching

7.2.4 Geometric Features

7.2.5 The User Interface

8 Evaluation

8.1 Testing Plan

8.2 Hypotheses

8.3 Test Results

8.3.1 Class a - Normal Conditions

8.3.2 Class b - Illumination Variations

8.3.3 Class c - Expression Variations

8.3.4 Class d - Expression Variations

8.3.5 Class e - Pose Variations
8.3.6 Class f - Pose Variations ........................................... 75

8.4 Results Discussion .................................................. 76
  8.4.1 General Performance ............................................. 77
  8.4.2 Performance Under Variations ................................. 78

9 Conclusion ............................................................. 85
  9.1 Critical Analysis ................................................... 85
  9.2 Future Work ....................................................... 86
List of Figures

2.1 The inverted image on the left is normal while the one on the right has the eyes and mouth inverted. .................. 15

2.2 When the image is viewed in the correct orientation, the difference between the normal image and the one with the inverted features is more noticeable. .................. 15

2.3 Example of a face detection method .................. 19

2.4 Feature extraction through Active Shape Models. A model of the object is iteratively deformed until it best matches the object. .................. 20

2.5 The manifold of different faces will not be linear. However, a small number of dimensions will be sufficient to describe it. Taken from [24] . 22

3.1 The process flow of the FaceVACS system [23] .................. 28

4.1 The same person under different lighting conditions .............. 36

4.2 The four features used by Viola and Jones for the detection of faces [53] 37

4.3 A typical contour plotted on a face. .................. 39

4.4 A typical contour point is displayed. The top image is zoomed in, showing clearly the intensity variation between the face and the background 40

4.5 An example of the same image at several pyramid levels .................. 41

5.1 In-plane rotation ............................... 45

5.2 Increasing the angle of rotation makes recognition more difficult . . . . . 46

6.1 Images are converted to column vectors, representing a point in a the high-dimensional image space .................. 49
6.2 Faces in the image space ........................................... 50

7.1 A general overview of the system ................................. 54

7.2 Active Shape Models - Training and Searching stages .......... 56

7.3 The training stage of the Eigenfaces method .................... 57

7.4 Feature points on a face ........................................... 64

7.5 The user interface .................................................. 65

8.1 A 3-pixel image can be converted to a point in 3-dimensional space. The same can be applied to a larger number of pixels. ......................... 69

8.2 As it can be seen, when the query image is identical to an element of the database, recognition is easy. ................................. 77

8.3 Even though both the pose and expression are almost identical, the illumination changes in these two images would greatly affect the recognition of the one using the other as the query image. ......................... 79

8.4 A sample of the results related to illumination. As it can be seen, the Geometric method successfully recognised the individual. The other two methods however are affected by illumination variations ............... 80

8.5 Overlapping the two images it is easy to see that most of the face looks the same. .................................................. 81

8.6 The results for this test show that the Template matching and Eigenfaces approach are not greatly affected by expression variations. The Geometric-based method on the other hand does not perform as well. ................................. 82

8.7 Objects appear different under rotation. The same is true – although less visible – for faces and it can affect the performance of face recognition methods. .................................................. 83

8.8 As it can be seen in the results for this test, none of the methods returned the correct result. However, as it is especially clear through the Eigenfaces results, individuals with similar poses were returned. ......................... 84

8.9 Distances between the features and the outline of the face are distorted when the face is rotated, making feature-based recognition very difficult. ................................. 84

1 Class a - Normal Conditions: Query image 1_2 ................. 92

2 Class a - Normal Conditions: Query image 2_1 ................. 92
3  Class a - Normal Conditions: Query image 2_5  
4  Class b - Illumination Variations: Query image 1_1  
5  Class b - Illumination Variations: Query image 2_2  
6  Class b - Illumination Variations: Query image 2_4  
7  Class b - Illumination Variations: Query image 2_6  
8  Class c - Expression Variations I: Query image 1_1  
9  Class c - Expression Variations I: Query image 2_1  
10 Class c - Expression Variations I: Query image 2_3  
11 Class d - Expression Variations II: Query image 1_1  
12 Class d - Expression Variations II: Query image 2_1  
13 Class d - Expression Variations II: Query image 2_4  
14 Class d - Expression Variations II: Query image 2_6  
15 Class e - Pose Variations I: Query image 1_2  
16 Class e - Pose Variations I: Query image 2_3  
17 Class e - Pose Variations I: Query image 2_6  
18 Class f - Pose Variations II: Query image 1_1  
19 Class f - Pose Variations II: Query image 2_1  
20 Class f - Pose Variations II: Query image 2_3
List of Tables

2.1 Areas of application for face recognition systems . . . . . . . . . . . . . . 14
2.2 Face Recognition Methods . . . . . . . . . . . . . . . . . . . . . . . . . . 17
2.3 Various correlation methods . . . . . . . . . . . . . . . . . . . . . . . . . 21

3.1 An overview of the major milestones for this project. . . . . . . . . . . . 26
3.2 Risk Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

7.1 The face detection function . . . . . . . . . . . . . . . . . . . . . . . . . 60
7.2 The image preprocessing function . . . . . . . . . . . . . . . . . . . . . . 61
7.3 Function for creating the model for Eigenfaces . . . . . . . . . . . . . . 61
7.4 Function for enrolling images for the Eigenfaces method . . . . . . . . . 61
7.5 Eigenfaces recognition function . . . . . . . . . . . . . . . . . . . . . . . 62
7.6 Function to insert an image to the template database . . . . . . . . . . . 63
7.7 The template matching function . . . . . . . . . . . . . . . . . . . . . . . 63
7.8 Distances used to classify different faces. Refer to fig.7.4 for details. . . 65
7.9 The functionality provided through the interface . . . . . . . . . . . . . . 66

8.1 The training sets used . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68
Chapter 1

Introduction

1.1 The Face Recognition Problem

A ubiquitous property of human perception is our ability to distinguish between different faces even when they look really similar and recognise hundreds of different individuals with almost no effort. Automated face recognition is an area within Computer Vision inspired by this ability. Facial recognition systems focus on extracting faces from static images and video sequences and deciding whether they belong to a database of known individuals.

Machine based face recognition can be divided into two main problems: locating the face in the image and comparing that face with the contents of a database of known individuals in order to obtain the best match. Before looking at specific methods for performing these processes however, one must understand the challenges behind automatic recognition of faces.

The difficulty in recognising or identifying a sample within a class of similar looking objects lies in the small structural variation between objects of the same class. For instance, distinguishing between a car and a human is a simpler problem as it is easier to encode our knowledge about what each of them should look like in our algorithms. However, when we try to identify different cars accurately – or in our case different individuals – the complexity of the problem increases.

In order to understand the problem of mechanical recognition of faces better, one must look at how a face can be encoded in a computer in a semantically meaningful way. Little variation exists between different faces; so little that variations due to different lighting conditions can be more important than differences between two individuals. Consequently, in order to successfully distinguish a face from a non-face object, and more importantly to recognise that face, it is critical to extract these small variations between faces as well as the general face structure.
1.2 Applications

Face recognition has numerous applications based on either the identification or the verification of the presence of individuals in still images or video sequences. From entertainment to security, the ability to automatically recognise an individual can simplify or improve a number of tasks.

According to Zhao et al. [56], the main areas of application of face recognition are Information Security, surveillance and law enforcement, security application based on smart card and of course entertainment.

- **Information Security:** There are numerous instances in our everyday life where we are required to input passwords or PIN codes to use a device or access personal data. While such forms of security measures are essential, they are not always desirable or user-friendly. Using a robust face recognition system, such applications would be able to identify the user and allow or deny access accordingly. Systems like a computer logging on to the user’s account automatically when the user approached or an ATM only allowing the owner of the card to withdraw money could all be made possible.

- **Surveillance & Law enforcement:** Video surveillance has become a part of today’s society. Although opinions about CCTV are varied, security could in several occasions be improved using an automatic face recognition system.

- **Smart card based applications:** Driving licences, passports and ID cards all require mugshots of the individuals to be taken under controlled conditions. Using a face recognition system, these mugshots could be used potentially simplifying processes such as airport checks.

- **Entertainment:** Potential applications could include video games that can recognise the player and load their preferences, human-computer interaction systems or even training applications that adapt to their users.

The above list is but a sample of potential applications that could benefit from face recognition functionality. However, although a number of systems and methods exist that provide satisfactory results, applications where automatic recognition of faces could really prove revolutionary, would require very fast and very accurate results with minimal input, something that most current methods cannot provide.

1.3 Dissertation Outline

The aim of this project is to give the reader a thorough understanding of Face recognition as a part of Computer Vision. The problems making this process difficult are analysed and a variety of solutions are suggested. Moreover, a representative set of recognition methods is selected and investigated in depth in order to evaluate the advantages and limitations of each.
The report starts with a survey on the existing literature on the subject performed by the author early through the project, which is followed by a section detailing the project plan and the requirements for an automated face recognition system. The rest of the report is structured based on the steps a typical recognition system would follow, namely face detection, normalisation and recognition. The next sections are focused on the design and implementation of the application as well as the evaluation of the implemented methods. Finally, the report is concluded with the author’s thoughts on the success of the project and possible future extensions.
Chapter 2

Literature Review

2.1 The Face Recognition Problem

Machine recognition of faces is the area of research focused on the identification of individuals from still images or video. The face recognition problem according to Zhao et al. can be described as follows: “Given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces.” [56]. This is achieved in a number of steps: firstly, a detection mechanism is used to locate the face in the scene, then the facial features are extracted and finally the identity of the face is recognised or verified.

Facial recognition techniques can be classified in 3 broad categories according to the parts of the face used in order to identify the person [56] [54]. Holistic approaches use information about the face as a whole in the recognition process, feature (or structural) based methods on the other hand rely on facial features and the geometric or structural relations between them. Finally, some techniques cross the boundaries between these two categories and are classified as hybrid methods.

Face recognition has been an active area of research for more than 30 years [56] attracting academics from several different disciplines. Several approaches have been suggested with different levels of success. One of the earliest attempts on machine recognition of faces can be attributed to W. Bledsoe and his work in Panoramic Research Inc in the late 1960’s [6]. Another study that helped form the basis of face recognition in computing was Kelly’s dissertation project where an automated facial recognition system was developed [30]. Parallel to that was Kanade’s study [29] which was one of the first attempts to create a facial recognition system that would only use one image per individual. No significant progress was observed during the 1980s on facial recognition. Although the basis was already formed, the technology was not yet capable of supporting an automated system that would be able to recognise individuals faster or more accurately than humans [13].

Interest in face recognition has grown since the 1990s, partly due to the technological advancements in software and in hardware as well as due to the number of potential
applications of such a system [11], [13]. This has resulted in the development of a number of methods as well as in several attempts to standardise the evaluation process for these methods [56], [54].

This project will look at several issues related to facial recognition such as its commercial applications and its influences from other disciplines such as psychology and neuroscience. Moreover, a number of approaches and different methods will be reviewed. Finally, the specific aspects required for a fully automated face recognition system will be discussed in more detail.

2.2 Applications of Face Recognition

The current state of technology makes it possible to create an automated face recognition system that could effectively identify large numbers of individuals without any human intervention. This can have several practical applications where being able to identify individuals only using their appearance even if no other personal data is available is essential.

A number of commercial applications of face recognition relate to security: user identification using controlled images combined with smart cards, law enforcement through video surveillance [56] or through shared image databases [45] where individuals that have caused problems in the past are flagged or even face recognition systems to restrict access to personal data. Other applications include parental control, entertainment such as video games [56] and photo categorisation similar to what Google is planning to implement in their photo library application, Picasa [4].

A detailed review of several modern commercial facial recognition systems was published on USA Today [47]. Three systems were reviewed in the article, the first was Acsys FRS Discovery, a system geared towards surveillance and user access control that used a technology named HNet (Holographic/Quantum Neural Technology) [3]. The second system was Cognitec’s FaceVACS-Entry that combined a feature-based approach with texture and illumination analysis. Finally, the last system reviewed was the one developed by Neven Vision, a company recently acquired by Google [47] [4]. This system included a face detector and feature extractor and used template based on the extracted facial features.

Although the performance of these applications is satisfactory in both video and still image tests, they rely on the cooperation of subjects in order to form templates. In uncontrolled conditions where position, angle, illumination and background can vary, performance drops significantly. As a result, it is clear that such technologies are not yet at the stage where they can be reliably used in order to locate specific individuals through video surveillance or as an alternative to other biometric methods for security such as fingerprint readers.

A table of typical applications is given by Zhao et al. in their survey on face recognition which summarises the several different domains where accurate recognition of faces by a computer could prove useful. Different areas where face recognition could be
Table 2.1: Areas of application for face recognition systems

<table>
<thead>
<tr>
<th>Areas</th>
<th>Applications</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entertainment</td>
<td>Video game, Virtual reality, Training programs, Human-robot-interaction, Human-computer-interaction</td>
<td>FaceIt Identix technology integrated in Nikon Coolpix. [16]</td>
</tr>
<tr>
<td>Smart Cards</td>
<td>Drivers’ licenses, Entitlement programs Immigration, national ID, Passports, Voter registration, Welfare fraud</td>
<td>Driver’s licence photo database used to prevent identity theft. [1]</td>
</tr>
<tr>
<td>Law Enforcement and Surveillance</td>
<td>Advanced video surveillance, CCTV control and surveillance, Portal control, Post-event analysis, Shoplifting, Suspect tracking and investigation</td>
<td>Dectel Face Recognition technology integrated in CCTV systems of Birmingham city center and Newham in London. [17]</td>
</tr>
</tbody>
</table>

applied have varying requirements regarding the accuracy of the recognition process. For instance, security critical applications would require a higher degree of accuracy compared to face recognition features implemented as part of a video game or some other entertainment system (Table 2.1).

### 2.3 Psychophysical Background

Face recognition as a human ability has always been a challenging problem in psychology and neuroscience. Humans use all available stimuli in order to distinguish between faces and as a result face identification is very accurate. However humans have memory limitations and consequently only a small number of individuals can be accurately remembered. Computers on the other hand can store almost unlimited amounts of data so it is up to the specific method to take advantage of the virtually unlimited storage in order to identify individuals as accurately as possible.

Numerous studies indicate that recognition of faces is a separate process from object recognition in humans. Gauthier and Logothetis [21] suggest that facial recognition uses different parts of the brain compared to recognition of objects. However, they note that a similar process takes place when humans are required to recognise any object that falls into a general category but small details exist that differentiate it (such as cars).
According to Bruce [10], face recognition cannot be strictly classified as a holistic process. Facial features, especially when they dominate the person’s appearance can help in the identification of an individual. However, the face is still considered an object as a whole as it can be demonstrated with an interesting experiment known as the ‘Thatcher Illusion’ where the features are inverted on the image of a face but the face is displayed at the correct orientation. When the image is rotated, the result looks more grotesque while it hardly seemed unusual before [50] (Fig. 2.1, 2.2).

Figure 2.1: The inverted image on the left is normal while the one on the right has the eyes and mouth inverted.

Figure 2.2: When the image is viewed in the correct orientation, the difference between the normal image and the one with the inverted features is more noticeable.

Another important issue in face recognition in humans is the importance of each of the facial features in the actual identification process. Studies have shown that the upper half of the face as well as features such as hair or the shape of the face are more significant than others such as the nose [56] [10].

Although face recognition in humans and computers cannot be directly related as it is difficult to simulate in a computer system the various stimuli used by humans in order to identify a person, the issues making face recognition such a challenging problem can give us some insight on how to approach this problem with computers.
2.4 Overview of Face Recognition Methods

During the past 30 years of research on face recognition, several approaches have been suggested. As suggested by Zhao et al. and Chellappa et al., the face recognition process for an automated system starts by locating faces in the input, extracts facial features and finally performs the actual recognition [1] [6].

Face recognition methods can use still images or video feed as input. However, this project will only focus on the first case as the techniques used in face recognition from video use some other aspects, such as motion, that are outside the scope of this paper.

The following sub-sections will provide a high-level classification of face recognition techniques and will then focus on face detection and segmentation, feature extraction and recognition through template matching offering an overview important work done for each of these areas.

2.4.1 Method Categorisation

Methods can be categorised according to the aspects of the facial image used in order to perform the identification [56]. Holistic approaches consider the whole area of the face and use that as a single object. Feature based methods on the other hand focus on specific facial features and the relations between them and finally some hybrid methods exist that use features but also take into account the shape and overall configuration of the face.

One of the most widely used holistic approaches was suggested by Kirby and Sirovich [31] known as eigenpictures uses principal component analysis in order to represent faces as economically as possible.

"In brief, we demonstrate that any particular face can be economically represented in terms of a best coordinate system that we term eigenpictures. These are the eigenfunctions of the averaged covariance of the ensemble of faces." [31]

An important advantage of this method is the remarkably compact representation of faces. A single eigenpicture will contribute to the representation of more than one faces. As a result the only data that needs to be stored in the database for each individual is a set of weights for every eigenpictures related to that image. Several methods were developed that were based on the ideas proposed by Kirby and Sirovich [31].

Several feature based methods exist either focusing on the actual features or on the geometric relations between them. In the first case, one of the techniques that can be used is template matching. A template for each of the selected features is stored in the database for each individual. In order to recognise a face, the templates are compared to the respective areas of the input image using an appropriate method such as correlation and the closest match is returned. In geometric methods on the other
hand, information such as the position and size of each feature are stored as well as the face outline are stored [11].

As it would be expected, there is a large number of hybrid methods, using various different techniques which makes it difficult to categorise them. The method suggested by Pentland et al. [38] and Quintilliano et al. [43] is based on Eigenfaces. However in this case in order to improve recognition even when parts of the face are not visible, facial features such as the eyes, mouth and nose can also be used in the same way.

The methods described above however are only a few samples of the variety of approaches to face recognition. Eigenfaces, template-based and geometric-based approaches have been some of the first methods that were developed and have formed the basis for a number of others. Other recent approaches include 3D representations where the exact geometry each face is recorded [9], [34].

Numerous different techniques have been used in face recognition, taking advantage of different characteristics of facial images. However there are still several shortcomings in facial recognition systems, preventing them from being used as the only method for accurately recognising individuals in critical situations. Three major factors that can greatly influence the performance of most methods are illumination, pose variations and outdoor scenes [56].

<table>
<thead>
<tr>
<th>Holistic</th>
<th>Feature Based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenfaces</td>
<td>Face recognition through eigen-decomposition of faces [38]</td>
</tr>
<tr>
<td>Fisherface</td>
<td>Similar to Eigenfaces method but using FLD [7]</td>
</tr>
<tr>
<td>Template Matching</td>
<td>Templates of the whole face used to recognise individuals [5]</td>
</tr>
<tr>
<td>Support Vector Machines</td>
<td>Linear Classifiers used for face recognition [5]</td>
</tr>
<tr>
<td>3D Methods</td>
<td>3D representations of faces used [34], [9]</td>
</tr>
<tr>
<td>Feature Based</td>
<td></td>
</tr>
<tr>
<td>Template Matching</td>
<td>Templates of specific facial features used for recognition. Different weights can be assigned to features [11]</td>
</tr>
<tr>
<td>Geometric Based</td>
<td>The location, size and geometric relations between features are used [11]</td>
</tr>
<tr>
<td>Eigenfeatures</td>
<td>Eigenvector decomposition performed on specific features [38]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Templates</td>
</tr>
<tr>
<td>Eigenpictures</td>
</tr>
</tbody>
</table>

Table 2.2: Face Recognition Methods

According to several surveys on face recognition techniques [56], [54] although several systems exist that can accurately and quickly recognise individuals in controlled conditions, when these conditions change, the performance drops. Consequently, systems such as real time monitoring of live video feed cannot be achieved, limiting thus the
practical applications of face recognition. The reader is encouraged to refer to the 2000 and 2002 FRVT results [8], [39] for further information.

2.4.2 Face Detection and Segmentation

An important aspect for automated facial recognition systems is the detection and segmentation of faces in images. According to Yang et al. [54] face detection can be described as the process of locating regions of the input image where faces are present. Face segmentation takes this process one step further as faces are not only detected and located within a region but the exact shape of the face will be extracted from the background [35].

An effective method for face detection is described in [33], which combines skin color detection and template matching for face segmentation. A skin color model is built which can match different skin tones under varying lighting conditions. After a skin area is detected, the algorithm proceeds by counting the gaps in the segmented region. If multiple gaps are detected (at least two for the eyes), the area is considered a face and template matching techniques are then used to refine the detected area. The center of the area is calculated and a face template is applied and compared to that area in order to verify whether it is a face object or not. The algorithm returns an intensity image where gray values represent the likelihood of a pixel belonging in a skin area. A face template is then applied and aligned to the image to verify that the matched area is a face.

Another method based on templates for the initial filtering and Support Vector Machines for the actual face segmentation is described by Ai et al. [5]. An average face template is used to perform the initial face detection in the image. The filtered results are then passed to the SVM which, after training on a set of face and non-face samples, will locate faces in the images more accurately.

The methods described are only a few examples of the variety of available techniques [26] [54]. Several methods can be used individually or combined in order to locate and accurately segment faces from a busy background. According to Srisuk and Kurutah [48], a set of requirements are defined for an effective face segmentation algorithm. Firstly, correct face detection should be as fast as possible and false positives should be minimised. Moreover, the algorithm should be able to deal with cases where the edge of the face is partially obscured. The algorithm should result in an automated system where face detection occurs without any interference by the user. Finally, face detection should be successful not only in simple backgrounds but in busy ones as well, where other objects are in the scene.

2.4.3 Feature Extraction

Several face recognition methods focus on facial features in order to recognise an individual. Depending on the detection method chosen, it may be possible to locate a face in an image with great accuracy. However, in order to accurately locate features such
as the eyes or mouth within that face, more detailed analysis of the image is required.

According to Zhao et al. [56], there are 3 main types of feature extraction techniques:

- Methods that detect facial features using edge and line information on the face.
- Template based methods that use generic templates for each feature that needs to be located and try to match that template with a section of the face image.
- Structural methods that use information about the geometric relations of the features.

A template based approach suggested by Yuille et al. [55] uses parameterized templates for the eyes and mouth that are compared to the image using an energy function. As the template is moved across the image, the energy function will be minimised the closer we move to a match. Significant research has been done by Cootes et al. on methods using statistical shape models. Active Shape Models (ASM) can be applied to a face and are iteratively resized and translated in order to match the nearest edge. The location and size of features are known for the model, so as the parameters are changed, when the model is matched to the face, the location and size of the features on the face can be found [15].

The choice of feature extraction method depends on the requirements of each system. In a controlled environment where the size and pose remain constant, a simple template based method will be effective. However, on a real-time system where faces may be rotated or under varying lighting conditions a more flexible method should be used.
2.4.4 Template Matching

One of the problems that Computer Vision techniques are trying to address is that of identifying objects when their general shape and characteristics are similar and only small variations exist to distinguish between them. Several categories of objects exist where such a technique would be useful like characters, road signs and faces. An effective method for that is template matching [19].

Template matching is performed by comparing the input image with a predefined template image using an appropriate method. The template is usually an image that clearly represents the image we are trying to match or a smaller part from that image. The template is translated and compared to each part of the input image and the similarity between them is calculated [52].

More formally, a template matching algorithm uses images from two sets: the target which contains the known images and the query set containing the images that need to be matched and recognised [40]. Several different methods can be used in order to calculate the similarity of an image from the query set with one in the target set.

Template matching methods can be classified according to a number of parameters. Firstly, specific methods can be categorized depending on the type of template used. Templates can represent the whole face or distinct facial features [54]. In the second case, only certain features may be used if they are considered more discriminating than others [11]. Moreover, the size of the templates can vary. According to the same paper [11], recognition can occur even at image sizes as small as 36x36 pixels giving the benefit of a smaller database size. However, larger images can improve the recognition performance.

An important aspect of template matching techniques is the method used to compare the template with the input image. One technique suggested by Brunelli and Poggio [11] in their comparison of Geometric and Template based methods for face recognition is correlation. Another method suggested in [52] for calculating the similarity between the image and the template is by calculating the Euclidean distance between two areas on the images. This is defined by
A plain intensity image is used with no processing.

The intensity is normalised locally by dividing with the average local brightness.

The rate of change of the intensity is considered instead of the intensity.

The laplacian of the intensity is taken.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I$</td>
<td>A plain intensity image is used with no processing.</td>
</tr>
<tr>
<td>$I/ &lt; I &gt;$</td>
<td>The intensity is normalised locally by dividing with the average local brightness.</td>
</tr>
<tr>
<td>$D(I)$</td>
<td>The rate of change of the intensity is considered instead of the intensity.</td>
</tr>
<tr>
<td>$DD(I)$</td>
<td>The laplacian of the intensity is taken.</td>
</tr>
</tbody>
</table>

Table 2.3: Various correlation methods

$$E(m,n) = \sqrt{\sum_i \sum_j [g(i,j) - t(i - m, j - n)]^2}$$

where $g(i,j)$ is the test image and $t(i,j)$ is the template image. In simpler terms, the template is moved across the image and for each position $(m,n)$, the Euclidian distance is calculated. The position returning the lowest $E$ is thus the best match for this template [52].

Finally, in order to improve the performance of the algorithm intensity, scale and rotation normalization may be used on the image. Four variations for intensity normalization are given in [11] as can be seen in Table 2.3.

### 2.4.5 Statistical Methods

A popular class of techniques for face recognition is based on statistical analysis of the variations on faces [56]. By looking at a sufficiently large set of faces, common aspects that appear in the majority of them will be statistically more dominant. Similarly, variations that do not appear often will not be as important.

A number of methods have been developed using such a model in order to describe faces of different individuals [56]. Faces are typically analysed using techniques such as PCA [32] in simple cases or other non-linear techniques in order to determine mathematically commonly occurring variations.

A popular method belonging in this class is the Eigenfaces technique, first suggested by Turk and Pentland in 1991 [51]. According to this approach, the database of face images is decomposed to a set of Eigenvectors and their respective Eigenvalues. Using this representation, faces can be described using a small number of dimensions, which in turn simplifies the searching problem.

Eigenfaces however is not the only linear statistical method available for face recognition. Several variations of the original method exist either using probabilistic information to account for noise in the image [36] or focusing on specific facial features [38].

21
Other methods include multi-linear approaches where analysis of the faces is extended to cover variations in pose or illumination such as the Tensorfaces approach [46].

Linear methods are based on the assumption that the manifold created by considering each face as a point in a high-dimensional space is linear. This however is typically not true (fig. 2.5. Based on this fact, several non-linear methods have been developed. These methods can be viewed as a generalisation of linear approaches and can achieve more accurate results as the variations in faces are more accurately described.

![Figure 2.5: The manifold of different faces will not be linear. However, a small number of dimensions will be sufficient to describe it. Taken from [24]](image)

Concluding, statistical approaches provide quite accurate solutions to the face recognition problem. Using multi-linear or non-linear techniques variations not due to the differences between individuals can be also described [46]. Such methods try to extract common aspects on faces from the images and using that information they are able to simplify the problem of recognition of not only faces but other objects as well.

### 2.5 Evaluation Methods

In order to produce a useful face recognition system it is important to ensure that it can accurately recognise individuals. Face recognition, like any pattern recognition method, is a statistical process [56]. As a result, evaluation methods focus on the statistical distribution of successful or failed matches. Face recognition evaluation protocols combine large image databases with a set of evaluation methods to define how the images should be used. Algorithms need to minimize the number of false positives while achieving the largest possible number of correct matches.

A number of methods were suggested and reviewed by Zhao et al. [56]. The FERET protocol was created by DARPA (Defence Advanced Research Products Agency) in 1993 and is still funded in order to support research on facial recognition technologies
The evaluation protocol includes a database of 14,126 facial images in total. 1199 sets of individuals are provided including at least two frontal views and several variations respecting position and illumination. Some images are also provided where individuals pull their hair back or wear glasses.

The test is performed by giving each algorithm two sets of images, the target and the query set. The algorithm will then have to report the similarity between each image in the query set and the target set. The evaluation protocol is designed in such way so that each algorithm can use a different similarity measure.

Although FERET offers a standardised evaluation method for face recognition methods, it is intended for algorithms that only deal with controlled conditions. The images are obtained using a constant, plain background with small variations in lighting and pose. In order to evaluate some of the more recent face recognition technologies, the FRVT test was performed in 2000 [56]. This test involves three stages: a technology evaluation, scenario evaluation and finally operational evaluation [8]. Both the identification and the verification performance of the algorithms is tested in varying face angles and lighting conditions. The methods are also evaluated in outdoors lighting by comparing images taken outdoors with ones taken indoors such as the ones in the FERET database.

One of the most recent face recognition evaluations is the FRVT 2002 where more than 120,000 images were used. FRVT 2002 tried to evaluate the performance of methods with respect to the size of the database as well as demographics [39]. Key to this test was the HCInt (High computational intensity test). It was shown that although face recognition systems that use images taken indoors can achieve success rates of up to 90%, outdoors imagery still remains a challenge as the success rates were only as high as 50%.

The above evaluation protocols, together with a number of other databases of facial images such as the XM2VTS [12] or the face database provided by Yale [22] have contributed in the improvement of face recognition techniques. It is essential that new methods are tested using a standardised, controlled protocol so that meaningful comparisons of the different techniques are possible.

### 2.6 Conclusion

Face recognition has been an active area of research in the past 30 years posing a challenging problem for several disciplines. The problem of recognising human faces using an automated system is divided in face identification and verification. Face recognition systems can either use video as the input or still images. This project focuses on the latter.

Several methods have been developed, using a broad range of computer vision techniques. Face recognition techniques can be classified as holistic, feature based or hybrid. Holistic methods are based on information about the face as a whole such as the method developed by Kirby and Sirovich where eigenpictures are used in order to store information about the face [31]. Feature based methods focus on details of specific
facial features in order to distinguish between faces and finally hybrid methods use a combination of holistic information as well as features.

Face recognition, although still an open research area, has numerous practical applications varying from entertainment to security and law enforcement [56]. Several commercial systems are available, such as the system developed by Neven Vision based on neural networks, that offer satisfactory performance [47] in controlled environments.

In order to evaluate face recognition algorithms, several evaluation protocols and tests have been defined. The first one that formed the basis for subsequent tests is the FERET evaluation protocol and database, developed in 1993 by DARPA [41]. Through FERET, researchers can acquire free of charge a large database of facial images taken under controlled conditions. In 2000, the first FRVT test was performed in order to accurately compare the available face recognition systems [8]. The test was repeated with an expanded set of images in 2002, showing that although some algorithms can achieve successful recognition rates of up to 90% in laboratory environment, when outdoors images are used performance falls to 50% [39].

Concluding, face recognition has emerged through research in areas varying from neuropsychology to machine vision and image processing. Although a number of techniques are available coming from any of these disciplines, the technology has not yet reached the stage where a fully automated system that would function in real world conditions is possible. Pose and illumination variations, as well as outdoors backgrounds can greatly affect the performance of existing algorithms. Although machine recognition of faces offers the advantage of speed and virtually unlimited storage, it still remains an open research area as no method exists yet that can mimic the effectiveness of face recognition in humans.
Chapter 3

Project Planning and Requirement Analysis

Before starting the project, it was critical to decide what was possible within the available time and organise the project appropriately to allow for the best possible results. Due to the nature of this project it was possible to build the final system incrementally. Each of the three methods used was built separately and only after their completion they were integrated with the final system.

Part of the aim of this project was to perform a comprehensive comparison of different face recognition methods. As a result, considering each methods as in independent module during the development process was well-suited. As such, an incremental approach was followed where the final system was built in a number of discrete milestones. This approach provided a strategy that would allow the completion of as many objectives as possible even if something unpredictable occurred during the project.

3.1 Project Planning

A plan for this project was compiled before starting in order to ensure that the time allocated for each task would be sufficient. The plan was also used during the project in order to monitor the process of each task. A detailed plan of this project can be found in Appendix II.

A number of milestones were specified, considering each face recognition method as an individual component. Extensive research on the subject was carried out not only at the beginning of the project but throughout the development process. It should be noted that a substantial amount of time was allocated for the evaluation and comparison of the different methods as this was considered one of the most important aspects of this project.
<table>
<thead>
<tr>
<th>Phase 1</th>
<th>Literature Survey, Requirements specification and Project Plan</th>
<th>20 Nov 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 2</td>
<td>Core System Prototype, Face Detection</td>
<td>15 Dec 2006</td>
</tr>
<tr>
<td>Phase 3</td>
<td>Implementation of Additional Face Recognition Methods, Refinement of the Prototype</td>
<td>15 Feb 2007</td>
</tr>
<tr>
<td>Phase 4</td>
<td>System Review, Evaluation of the Face Recognition Algorithms, Completion of Documentation</td>
<td>30 Apr 2007</td>
</tr>
</tbody>
</table>

Table 3.1: An overview of the major milestones for this project.

### 3.2 Risk Analysis

Although a detailed project plan was devised, giving sufficient amount of time for the different tasks under normal conditions, it was essential to consider several risks that could hinder the project. The following table lists possible risks as well as suggested solutions that could minimise their effects.

### 3.3 Resources

This section specifies the various resources required in order to successfully complete this project.

- **Software:**
  
  Matlab. The system will be developed in Matlab (Mathworks).

  OpenCV Library. Certain functions from the OpenCV library will be used, the library is available online.

- **Hardware:**
  
  A laptop or desktop computer capable of running Matlab efficiently.

  Digital camera. It will be used to capture images of faces if required for evaluating the system.

- **Human Resources:**
  
  Meetings with the supervisor will be arranged every week.

- **Literature:**
  
  Relevant papers from IEEE, Siggraph, ACM and other journals and conferences. Available online or through the library.

  Computer Vision books. Available through the library and owned.
### Risk Analysis

<table>
<thead>
<tr>
<th>Risk</th>
<th>Probability</th>
<th>Contingency Plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change of requirements</td>
<td>Medium</td>
<td>The requirements will be reviewed and adjusted throughout the project</td>
</tr>
<tr>
<td>Illness</td>
<td>Medium</td>
<td>Buffer time is allocated in the project plan to cover for minor illnesses or other similar delays</td>
</tr>
<tr>
<td>Unrealistic prediction of time required for tasks</td>
<td>Low</td>
<td>The plan will be reviewed throughout the project and buffer time is allocated to deal with small delays</td>
</tr>
<tr>
<td>Implementation delays</td>
<td>Medium</td>
<td>Buffer time in the project plan allows for small delays in case of such problems</td>
</tr>
<tr>
<td>Hardware failures</td>
<td>Medium</td>
<td>All the work done will be backed up regularly in several locations</td>
</tr>
<tr>
<td>Software problems</td>
<td>Low</td>
<td>Ensure that the required software is available in more than one location</td>
</tr>
<tr>
<td>Resources are not available</td>
<td>Medium</td>
<td>Ensure alternative resources are located if required</td>
</tr>
</tbody>
</table>

Table 3.2: Risk Analysis

### 3.4 Requirements Elicitation

The aim of the system implemented was to evaluate several diametrically different face recognition methods and compare their strengths and limitations. This high-level requirement was the root of the requirement analysis for this project. A number of methods were used to help specify a complete and consistent set of requirements. Existing face recognition systems were reviewed in order to determine the factors making such systems usable in practical situations.

Moreover, it was understood that an important part of face recognition research is the various evaluations protocols available that provide researchers with a complete solution for testing their work. Evaluation protocols such as FERET and FRVT define to a great extent the state of the art in face recognition. As a result, as part of the requirement elicitation process, these evaluation protocols were investigated to help understanding what is required for an effective face recognition system.

Finally, some use cases were performed in order to acquire an interaction-oriented view of the system’s requirements. Several scenarios were investigated focusing on normal as well as unexpected conditions and inputs.
3.4.1 Existing Applications

**Cognitec FaceVACS**

The first application reviewed was the system developed by Cognitec, FaceVACS, one of the best performing commercial face recognition applications according to the FRVT 2002 test [39]. Cognitec provides several variations of its system for applications such as Entry Control, recognition from large databases and more.

The Cognitec system provides a fully automated recognition system accepting numerous inputs and being capable of both identification and verification of individuals. The process followed can be summarised in Fig. 3.1. This summary of the flow of the system provides some insight on tasks required by such a system.

![Figure 3.1: The process flow of the FaceVACS system [23]](image)

Automatic face and feature extraction functionality is provided by the FaceVACS system, an essential step in the recognition process. Moreover, images are normalised to deal with illumination, size and pose variations. Pose and size normalisation is achieved using information about the location of the eyes. Lighting variations on the other hand are reduced through histogram equalisation and contrast normalisation [23].

A feature-based recognition approach is used. Several local features of the face are extracted and transformed into a vector. This process is repeated for more than one images of each individual if possible to ensure that the largest amount of variation is covered during the enrollment stage.

As this system is aimed for commercial applications, it is essential that recognition is as accurate as possible. In order to achieve that, Cognitec focuses on normalisation and preprocessing of the image such that external parameters do not affect the recognition process.
Identix FaceIt

This solution provided by Identix, is a fully automated recognition system aimed at security and access restriction applications. The company provides several products varying from face detection and recognition to liveness challenge tests and is also one of the best performing applications in the most recent FRVT tests [39].

FaceIt uses a combination of face recognition techniques and biometric data in order to accurately recognise an individual. Firstly, geometric-based information is collected such as distances between facial features. Using that data, an initial search is performed limiting the search space. Then using skin biometrics, an accurate match is determined.

Due to the nature of the methods used, the FaceIt system is relatively insensitive to variations due to illumination or expression as well as cases where the individual has a different hair style or facial hair [27]. The system however provides recognition only under small rotation variations, showing that the methods used are affected by pose.

Reviewing the FaceIt system, some insight was gained on the importance of different variations in images and how they can affect different methods. Moreover, as skin biometrics are used in addition to the geometric-based method, it could be deduced that the geometric description may not be sufficient by itself for effective recognition.

3.4.2 Evaluation Protocols

A number of evaluation protocols have been developed in order to assist research in face recognition and define the state of the art. Such protocols suggest a number of tests that can be performed and usually provide large databases of face images that can be used in order to test systems in a standardised way.

Focusing on the image datasets used in FRVT 2002 [39], it can be seen that although a large set of images were taken under highly controlled conditions, the methods were also tested using outdoor images with variable backgrounds, varying illumination and other such noise. Once again, it can be seen that the capability of methods to recognise individuals under non-standard conditions is important.

Another important point in the FRVT 2002 test is the use of mostly frontal images. Several commercial applications were tested, some with exceptional recognition rates even under highly variable conditions. However, the lack of focus in recognition under pose variation could lead to the assumption that in practical situations, pose invariance underes small angles is sufficient while larger angles can simply be ignored, in contrast to illumination invariance which must deal with greater variations.

A number of other such evaluation protocols and tests exist. However, similar conclusions can be derived by the more recent FRVT 2006 test as well as the FERET evaluation protocol.
3.4.3 Use Cases

In order to investigate required system functionality under various different situations, a number of use cases were created. As use cases is an interaction based method, several different stages were considered where the user would need to give some input to the system.

Part of the aim of this project is the implementation of an automated face recognition system. As a result, interaction occurs mainly before the recognition process where the user selects an image to be recognised, and after the system produces the recognition results. It is thus essential that the requirements take into account different possibilities related to these forms of interaction.

Focusing on the input images, different situations that can occur include various file formats such as .jpg or .gif, different image sizes as well as colour or grayscale content. It can be seen that recognition is highly dependant on the given input. This leads to a number of conclusions. Firstly, a number of file formats should be handled by the system. Moreover, different sizes as well as both colour and grayscale images should be accepted. Finally, looking at the output based interaction with the system, a user may require more detailed information about results. In order to avoid cluttering the interface, it could be desirable to provide such detailed information on request.

3.5 Requirements

3.5.1 Functional Requirements

This section covers requirements related to the expected functionality of the system.

<table>
<thead>
<tr>
<th></th>
<th>Two or more sufficiently different methods for face recognition shall be implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>High</td>
</tr>
<tr>
<td>Details</td>
<td>1.1 A complete implementation of the Eigenfaces method shall be provided</td>
</tr>
<tr>
<td></td>
<td>1.2 A complete implementation of a Template-based approach shall be provided</td>
</tr>
<tr>
<td></td>
<td>1.3 A complete implementation of a Geometric-based approach shall be provided if possible</td>
</tr>
<tr>
<td>Rationale</td>
<td>This is part of the aim of the dissertation. Ideally, all three methods will be developed.</td>
</tr>
<tr>
<td>2</td>
<td>A user interface incorporating the different methods and their results shall be implemented</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>Priority</strong></td>
<td>High</td>
</tr>
</tbody>
</table>
| **Details** | 2.1 The interface shall provide functionality to select an input image to be recognised  
2.2 Results from each method shall be displayed clearly and ordered by the closest match |
| **Rationale** | The interface will allow the methods to be tested and evaluated. |

<table>
<thead>
<tr>
<th>3</th>
<th>Faces in images shall be detected automatically.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>Medium</td>
</tr>
</tbody>
</table>
| **Details** | 3.1 A method such as the Viola-Jones approach shall be used.  
3.2 An automatic feature detection method shall be developed.(Low) |
| **Rationale** | Automatic face detection will improve the usability of the system and will help provide more accurate results. Feature detection functionality would also be desirable but not essential. |

<table>
<thead>
<tr>
<th>4</th>
<th>Face recognition methods shall be able to identify a sufficiently large number of faces.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>High</td>
</tr>
</tbody>
</table>
| **Details** | 4.1 A professionally created database of faces shall be used in order to ensure that the evaluation of the different methods is as accurate as possible.  
4.2 The methods provided shall be trained (if appropriate) using the same database of faces. |
| **Rationale** | Using an accredited database in order to evaluate the methods will help making this evaluation more meaningful and accurate. Databases such as FERET [18] or the Yale’s Face Database [22] provide a variety of subjects, poses and illumination conditions. |

<table>
<thead>
<tr>
<th>5</th>
<th>The system shall provide detailed information about each individual result.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>Medium</td>
</tr>
<tr>
<td><strong>Details</strong></td>
<td>Details about the quality of the match as well as the image will be provided to help the evaluation process.</td>
</tr>
<tr>
<td><strong>Rationale</strong></td>
<td>While evaluating the different methods it will be useful to have as much information as possible to understand the strengths and limitations of each method.</td>
</tr>
</tbody>
</table>
6  |  Functionality for training each method shall be provided.
---|---
Priority | High
Details | 6.1 Depending on the method, it shall be possible to train it for a number of different images.
| 6.2 The training data shall be used for the recognition process.
| 6.3 The training process shall be simplified to make training with a large number of images possible.
Rationale | The quality of the training in methods such as Eigenfaces plays an important role to the recognition performance of the method. As a result, it is important that training is efficient.

3.5.2 Non-Functional Requirements

Non-Functional requirements are concerned with aspects of the application such as usability and performance or more generally, aspects not related to the functionality of the system.

7  |  The application shall run on Windows XP.
---|---
Priority | High
Details | 7.1 The application shall run on BUCCS machines.
Rationale | The application will be developed on a machine running Windows XP.

8  |  The application shall function correctly using Matlab 7 or more recent versions.
---|---
Priority | High
Details | The application will be developed under Matlab 7.
Rationale | This is the most current stable version of Matlab.

9  |  The system should be responsive and use processing resources efficiently.
---|---
Priority | High
Details | 9.1 In situations where the system performs intensive operations and thus cannot respond in real time, the user should be informed about the system’s current actions and progress.
| 9.2 The Matlab code should be vectorised where possible to improve performance.
Rationale | By the nature of the application, some operations will be intensive. However, Matlab provides several ways to improve performance. Where possible, such improvements should be used.
<table>
<thead>
<tr>
<th>10</th>
<th>The system should accept various image formats.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>Medium</td>
</tr>
</tbody>
</table>
| **Details** | 10.1 The system should be able to handle common image file formats such as .gif, .jpg and .bmp.  
|           | 10.2 The system should accept images of various sizes.  
|           | 10.3 The system should be able to handle both grayscale and colour images. |
| **Rationale** | This requirement improves the usability of the system and gives more flexibility as different image databases come in different formats. |

<table>
<thead>
<tr>
<th>11</th>
<th>The system should be usable and the interface should be easy to understand.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>High</td>
</tr>
</tbody>
</table>
| **Details** | 11.1 The interface should not be cluttered.  
|           | 11.2 Elements on the user interface should be clearly labeled and organised. |
| **Rationale** | A simple, clear interface will improve usability and will simplify the evaluation process. |

<table>
<thead>
<tr>
<th>12</th>
<th>The system should be able to handle unexpected inputs.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>Medium</td>
</tr>
<tr>
<td><strong>Details</strong></td>
<td>12.1 When unexpected conditions arise, the system should be able to handle them gracefully and inform the user about the problem.</td>
</tr>
<tr>
<td><strong>Rationale</strong></td>
<td>Although the probability of unexpected inputs is not very high as the system will be used mainly in order to evaluate the different methods, ensuring that non-standard conditions are dealt with is important.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13</th>
<th>The code should be readable and maintainable.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Priority</strong></td>
<td>Medium</td>
</tr>
</tbody>
</table>
| **Details** | 13.1 The code should be well-commented.  
|           | 13.2 The application should be coded following the Matlab coding guidelines.  
|           | 13.3 Functionality of the system should be clearly separated in different methods and m-files. |
| **Rationale** | The above requirements will make the application easily maintainable and more reusable. Separating the functionality clearly will enable other researchers to use some of the functions. |

### 3.5.3 Project Requirements

These requirements focus on the project in general rather than the application as a piece of software.
### 3.5.4 Requirements Overview

Based on the requirements, a number of general guidelines were derived that were followed throughout the design and implementation stages. The reason for that was to ensure that the system would be built around the requirements and it would thus follow closely the aims of the project.

These guidelines were outlined as follows:

- A set of different face recognition methods will be developed, demonstrating the variety of approaches available in this area of research.

- These methods will be built in an extensible and reusable way allowing them to be used for future research. Functionality for easily training the algorithm, enrolling a database of images and searching for a face will be provided for each method.

- Evaluation of the different approaches under different conditions not only to prove their correctness but in order to investigate their limitations and potential will be a central theme for the application.

- Parallel to the previous point, an automated system will be developed to the extent that this is possible, in order to demonstrate the different recognition methods.
Chapter 4

Face Detection

A crucial part of automated face recognition systems is their ability to successfully locate and extract faces from images. In order to achieve that without requiring user input, face objects need to be extracted from their background and the boundaries of these faces need to be detected.

The face detection problem can be divided in a number of different levels:

- Firstly, a face must be successfully distinguished from non-face objects. Locating faces in a busy image is a trivial task for human vision but it can prove challenging for a computer system. The computer’s understanding of what a face should look like is limited to the analysis performed on sets of pixels and as a result semantic segmentation of an image as performed by humans is very difficult – if not impossible. Consequently, methods focus on the characteristics of a face that make it different to other objects such as the color of the skin or the general structure of shadows on a face.

- Detecting a face in an image alone may not provide enough information in order to segment that face from its background. As a result, the boundaries of the face must be located in order to not simply locate the presence of a face but to extract it from the image.

- A large group of face recognition methods are based on facial features. Systems using such methods need to successfully locate and extract features such as the eyes or mouth from faces as well as the face itself.

4.1 Locating the Face

Real world images can contain busy backgrounds and several different objects making the amount of information available in them overwhelming compared to the slight variations between different faces. Most face recognition systems need to be able to extract faces from the image in order to effectively process them. Face detection at
the simplest level can be understood as the process of locating faces in images and extracting them from their background. Although it is a well defined problem in theory, variations in the image can pose several challenges:

- **Illumination.** Varying lighting conditions can greatly affect the appearance of a face. Details can be lost in darker areas or intense highlights. Moreover, characteristics of the face such as edges or brightness differences near feature points become less important when lighting conditions are more extreme. For instance, the same individual is pictured in figure 4.1 under varying illumination. Algorithms aiming to detect the face based on the general structure of shadows and edges might fail in the second image (Fig. 4.1).

![Figure 4.1: The same person under different lighting conditions](image)

- **Pose.** Face rotation, size variations as well as the difficulty to predict where faces could be located in an image are obstacles that face detection algorithms need to overcome. A rotated face may be easily recognisable by a human viewer, however with enough rotation, the structure of the face from the new angle is sufficiently different to make its detection difficult for a face detection system. Similarly, extra information may be required in order to detect faces closer or further away from the camera.

- **Expression.** In controlled situations, such as passport or driving license photographs, facial expressions are limited. However, in less strict conditions, different expressions must be taken into account when trying to locate a face in an image, especially when detection is based on facial features.

- **Occlusion.** Faces in images can be partially occluded by other objects in the scene, accessories such as hats or glasses or simply by a rebellious hairstyle. Effective face detection systems should be able to successfully locate a face even using partial information depending on the application. However, it is important in this case to decide what percentage of the face should be visible in order for an object to be classified as a face.

The above challenges alone can make face detection a difficult problem. However, in addition to them, most face detection or recognition systems would greatly benefit or even require fast detection of faces in images or video sequences.
Based on the classification defined by Yang et al. [54] detection methods can be divided in four main categories based on the techniques they use.

- **Knowledge based methods:** This class of methods tries to locate faces using information about what humans perceive as a face such as typical distances between features.

- **Feature invariant methods:** In this case, methods focus on elements of the face that stay invariant when pose, expression or illumination change.

- **Template matching methods:** This class of methods uses a number of templates for the face or for facial features and tries to locate a face by correlating each template with the given image.

- **Appearance based methods:** These methods use a model of a face created through training over a set of different faces.

In the following sections, the methods used in this project for locating faces and features will be discussed in more detail.

### 4.2 Face Detection Using Haar-like Features

Several different techniques can be applied in order to locate faces in images. They can be based on skin color information, the general structure of a face, or even templates of different faces. The method chosen for this project is based on Haar-like features and is developed by Viola and Jones [53]. Haar features, as used for the detection or classification of objects, can be understood as rectangles containing darker and lighter sections. Different features are applied on sections of the image. Depending on the feature shape, operations are performed between the pixels covered by the darker and the lighter sections to determine if the structure of the tested area matches the expectations of the algorithm.

![Figure 4.2: The four features used by Viola and Jones for the detection of faces [53]](image)

Features can have different shapes or sizes and they can be combined in different ways to describe more complex objects. The set of all possible combinations for features
of a certain size however is very large. Although features can be evaluated quickly, the number of calculations that would be required for the complete set would make this method impractical. To solve this problem, a weak learning algorithm based on AdaBoost [20] is used to select a small number of features that when combined can effectively describe a face. At each round, a single feature is selected from the complete set that gives the lowest error which forms a weak classifier. After looking through the possible features and rejecting a large number of them, the remaining ones are combined to form a strong classifier.

Classifiers are then applied to the image in a cascading manner in order to locate a face. At each round, a successively more complex classifier is used. If at any round a part of the image is rejected, it will then be ignored by subsequent classifiers, thus gradually limiting the search space. More complex classifiers are only applied to regions of the image with potential interest. The cascade application of classifiers greatly improves the performance of the method as the section of the image that needs to be searched is reduced at every round.

Another aspect of the Viola-Jones method that improves its performance is a representation of the image known as the Integral Image which is used instead of the intensity image. The Integral Image at each pixel can be calculated by the following formula

\[ ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \]

which is simply the sum of the pixels left and above each pixel. Using this representation of the image, features applied on different sections can be evaluated rapidly at various scales.

A number of reasons lead to the choice of this method as the primary face detection solution for this project. The Viola-Jones method can successfully detect upright faces in most situations without requiring any user input. This is critical for automatic face recognition systems as the correct detection and extraction of the faces in an image is the starting point of the recognition process. Furthermore, although fast detection was not essential in this situation, the almost real-time performance of this method gives it a clear advantage.

As this project focuses mainly on face recognition, an existing implementation of this method was used to provide the required face detection functionality. A well-known collection of various Vision algorithms is OpenCV, a C/C++ open-source library provided by Intel [28]. The face detection method provided with this library is an implementation of the Viola-Jones algorithm with a number of additional improvements. Training of the algorithm is already performed and the appropriate classifiers required for face detection are selected.
4.3 Detailed Detection Using ASM

A class of face detection methods is based on models representing faces. Models can be static, useful in cases where the exact size or shape of the object is known, or deformable. In the latter case, representations of the face can vary from contour snakes to statistical, deformable models. One of the methods considered for this project was Active Shape Models, developed by Tim Cootes [15]. Faces are represented by a set of points forming a contour around the outline of the face as well as several facial features. The algorithm is trained over a collection of images in order to create a statistical model of what a face is expected to look like. The model is then applied on the input images and is iteratively deformed to best match the face.

4.3.1 Training

In order to detect faces through ASM, the algorithm needs to be trained over a set of images that cover the variations of the face space. Points are plotted on each face to form the contours. The contours are then transformed and aligned to each other as well as the average model. Using this realigned set of contours, statistical analysis is performed in order to capture the variation in terms of shape and intensity at the selected points [25].

Figure 4.3: A typical contour plotted on a face.

In more detail, the normal along each point on the average aligned model is calculated and the intensity of the pixels across that normal are extracted and stored. This forms the average intensity profile which encodes knowledge about what that area of the face is expected to be like. For instance, the profile of a point on an edge of the face might have dark pixels closely followed by lighter ones (Fig. 4.4).

In order to improve the performance of the algorithm, intensity profile analysis can be
Figure 4.4: A typical contour point is displayed. The top image is zoomed in, showing clearly the intensity variation between the face and the background.

performed at multiple resolutions. An image pyramid is created where each level has successively lower resolution (fig. 4.5). The first level of the pyramid is the input image and depending on its size, an appropriate number of steps is chosen. The intensity profile is then computed not only for every point on the contour but for every pyramid level. As the resolution is lowered, details in the image are lost. Consequently, higher pyramid levels capture larger variations while levels approaching the input image are more sensitive to smaller variations in the image.

After the intensity profile is created, statistical analysis is performed over the aligned data in order to determine the acceptable level of variation for each point as well as the way it can vary. A number of eigenvectors and their corresponding eigenvalues are calculated that can then be used when searching for a face to limit how the model can change to fit the given image.

If an effective training set is used, the statistical model combined with the intensity profiles can provide enough information to detect a face in a new image.

4.3.2 Searching

After training the algorithm, in order to locate a face in an image, the average, aligned model is applied to the image. It is important at this stage to initialise this model appropriately. A number of options are available for positioning and rotating the model as required. The user can explicitly define initialisation parameters for the model or a less detailed method can be used to automatically detect the rough location of the face or an area surrounding the face. Using that information, the model could be successfully positioned and resized in order to better fit with the given image.

After the initialization process, the model is iteratively deformed based on the information collected during the training stage. In the simpler case where only a single resolution is considered, at each iteration, the algorithm will investigate the normal along each point of the contour and compare it with the intensity profile corresponding to that point. Then a new point is selected along the normal so that the error between the new location and the intensity profile is minimised. The displacement is limited by the statistical description of the face in order to ensure that the new location will still fit with the algorithm’s expectation about what is a face.
In a more complex situation, multiple resolutions will be used and the process described will be repeated for each level. Initially, coarser resolutions will be considered as they describe more prominent variations in the image such as sharp edges or generally dark areas. When the model converges to a new location in lower resolutions, the next pyramid levels will be used with the new model until convergence is achieved on the original image.

4.3.3 Conclusions

Active Shape Models is a flexible method that has applications not only in face detection but in fact the detection of any shape. A model capturing the statistical variation of the object of interest is created which is then used to define how the contour should deform to best match the new object.

A number of disadvantages however make this method impractical in certain situations. In order to form an accurate contour of a complex object a number of points will be required. For instance, to create a contour of a face and several facial features 40 or 50 points may be required. As the training process is to a large extent manual, the user will be required to select the appropriate points on a large number of images in order to ensure that the training set successfully covers the variations between faces. Clearly, this can be a laborious process.

Moreover, although the use of resolution pyramids can greatly improve the detection performance of the algorithm, the speed of the detection is hindered as the intensity profiles are compared for each point on each resolution level. In several occasions, detection and even recognition of individuals needs to occur almost real time for the system to be useful.

As already mentioned, this method is sensitive to the initialisation of the model. If
inaccurate initialisation is to be considered, then greater displacements along the normals would have to be allowed to enable the model to deform appropriately. However, in that case, it would be difficult to restrict deformations that escape from the expected face shape. On the other hand, if only small displacements are allowed, in cases where the model is not correctly positioned or resized, detection will be difficult.

Finally, important for the success of this method is illumination invariance. The deformation of the model depends on the intensity profiles collected through training. Hence, if the lighting conditions of the input image are very different to the ones used for the training set, it is possible that the algorithm will converge to an incorrect location.
Chapter 5

Normalization

Vision tasks involving comparisons between different objects of the same class can be thought of as search problems within that class – or space – or objects. As a result, variations not covered by that space can increase the difficulty of such problems.

Face recognition is one of these problems. Unknown faces are analysed and compared to a database of known individuals in order to find the best match. Ideally, high quality images would be available, where the individual would be facing the camera, baring a neutral expression. The lighting would bring out the face details without creating dramatic shadows or highlights and no other items or accessories would be present in the image. Assuming conditions such as these would greatly simplify the task of face recognition. Such an assumption however would be clearly unrealistic.

In any practical environment, achieving ideal conditions is almost impossible. Take for instance possible security applications of face recognition such as a monitoring system at a public area. Individuals would be moving and not always facing the camera, lighting conditions would change, some people could be wearing hats or glasses, have different hairstyles and so on.

Consequently, as such parameters cannot be controlled before the images are taken, it is important that they are minimised through pre-processing. Three main causes of variation will be covered in the following sections, together with representative solutions for each of them.

5.1 Intensity Normalisation

One of the major problems of face recognition is illumination. Even small variations in the lighting when a photograph is taken can adversely affect the appearance of a face. Although such variations may not be visible by the human eye, the changes in the image can be large enough to make the recognition of that face a difficult task. In realistic situations, where lighting may come from different angles, the structure of the face will appear even more different.
It is no surprise that as a result of this limitation, numerous methods have been developed for changing the illumination of a given face in order to simplify the recognition process. According to Zhao and Chellappa [56], illumination normalisation approaches can be divided into four categories:

- Heuristic methods
- Image comparison methods
- Class-based methods and
- Model-based methods

Heuristic approaches focus mainly on the global illumination of the face and use some simple, but effective, techniques. Pentland et al. [38] chose to normalise the contrast of the image while Poggio et al. [49] used histogram equalisation. Other approaches include a method closely related to PCA-based recognition where the first few Principal Components of the decomposed set of faces is ignored in order to remove variations due to illumination.

Image comparison methods are based on the comparison of information derived from the image such as edge maps, the rate of change of intensities or other representations of the image created through various filters. Other approaches use distance measures in order to evaluate the illumination but according to Zhao et al. [56] their performance is arguable.

Finally, model-based methods use 3D models of the face that can be acquired using PCA on shape-from-shading information [56]. Other techniques include relighting 3D face models using more appropriate light sources in order to improve recognition [42].

Various other techniques have been developed to compensate for illumination variations in images and improve the performance of face recognition systems and the above examples are only a sample of the variety of methods available. As variations due to illumination in images is one of the major problems in face recognition and the recognition of objects in general, research on the subject has been extensive.

5.2 Size Invariance

Faces in images occur in various different sizes, which should be taken into account when developing a face recognition system. Methods using direct comparisons such as template or geometric-based techniques are sensitive to size variations. As a result, the algorithms need to be trained for various different sizes or alternatively, the detected face in the input image may be resized as required.

The first approach can improve the accuracy of the results as artifacts due to resizing are avoided, however it requires multiple images of the same person which may not be feasible in practical situations. Resizing faces to the desired dimension may cause
the loss of some information, it is however straightforward and allows face recognition systems to deal with different sizes easily.

5.3 Pose Invariance

In real-world applications of face recognition, it is difficult to control the pose of the individuals depicted. Inputs may be collected through surveillance cameras or even mobile phones and chances of capturing a frontal, upright face are low. As a result, it is critical that face recognition systems can successfully identify an individual even when conditions are less than ideal.

It is easy to see however, that as a face is rotated, its structure will appear changed. In fact, even small rotations cause enough structural variation to significantly hinder the recognition process. This is one of the main problems of face recognition and a number of approaches have been proposed to solve it.

Figure 5.1: In-plane rotation

Pose variations can occur in-plane or out-of-plane. In-plane rotations occur when the face is rotated in 2D (Fig. 5.1) while out-of-plane rotation is in 3D such as when faces are looking upwards or sideways (Fig. 5.2). The first problem is the simpler one as a solution is as simple as detecting that rotation and transforming the 2D image appropriately. In the latter case however, the difficulty of the problem increases the more the face is rotated.

Methods that deal with the second type of rotation can be categorised depending on whether they use multiple images of the same individual under different rotations or just a single image. In the first case, methods vary from simpler approaches where the input face is simply compared with images of different rotation until the best match is found under all available poses. Other more complex approaches that use multiple images both during training and searching include the construction of a 3D model of the face from all the available images or the use of cones. Such processes are computationally expensive and thus not useful in time critical systems.
A number of methods exist that use multiple images under different poses in order to train the system but only require a single image during the recognition process. For instance, the eigenface method developed by Turk and Pentland [51] was extended such that a different eigenmodel was constructed for several degrees of rotation. Another similar method was developed by Cootes et al.[14] where images rotated at 90°, 45° and 0° were used to create a statistical model of the face that was able to adapt to given images.

Although multi-image based approaches can give very good experimental results, their main limitation in practical situations is their requirement for multiple images. In most cases, a single image of the individual will be available for training the algorithm, usually at small or no rotation. Consequently, several methods have been developed to improve recognition across pose variation using a single image. Rotated faces can be transformed to an approximation of their frontal view, or applied to average 3D models in order to estimate how they would look under various rotation angles and poses. Such methods can be simpler computationally and are more desirable as a single image is sufficient. However, their performance is adversely affected by larger angles of rotation as important information such as some features on the image is lost.
Chapter 6

Face Recognition

Computer Vision focuses to a great extent on the construction of models that encode our knowledge about objects in a way that can be understood and used by a computer. If we were to look at an image without being able to deduce any semantic information from it, only a 2-dimensional array of colours would be seen.

Arrays of colours, or more precisely numbers, is in fact all a computer can see in an image. In order to be able to process data of this form in useful ways, Computer Vision techniques exist that can deduce information about the content of an image in several different levels.

Low-level methods focus on local characteristics such as edges and textures that by themselves cannot explain the contents of the image. Mid-level vision techniques can provide more useful information; attempts are made to cluster data in meaningful ways, fit general models on parts of the image or decide the image in semantically sound parts. In order to be able to recognise distinct and complex objects however, higher-level techniques are required. Face recognition belongs in this latter class of problems and makes use of a variety of high-level vision techniques. Although still an open problem, several methods exist that can provide satisfactory results.

As already defined in previous sections, face recognition focuses on the identification of faces in still images or videos. This can be understood as finding the best match for a face within a database of known individuals, deciding whether a face belongs in a certain collection or finding a known face in a set of images.

Before any of that can be achieved, a model should be created, encoding our knowledge about facial objects as well as the allowed levels of variation before an object is classified as non-face. Through this model, it should be possible to represent different faces such that different individuals can be clearly distinguished and recognised. Moreover, variations between different individuals should be given more significance compared to variations due to other factors. And this is exactly where the difficulty of face recognition – or any other recognition problem – lies.

Creating a representation of an object that can satisfy the above requirements is not a
simple task. The main difficulty is deciding what aspects of the object and the image
containing it should be considered. The choice of characteristics to describe each face
depends on the method that will be used to compare them. It is essential however,
that these characteristics remain important even when other factors in the image vary.

Face recognition methods can be categorised in various ways based on the information
they use in order to recognise individuals. Methods can be holistic or feature-based
depending on whether the whole face is considered or only specific features. Several
hybrid methods have also been developed, making use of both types of information [56].
Further classification of methods is possible depending on the analysis they perform on
the images, which can vary from statistical to geometric to intensity-based.

The following sections will focus on the specific methods used for this project. The
methods chosen are listed below:

- Eigenfaces, developed by Turk and Pentland [51], a holistic method based on
  Principal Component Analysis
- A geometric-based method where distances between manually located features
  are used
- A template-based method where templates of different individuals are compared
  through normalised cross-correlation.

6.1 Eigenfaces

Images are represented on a computer as 2D arrays of numbers corresponding to the
color or the intensity value of each pixel. Rewriting this array as a single long vector,
an image of \( m \times n \) size can now be thought of as a point in an \( m \times n \)-dimensional space.

Repeating this process for a number of images results in a set of \( m \times n \)-dimensional
points. If all the images used depict objects of the same class, such as faces, it is
safe to assume that a large number of their pixels will be similar. Using the vector
representation, this simply means that several coordinates of each \( m \times n \)-dimensional
point will coincide. As a result, the set of points corresponding to a collection of faces
will form a manifold – or surface – of a lower number of dimensions than the original
points. Based on that observation, the vectors best describing that manifold could be
found, greatly reducing the amount of information required to represent a face.

The algorithm can be divided in a training stage where the database of faces is analysed
and an application stage where unknown faces are recognised. Training is performed
through a small number of steps:

1. The images are acquired and reshaped to form a set of column vectors \([X_1,X_2,...,X_n]\)
2. The average image vector is computed \( \bar{X} \)
Figure 6.1: Images are converted to column vectors, representing a point in a high-dimensional image space

3. Each image vector is centered at the origin by $[X_1 - \bar{X}, X_2 - \bar{X}, ..., X_n - \bar{X}]$

4. The covariance matrix of the set of newly aligned set of vectors is computed

5. The covariance matrix is decomposed to its eigenvectors and eigenvalues

The eigenfaces method provides a simple process for creating a model of the face, however not without some limitations. First of all, a Gaussian distribution of the images is assumed, which, although largely true, is still a strong assumption. Moreover, when faces are projected to their lower-dimension eigenspace it is assumed that this space – or hyper-plane – is flat. Finally, the greatest limitation of the algorithm as described is the complexity of some of the calculations. As an example, consider images of size 100 × 100. The size of the covariance matrix in this case will be 10000 × 10000, which clearly is a very large number. Decomposing this matrix thus becomes a very complex process if not simply infeasible.

The covariance matrix according to the above steps would be computed using

$$C = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})(X_i - \bar{X})^T$$
Decomposing $C$ to its eigenvectors and the respective eigenvalues would give $C = U L U^T$ where the number of eigenvectors would be equal to the number of dimensions of the image space. Images however only cover a lower dimension manifold within that space and as a result, a smaller number of eigenvectors is only required.

Letting $Z = [X - \overline{X}]$, the SVD of $Z$ can be taken such that $Z = U S V^T$. The outer product of $Z Z^T$ can then be rewritten as

$$Z Z^T = (U S V^T)(U S V^T)^T = U S V^T V S^T U^T = U S S^T U^T = U S^2 U^T$$

which is equivalent to the eigen-decomposition $U L U^T$.

Looking at the inner product $Z^T Z$, it is clear that its size will depend on the number of images which is typically much smaller. The inner product of $Z$ can now be rewritten as

$$Z^T Z = V S^2 V^T$$

where $S^2$ corresponds to the eigenvalues as before, containing however only a number of values equal to the number of images. In order to obtain the respective eigenvectors, it is sufficient to solve the equation $Z = U S V^T$ for $U = Z V S^{-1}$. This calculation results in a smaller set of eigenvectors, however, in most situations the number of images and thus of eigenvectors will be sufficient for giving an accurate description of the faces.

Figure 6.2: Faces in the image space
6.2 Geometric Features

Geometric-based methods have been historically some of the earliest approaches to the face recognition problem. Face recognition has been viewed by many as a pattern matching problem. Starting with Takeo Kanade in 1973 [29], several methods have been developed that use distances or other structural information about facial features to encode the overall configuration of the face.

The choice of features to be used depends on the specific method. However, four requirements – or guidelines – were suggested by Brunelli and Poggio [11]

- It should be easy to estimate the exact location of the features
- Features should be affected as little as possible by varying illumination
- Small changes to facial expression should not affect the appearance and structure of features
- The features selected should provide high information content, making it possible to identify someone using only that information.

In any feature-based approach, before information about the features is extracted from the image, it is important that the location of the features within the face is known. This is a process that can be performed manually by allowing the user to locate the required features on the face. However, this has the obvious disadvantage of requiring user input and as a result is not appropriate for situations where automation is a requirement. Alternatively, features can be located automatically using a variety of techniques, a more complex but certainly more desirable approach.

The main challenge of this method and in fact any feature-based approach, lies in the difficulty of achieving automatic and accurate feature detection. Some ways to do this were discussed in section 4.3, however several other methods exist. Brunelli and Poggio [11] for instance used template matching with templates of the features at different resolution levels. Correlation was first performed at coarser levels, gradually reducing the search space. Other methods include analysis of edge information from the images [29].

After obtaining the location of the features, a vector is formed containing distances or other structural information about features, such as relative angles. This vector is then used to represent the face. Assuming that the features selected describe effectively the overall structure of the face, recognition of an individual is possible using only that information.

A number of different techniques can be used to search for the best match for a face. One of the simplest approaches is considering each face vector as a unique point in a high dimensional space. Under this assumption, similar looking faces will have corresponding points closer to each other. The best match can then be decided as the one minimising a measure such as the Euclidean distance between the new face and some known face.
A different approach is decomposing the set of vectors representing the faces into their principal components. Each vector can then be represented as a linear combination of the eigenvectors describing the complete set. The most important variations are captured by the first eigenvectors while subsequent ones represent smaller changes. Unknown vectors can be associated with their nearest neighbour in order to find the best match. This method introduces some of the advantages of PCA into geometric-based approaches and can greatly improve performance.

6.3 Template Matching

Template matching is a method used in several Computer Vision problems. A template is a section of an image corresponding to what we are searching for. Templates are moved across the input image and compared to the section they cover using some similarity measure.

This is a very similar process to convolution with the difference that in this case the filter does not affect the image but is simply compared to it. Filters tend to have a stronger response when they are similar to the area they are applied to [19]. However a strong response can also be achieved simply from a bright area of the image.

To solve this issue, a normalised measure can be used to compare the image and the template. One such measure is the root sum of the squares of the values corresponding to the pixels where the filter is applied:

$$C(u,v) = \frac{\sum_{x,y}(I(u+x,v+y)-T_{u,v})T(x,y)}{\sum_{x,y}(I(u+x,v+y)-T_{u,v})^2 \sum_{x,y} T(x,y)^2}$$

This method is known as normalised cross-correlation and reduces the effect of illumination. However, although a significant improvement compared to simple correlation, applied to face recognition, it can be seen that illumination variations between the template and the target image can greatly affect performance. A number of methods for illumination normalisation are discussed in section 5.1.

Variation on the intensity of the image are not the only factor influencing the performance of this method however. Brunelli and Poggio [11] have used Gaussian pyramids of the image as discussed in section 5.2 to compensate for different image sizes and resolutions. Moreover, it is easy to see that even small rotations could change the apparent structure of the face making this method quite sensitive to pose as well.

Concluding, although template matching is a simple method that can give accurate results even at low resolutions, it is very sensitive to global variations on the image. Moreover, due to its nature, it is not a fast method, a characteristic unsuitable for real-time recognition systems where speed is required.
Chapter 7

The Application

7.1 Design

Due to the nature of the project, the design of the system was focused on providing a robust and usable interface where the different face recognition methods could be evaluated. The first step in designing the system was selecting the methods that would be implemented.

Face recognition has been an active area of research for many years and as it would be expected, a large number of different methods have been developed. The three methods chosen for this project were the Eigenfaces approach, Geometric-based features and Template matching. A number of factors led to this choice, the main one being their influence in face recognition research.

It was decided to implement each method independently in an evolutionary manner. Each method was built as a separate module using a standardised set of input and output parameters across all three algorithms.

The required functionality for each method was divided between training and searching part. After completing the implementation of the face recognition methods, the user interface was built to provide a simple and straightforward way of interacting with the system.

Throughout the design and development process, special consideration was taken in order to ensure that the system would be extensible and more importantly reusable. Keeping each method independent and providing an interface where different face recognition algorithm can be essentially ’plugged in’ in a standardised way would make the system potentially useful for future research on the subject.

This section will give an in-depth account of the complete design of the system. The structure of the section will follow the logical separation of the functionality of the system. An overview of the different parts of the application can be seen in fig. 7.1.
7.1.1 Face Detection and Feature Extraction

The Viola-Jones method

A crucial part of face recognition and important aspect of this project is the problem of detecting a face within an image. Automated face recognition systems need to be able to accurately locate a face without the assistance of a user.

A number of methods exist providing such functionality. After extensive research, the method developed by Viola and Jones [53] was chosen as the most appropriate according to the system requirements.

Face detection, although important for the correct function of the recognition methods, was not a central part of the focus of this project in the strict sense. As such, it was decided to use an existing implementation of this method to ensure the maximum amount of time would be allocated to the development and evaluation of the actual recognition system. The Viola-Jones algorithm is implemented as part of the OpenCV [28] vision library and is widely available.

Face detection functionality was required by all three method during both training and searching stages. Consequently, it was decided to implement this functionality separately so that it would be globally accessible across the system.

Active Shape Models

Although detailed feature extraction was not used in the final system for reasons discussed in following chapters, it was part of the initial system design. It was decided to focus on a contour based approach as the means of locating various features on the face.

The algorithm developed by Tim Cootes [15] was chosen as it could provide functionality for locating facial features as well as extracting geometric information about the face. Active Shape Models (ASM) is a method requiring extensive training to ensure
accurate detection and as a result it seemed necessary to provide simple and usable functions to make this process easier.

Some user input was required for the training if the method. More specifically, the user would need to plot a number of points on each image of the training set in order to define the contours. After plotting the required contours, the training data was analysed automatically and returned to the user.

While designing the training process for ASM, it was quickly realised that such a process can be quite time consuming. As a result, in order to reduce errors, the decision was made to automatically save data at each step. This decision was followed throughout the system, ensuring that tedious processes would not have to be repeated simply because of an accidental action.

After acquiring the training data, the ASM method required a number of steps to create the model to be used for the face and feature extraction. These steps are summarised in fig. 7.1.1 and were developed independently where possible to simplify testing.

In order to search for a face in an image, ASM uses the statistical model created during the training process and iteratively adjusts the model to match the new face. The design of the searching stage can be seen in 7.1.1 (right).

7.1.2 Face Recognition

One of the most important design decisions for this project was as explained the implementation of the three face recognition as independent modules. This project aims to evaluate different approaches for face recognition, providing on the same time a system that could be used to assist further research. A direct consequence of that was the decision to separate the interface and the actual recognition functionality as much as possible so that each algorithm could be used as part of the system as well as individually.

**Eigenfaces**

It was decided to focus on the Eigenfaces approach first as it is more recent and potentially a very effective recognition method. The functionality for Eigenfaces was divided between training and searching functions. The training part of the method was designed in a way that could simplify the process of creating an Eigenmodel as well as inserting individuals to the database. More details about the training stage of Eigenfaces can be seen in fig. 7.1.2.

One of the issues that became apparent while designing this method was the structure of the face database. It was decided to use a Matlab structure to store information about each individual. As personal information was not required, the filename for each image was stored as well as other relevant information depending on each method. It should be noted that although a database in the traditional sense was considered and could be a more desirable option, it was decided that the format described would be sufficient for the purposes of this project.
Figure 7.2: Active Shape Models - Training and Searching stages
Figure 7.3: The training stage of the Eigenfaces method
The searching part of the method was designed according to the initial decisions. More specifically, in order to enable the method to be used as an easy-to-integrate part of the system as well as individually, it was decided to only require an image as the input and load all the necessary data (such as training data and the database of known faces) from pre-specified `.mat` files.

**Template Matching**

This method, although based on a simple concept, gave rise to a number of questions. Template matching can be used locally, comparing templates of features of the face, or globally trying to match the face as a whole.

In the first case it is easy to see that global illumination will not affect recognition very much but variations due to facial expressions can play an important role. On the other hand, if templates of the whole face are used, expressions will not affect the overall appearance of the individual significantly. As face recognition methods will be evaluated mainly under pose and illumination variations, it was decided to use holistic templates as they would provide more useful information during evaluation.

The size of the templates was another issue that needed to be resolved before proceeding with the implementation of this method. Template matching across different sizes or resolutions is a computationally complex process. As a result, it was decided to resize the faces extracted after the face detection process to a specific size.

As mentioned, the method was expected to be sensitive to several variations. The decision was made to normalise the global illumination of the face to improve recognition. However this was only done without eliminating larger variations as that information could be useful for evaluating the performance of the method under such conditions.

Another important aspect of this method was the choice of similarity measure to be used when comparing the two images. The method used after considering the alternatives was normalised cross correlation as it could provide more accurate results compared to simple correlation.

Finally, it is worth noting that this method too was divided in an enrolment and a searching stage. The latter was designed in a similar manner to the searching part of the Eigenfaces algorithm while the former was focused on ease of use to simplify the training process.

**Geometric Features**

This method was designed in a similar way to both Eigenfaces and Template Matching. Training and searching functionality was again separated. The searching part followed the same guidelines in order to improve usability and easy integration with the rest of the system.

An important design decision for that method that should be discussed was the choice of feature points on the face. Recognition through geometric-based methods is performed using information about the face such as distances between certain features or their
actual sizes.

The choice of feature points was influenced mainly by Kanade’s [29] research. According to Brunelli and Poggio, features should be easy to detect and as informative as possible. The first requirement meant that feature point that could be easily hidden by a different hair style, facial hair or accessories were not appropriate. As such, it was decided that points such as the top or the sides of the head should not be used.

According to the second requirement, features providing useful information should be used. The features chosen were the following:

- Center of eyes
- Center of nose
- Sides of nose
- Center of mouth
- Corners of mouth

This selection was made is it could account for vertical, horizontal and diagonal distances between features giving a good description of the overall configuration of the face. Moreover, some of the information used included sizes of certain features that would stay invariant under different expressions.

Finally, concerning the structure of the database used in this method, a format similar to the other two methods was used. Geometric information extracted from the face was stored as a vector, associated to the path linking to the appropriate image.

### 7.2 Implementation

The application for this project was implemented using Matlab following requirement 8. This chapter will focus on the implementation details for each part of the project as well as the integration of the individual parts with the interface to form the completed system.

Chapters 4, 5 and 6 give a detailed account of the algorithmic logic behind different parts of the system. As a result aspects of the application that are simply a direct implementation of that logic will not be discussed in detail. Instead, details about the implementation that do not follow directly from that analysis or the system design will be explained.

Before developing the application, it was important to have a good understanding of the way images are handled in Matlab and of the platform in general. Matlab stores images as matrices. Intensity or grayscale images only require a single matrix whose dimensions match those of the image while colour images can be represented in a
number of ways. The standard RGB representation was used throughout this project where information about each colour is stored in a separate matrix. To determine the colour of a pixel, the colour values from each matrix are taken.

Throughout the application, grayscale images were used. As such, when a colour image was encountered, the `rgb2gray` Matlab function was used to convert it to grayscale.

### 7.2.1 Face Detection

As part of the final system, automatic face detection was provided. The Viola-Jones [53] method as described in Chapter 4 was used. Following the design decisions for that part of the system, the OpenCV implementation of the method was used.

Interoperability between C libraries and Matlab is possible through the use of `.mex` files. According to the instructions provided by Mathworks [2], the four components required for such a file are the following:

1. The `mex.h` header file which should be included at the top of the `.mex` file
2. A function called `mexFunction` with a number of input parameters
3. An array named `mxArray` as an input parameter to the above function, where the actual Matlab arguments will be stored
4. The API functions to be used.

A `.mex` file provided by A. Noulas [37] was used for this purpose. Details about the signature of this function can be seen in Table 7.1.

<table>
<thead>
<tr>
<th>Function</th>
<th>MATLABfacedetect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>A character array containing the filename of the image</td>
</tr>
<tr>
<td>Output</td>
<td>An array containing the co-ordinates of a rectangle bounding the face in the image</td>
</tr>
<tr>
<td>Description</td>
<td>This method uses the face detection functionality provided in the OpenCV library to detect a face in the given image.</td>
</tr>
</tbody>
</table>

Table 7.1: The face detection function

This function was used as part of the main face extraction and preprocessing function, `fc_resize`. Given an image containing a single face and the desired size, `fc_resize` will detect that face, extract it from the image using the information returned from `MATLABfacedetect` and will finally resize the face proportionally so that either the width or height of the image is equal to the given number.
Function | \texttt{fc\_resize}  
--- | ---  
Input | An image containing a face and the desired size  
Output | The extracted and resized image of the face  
Description | This method uses the MATLAB\texttt{facedetect} function to locate a face in the image. The detected face is then extracted and resized to the chosen dimensions.

Table 7.2: The image preprocessing function

7.2.2 Eigenfaces

The implementation of this method followed closely the logic as described in chapters 4-6. Three different functions are used to provide the functionality for creating the database and the Eigenmodel as well as searching for a face. The signatures of these methods can be seen in tables 7.3, 7.4 and 7.5.

Function | \texttt{eig\_createModel}  
--- | ---  
Input | The number of images, the number of eigenvectors required, the size of the images, the path where images are located and finally, the file extension.  
Output | The eigenvalues and eigenvectors describing the given images  
Description | Using this function the user can create an eigenmodel from the selected set of images.

Table 7.3: Function for creating the model for Eigenfaces

Function | \texttt{eig\_createDatabase}  
--- | ---  
Input | The number of images to be enrolled, the path for the images and the file extension  
Output | A Matlab structure containing the database of known individuals  
Description | Using this function, the given images are projected to the eigenspace and saved together with the filename in order to form the database of known individuals.

Table 7.4: Function for enrolling images for the Eigenfaces method

Although the process for creating an Eigenmodel is straightforward, it is easy to see that several computations such as the calculation of the Covariance matrix are very complex. Singular Value Decomposition was used instead as explained in section 6.1, only computing a small subset of the eigenvectors.

After creating the Eigenmodel, the data is stored in the \texttt{eig.mat} file. The \texttt{eig} function uses this model to project each image of the training set to the eigenspace. This is a simple process:

The projected version of each image is then saved in \texttt{eig.mat} and is associated to the
Function | eig_match  
--- | ---  
Input | An image of an individual to be recognised  
Output | A Matlab structure containing the 10 best matches for the given image  
Description | The given image is projected to the eigenspace computed through training. The projected image is then compared with the elements of the database in order to obtain the best matches.

Table 7.5: Eigenfaces recognition function

```matlab
meanX = mean(X,2);
dX = X - repmat(meanX, 1, noOfImages);
[U,S,V] = svd(dX);
eigVectors = U;
eigValues = diag(S^2);
```

Finally, the searching function simply projects the new image in a similar way and computes the Euclidean distance between this simpler representation of the input image and each element of the database. The results are sorted using Matlab’s built-in sorting functionality and the ten top matches are returned in descending order.

### 7.2.3 Template Matching

Two functions where used in this case in order to provide the functionality for this method. The `temp_insertToDatabase` method was used for the enrolment stage, while `temp_search` was integrated in the interface to find individuals using the template matching method. No training was required for this method.

The `temp_insertToDatabase` function simply uses the `fc_resize` function to extract the face from the image and resize it to 100 × 100 pixels. The resized face is then stored together with the filename in the `temp_db.mat` file which forms the database.

The searching function finds the best match using normalised cross correlation in order to compare the input image with each element of the database. The input image is resized and the contrast is adjusted to improve recognition. It should be noted that although a Matlab function exists to perform the correlation, namely `normxcorr`, the equivalent OpenCV function was used as it greatly improved the performance.

Following the system requirements in order for the method to return output of the same form as the other face recognition algorithms, the 10 best matches were obtained. In order to achieve that, after finding the best match in the database, that image was
image = resize(originalImg, size);
image = reshape(image', totalNoOfPixels, 1);
projection = eigenvectors' * image;

<table>
<thead>
<tr>
<th>Function</th>
<th>temp_insertToDatabase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>The image to be inserted in the database.</td>
</tr>
<tr>
<td>Output</td>
<td>N/A</td>
</tr>
<tr>
<td>Description</td>
<td>This function will extract and resize the face to the appropriate dimensions and insert it to the database.</td>
</tr>
</tbody>
</table>

Table 7.6: Function to insert an image to the template database

removed. The search was repeated with the remaining images until all the required matches were found.

<table>
<thead>
<tr>
<th>Function</th>
<th>temp_search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>The image to be recognised</td>
</tr>
<tr>
<td>Output</td>
<td>A Matlab structure containing the 10 best matches</td>
</tr>
<tr>
<td>Description</td>
<td>Using normalised cross correlation between the input image and each individual in the database, this method finds the best matches.</td>
</tr>
</tbody>
</table>

Table 7.7: The template matching function

7.2.4 Geometric Features

The functionality of this method was again divided between enrolment and searching stages as follows from chapters 4-6. In order to insert images to the database, the `gm_createDatabase` function is used. Feature points (fig. 7.4 displayed on the image can be manually moved by the user in order to best fit the face. The initial locations of the feature points are based on a manually created model.

In order to improve the usability of this part of the system, an interactive method is used to move deature points to the appropriate locations. More specifically, the `interactive_move` [44] method is used where a point or graph plotted on the selected axes can be selected and moved. The new point locations can be then retrieved through the axes as follows:

The distances between features are then calculated and stored in a vector. Each image vector together with the corresponding filename is saved in the `gm_data.mat` file.

Searching functionality is provided through the `gm_search` function and it follows the guidelines described in section 3.5.4. The feature points are located manually on the input image following the process described earlier and a vector is created containing
distances between the features. The best matches to the given image are then found by computing the Euclidean distance between the input image vector and each element in the database.

The features selected to describe the face are the following:

- The center of the eyes
- The center of the nose
- The sides of the nose allowing the width of the nose to be measured
- The corners of the mouth
- The center of the mouth

16 different distances are used to form the vector corresponding to each face as can be seen in table 7.8.
Table 7.8: Distances used to classify different faces. Refer to fig.7.4 for details.

<table>
<thead>
<tr>
<th>Start point</th>
<th>End point</th>
<th>Start point</th>
<th>End point</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>8</td>
<td>10</td>
</tr>
</tbody>
</table>

7.2.5 The User Interface

After developing the three face recognition methods, the user interface was created using Matlab’s GUIDE environment. GUIDE provides a visual editor which greatly simplifies the interface development process.

The interface is divided into two main parts (fig. 7.5). The input image selected by the user is displayed on the left panel while the recognition result from each method are displayed on the right.

Functionality attached to the interface is triggered through a number of events related to elements such as buttons and context menus. After loading an image, the feature points used for the Geometric-based method are plotted on the face allowing the user to move them to the correct locations.

Figure 7.5: The user interface
When searching for an individual, the search function corresponding to each recognition method is called. As all methods are based on common guidelines, the matches are returned in a specific form. More precisely, each method returns a structure with a field named \texttt{file} where the full path of that specific result is contained. Each method can use other fields if required to pass additional information to the interface. Results returned from each method are stored in global variables named \texttt{EIG\_MATCHES}, \texttt{TEMP\_MATCHES} and \texttt{GM\_MATCHES}.

The first four results are displayed for each method. However, functionality for scrolling through all 10 results is provided. This is achieved simply by obtaining the current position of the slider and displaying the appropriate result.

A complete list of the functions available through the interface can be found in table 7.2.5.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open image of individual to be recognised</td>
<td>Select the first option in context menu of left panel</td>
</tr>
<tr>
<td>Move feature points</td>
<td>Click and drag a point on the image</td>
</tr>
<tr>
<td>Re-anable interactive movement for feature points</td>
<td>Select the second option in context menu of left panel</td>
</tr>
<tr>
<td>Search for matches</td>
<td>Click the 'Search' button</td>
</tr>
<tr>
<td>View details about a specific match</td>
<td>Select the first option in context menu of the desired match</td>
</tr>
<tr>
<td>View more matches for one method</td>
<td>Move the scroll bar under the results of that method</td>
</tr>
<tr>
<td>Clear all results and the input image</td>
<td>Click the 'Clear' button</td>
</tr>
<tr>
<td>Close the application</td>
<td>Click the 'Close' button</td>
</tr>
</tbody>
</table>

Table 7.9: The functionality provided through the interface
Chapter 8

Evaluation

An important part of this project is the evaluation and comparison of different face recognition methods. A variety of algorithms has been suggested over the past 3 decades as potential solutions to the face recognition problem. However, automated machine recognition of human faces is still largely considered an open problem.

Although most algorithms are tested as part of their development, testing typically focuses on a small subset of the possible situations that could occur in a real world application. Consequently, detailed evaluation of methods under a variety of conditions is desirable if not necessary to enable them to be useful in practical situations.

Moreover, face recognition is an area where the suggested approaches use widely different techniques and tools to solve the same problem. Several interesting questions arise through that observation. By expanding our understanding about the aspects of this problem that allow it to be solved by so different approaches, we could gain some insight about face and object recognition in general.

8.1 Testing Plan

The evaluation of the three different methods will focus on their recognition performance under various different conditions. A series of tests will be performed to evaluate the methods under 3 different modes of variation:

- illumination
- pose
- expression

The same structure will be followed for all the tests. More specifically, six sets of tests will be used. In each case, training sets of 15 different individuals will be used with one
or two images per individual. These sets will be referred to as the target sets while the input image to be recognised will be the query image. A detailed account of the contents of each training set is given in table 8.1.

<table>
<thead>
<tr>
<th>Training Set</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Frontally oriented and illuminated faces will be used in this training set. The aim of this test is to investigate the performance of the three algorithms under ideal conditions.</td>
</tr>
<tr>
<td>b</td>
<td>Images taken under varying illumination are used. Faces are frontally oriented in this case too. This test class will be used to evaluate how recognition performance is affected when lighting conditions are extreme.</td>
</tr>
<tr>
<td>c</td>
<td>The third test set will investigate the effect of varying expressions on the recognition process. Illumination and pose in this set are kept constant.</td>
</tr>
<tr>
<td>d</td>
<td>This set is similar to the previous. However, in this case two images are used per individual, one is taken using various facial expressions while in the second image expressions are neutral. The purpose of this set is to investigate how performance changes if more than one image is available for each individual.</td>
</tr>
<tr>
<td>e</td>
<td>This training set will evaluate how pose variations affect the performance of the three recognition methods. Illumination is kept constant to the extent that is possible. Out-of-plane rotation in 4 directions are used, with rotation angles of up to 45 degrees allowed.</td>
</tr>
<tr>
<td>f</td>
<td>This class of tests also focuses on pose variations. However, two images are used for each individual in this case: a frontally oriented and a rotated one.</td>
</tr>
</tbody>
</table>

Table 8.1: The training sets used

For each test class, the recognition performance will be measured using two metrics. Through the user interface all three methods will be tested simultaneously. The performance metrics will be the following:

1. Correct recognition is achieved if the correct individual is returned as the best match for each method. In cases where multiple images per individual are used it will be sufficient for any one of the images to appear as the best match.

2. If the correct result appears among the 3 top results, partial recognition is achieved. Using this metric, some insight can be gained concerning cases where some additional preprocessing on the images could be sufficient for improving the results.

For each test class 8 query images will be used to evaluate the performance of each method. Two images will be taken directly from the training set and will be used simply
to verify that the three methods function as expected. The rest of the images will depict known individuals. However, different images will be used to the ones contained in the training set.

It should be noted that although the number of images in each training set is quite small, the only method that could be affected is Eigenfaces. Nevertheless, it was decided to use that minimal amount of information in order to evaluate the method in the stricter conditions possible.

### 8.2 Hypotheses

As discussed in chapter 5, in any object recognition and classification problem, external variations not due to differences between the actual objects, will affect the performance of algorithms. However, different algorithms use different tools to achieve the same goal and as a result will be affected accordingly by variations in the images.

The face recognition methods chosen for this project use significantly different sets of tools in order to identify a person in an image. As a result, it was expected that the performance of each method would be affected by different variations.

According to the Eigenfaces approach as explained in section 6.1, an image is represented as a point in a high-dimensional space. The coordinates for each dimension correspond to the value of the respective pixel in the image. As a result, by changing the intensity value of a pixel, the image point will move along the corresponding direction.

Under extreme illumination, a large number of pixels will appear much darker (shadows) or lighter (highlights) to what they would normally look like. With great enough variations, the point corresponding to that image may be further apart from another
point of the same individual compared to images of a different person but under similar lighting conditions.

On the other hand, variations due to facial expression affect a small number of pixels. It is thus expected that Eigenfaces and the Template matching approaches will not be greatly affected by such variations. The Geometric-based method however, uses feature related information to correctly identify a face and as a consequence, varying expressions may affect its performance.

Finally, as no pose normalisation is applied to the images before the recognition stage, it is expected that all three methods will be affected by rotated faces in the images. By increasing the rotation angle, the face structure should appear increasingly different to the recognition methods, making difficult to correctly identify the individual.

8.3 Test Results

For each test class, all three methods were trained using the images in the respective training set. After completing the training and enrolment process, the face recognition application was run with each of the images in the corresponding query set. After obtaining the results, the performance of each method was recorder in terms of whether the correct individual appeared as the first result or in the top three results.

Query images are named according to the following convention: \( N_x y \)
where \( N \) is the name of the test class, \( x \) is the category of query images (1 for images contained in the database, 2 for different images) and corresponds to the number of the image for that specific category.

8.3.1 Class a - Normal Conditions

Frontally facing images were used for this set. Illumination and facial expressions were neutral. The query images used were also frontally oriented however lighting conditions as well as expressions varied.

**Query Set**
The eigenfaces method achieved partial recognition in every test performed. However, the best performing method in this case was the template matching method.

8.3.2 Class b - Illumination Variations

This test class focused on illumination variations. The images used to train the three methods in this case were taken under quite extreme lighting conditions. The pose of the individuals however remained unchanged.
The results for this method indicate that both the Eigenfaces and the template matching approach were affected by the change in illumination.

### 8.3.3 Class c - Expression Variations

This class of tests investigated the performance of the different methods when the individuals had varying facial expressions. Illumination and pose however were kept constant.

#### Query Set

![Query Set Images](image)

#### Training Set

![Training Set Images](image)
## Results

<table>
<thead>
<tr>
<th>Image</th>
<th>Eigenfaces</th>
<th>Geometric</th>
<th>Templates</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1,1</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>c1,2</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>c2,1</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>c2,2</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>c2,3</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>c2,4</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>c2,5</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>c2,6</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Total: $5/8$ $7/8$ $4/8$ $5/8$ $7/8$ $7/8$

The best performing method in this case was the template-based approach, not being able to recognise only one individual. The geometric approach on the other hand was more affected by the varying facial expressions.

### 8.3.4 Class d - Expression Variations

Facial expressions were again the focus for this test. However in this case, two images were used per individual with different facial expressions. A result was considered a match if any of the two images of the individual was recognised.

#### Query Set

![Query Set Images](image1)

#### Training Set

![Training Set Images](image2)

73
Results

<table>
<thead>
<tr>
<th>Image</th>
<th>Eigenfaces</th>
<th>Geometric</th>
<th>Templates</th>
</tr>
</thead>
<tbody>
<tr>
<td>d1,1</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>d1,2</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>d2,1</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>d2,2</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>d2,3</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>d2,4</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>d2,5</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>d2,6</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

**Total**

6/8  7/8  4/8  8/8  7/8  7/8

Although the performance of the Eigenfaces and the template-based method were not greatly improved by the presence of the second image, the geometric-based approach was able to at least partially recognise all the individuals compared to the previous test case.

### 8.3.5 Class e - Pose Variations

This class of tests focused on pose variations and their effect on the recognition performance of the three methods. Out-of-plane rotations of up to 45 degrees were considered for this test case.

**Query Set**

![Query Set Images](image1)

**Training Set**

![Training Set Images](image2)
Results

<table>
<thead>
<tr>
<th>Image</th>
<th>Eigenfaces</th>
<th>Geometric</th>
<th>Templates</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1_1</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>e1_2</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>e2_1</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>e2_2</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>e2_3</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>e2_4</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>e2_5</td>
<td></td>
<td></td>
<td>The face was not detected</td>
</tr>
<tr>
<td>e2_6</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

Total: 3/8 5/8 2/8 2/8 3/8 3/8

It is easy to see that the performance of all the recognition methods was greatly affected by the variations in pose. Two methods managed to successfully identify a single individual when the query image was not identical to the images in the training set while the Geometric-based method only matched correctly images taken from the database.

8.3.6 Class f - Pose Variations

This class of tests was performed using two images per individual. One of the images was taken under a varying pose while the second image was frontally oriented. The aim of this test was to investigate how performance changes compared to the previous test class when multiple images are used for each person.

Query Set

![Query Set Images]
Training Set

<table>
<thead>
<tr>
<th>Image</th>
<th>Eigenfaces</th>
<th>Geometric</th>
<th>Templates</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1_1</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>f1_2</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>f2_1</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>f2_2</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>f2_3</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>f2_4</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>f2_5</td>
<td></td>
<td>The face was not detected</td>
<td></td>
</tr>
<tr>
<td>f2_6</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Total: 3/8 6/8 4/8 5/8 4/8 4/8

As it can be seen, although recognition performances are still not satisfactory, compared to the results of the previous test case, they have improved.

Detailed results for several of the tests performed can be found in Appendix I.

8.4 Results Discussion

The results obtained in the previous section will now be discussed in detail. The overall performance of each face recognition method will be evaluated. Moreover, the results will be analysed comparatively, in order to identify the stronger aspects of each method or how different techniques can be improved in order to build a robust recognition system.
Figure 8.2: As it can be seen, when the query image is identical to an element of the database, recognition is easy.

### 8.4.1 General Performance

Looking at the overall performance of each method, the following results can be computed:

<table>
<thead>
<tr>
<th>Method</th>
<th>Best Match</th>
<th>Top 3 Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenfaces</td>
<td>56.5%</td>
<td>84.8%</td>
</tr>
<tr>
<td>Geometric-based</td>
<td>52.2%</td>
<td>76.1%</td>
</tr>
<tr>
<td>Template Matching</td>
<td>67.4%</td>
<td>71.7%</td>
</tr>
</tbody>
</table>

A number of interesting observations arise from these results. It can easily be seen that in terms of the best match returned for each test, the Template matching approach is the best performing method. Second comes the Eigenfaces approach while the Geometric-based method only recognised approximately half of the individual in the given query images.

Basic recognition was achieved by all the recognition methods in every test case. For these tests, query images were taken directly from the database, thus testing the methods ability to recognise identical images. The purpose for these tests was to verify that the face recognition methods were correctly implemented and could successfully identify individuals under ideal conditions (fig. 8.2).

Concentrating on the performance of the methods as a measure of the three top matches, it can be seen that recognition rates greatly improve. The Eigenfaces method achieved the highest recognition rates with the other two methods returning the correct individual within the top 3 results more than 70% of the time.
It is easy to see that such recognition rates would not be sufficient for a robust face recognition system. However, they give a good indication of how variations in the image affect the recognition performance of the methods. Although the three methods are capable of successfully identifying individuals under controlled conditions, it is clear that images would need to be preprocessed to improve recognition rates under varying conditions.

Looking at the recognition performance of the three methods comparatively, an interesting observation arises. The template matching approach gives the best results, however according to the second metric used, the Eigenfaces method is the best performing one.

A number of reasons led to these results. Using the Template matching approach, images are directly compared to each other in order to compute how similar they are. As a result, in tests where large parts of the template are similar to the image of the same individual from the database, the correct result will be returned.

The Eigenfaces approach on the other hand uses statistical information to identify a face. As only 15 images were used for each training set, only a minimal set of eigenvectors could be computed covering only the most dominant aspects of the faces. This decision was made in order to test the method under the stricter possible conditions. Consequently, when a new face is projected to the eigenspace, some useful information may be lost, affecting the recognition performance.

8.4.2 Performance Under Variations

The tests performed focused on common variations that can occur in images. Typically, such variations can greatly affect the performance of face recognition methods, an effect visible through the results described in the previous sections.

Even using a minimal training set, all 3 methods were capable of recognising images taken directly from the training set. This indicates that when the appearance of a face is close to the images used during the training process of the method, recognition is more effective.

However, in order to achieve that in practical situations, images would have to be taken under highly controlled conditions, which is typically not feasible. This section will look at how different variations affect performance and will discuss the reasons behind this effect as well as potential solutions.

Illumination

Varying lighting conditions were tested through the second training set (Class b). The images of the 15 individuals were illuminated under different light sources. In most cases, light was directed from either the left or the right producing dark shadows and intense highlights. Faces however were frontally oriented in both the training and the query set used.
The best performing method for this class of tests was unsurprisingly the Geometric-based approach. As the information used by this method in order to identify someone is based on the geometric structure of the faces, intensity based information is irrelevant.

Figure 8.3: Even though both the pose and expression are almost identical, the illumination changes in these two images would greatly affect the recognition of the one using the other as the query image.

On the other hand, the other two methods are affected by illumination. The Eigenfaces approach creates a statistical model of what a face is expected to look like. As the training set used in this case contained images under extreme lighting conditions, the statistical description of that set covered these variations too. Moreover, varying illumination can affect a large number of pixels in the image, accounting for a large part of the total variations between different faces. Consequently, the eigenvectors corresponding to these variations will be more significant, causing images of the same individual not to be recognised if the lighting is different in some cases.

Similarly, as the Template matching approach compares every pixel in the two images, if a large section of the face is darker due to a shadow, it is easy to see that the values of the pixels under that area will be very different to the equivalent part of the normally illuminated image. As a result, variations that affect the image globally will make recognition more difficult, explaining the reduced performance of the method in this case.

**Expressions**

Different facial expressions affect the appearance of the face globally as well as locally. It was hence important that the effect of such variations was determined with respect to the performance of the three methods.

Two training sets were used to investigate the effect of facial expressions. The first set, as discussed in section 8.3.3, contained a single image for each individual with a variety
Figure 8.4: A sample of the results related to illumination. As it can be seen, the Geometric method successfully recognised the individual. The other two methods however are affected by illumination variations.

of expressions displayed across the whole set. The second set on the other hand used two different images for each individual. In the first image the individual appeared with a distinct facial expression such as laughing or sleepy, while for the second image a neutral expression was used.

The query sets for both test cases contained images with different expressions and lighting conditions, pose however was kept constant. Looking at the overall results for each of the three methods, it is observed that the Template matching approach is the best performing one in both test cases.

This can be explained by looking closely at the structure and appearance of the faces under different facial expressions. Figure 8.6 shows the same individual first with a neutral expression and then smiling. Combining the two images, it can be seen that although locally the appearance of the features changes, large areas such as the cheeks, forehead and nose remain largely the same. As a result, when comparing the two images, although a small number of pixels will vary due to the different expressions, the general location of the darker and lighter areas on the face will remain the same.

On the other hand, such variations on facial features have a much greater effect on the performance of the Geometric-based method. Distances between feature points for different faces will usually not vary significantly. Consequently, successful recognition is sensitive to small variations that exist between different individuals. Expressions affect the appearance of features and as a result, measurements of distances or sizes of certain features will be affected.

This is an important problem related mainly to feature-based techniques. Although
focusing on specific features rather than the face as a whole can help overcome issues related to illumination and occlusion of parts of the face, even small changes on these features can reduce the performance of the method. Attention is thus drawn to the selection of feature points to be used. Points such as the bottom of the nose, center of the eyes or face outline remain invariant in most situations. Points on the mouth or eyebrows on the other hand can easily change and hence are not appropriate.

An interesting observation is made based on the results of the second class of tests related to expression variations. The performance of both the Eigenfaces and the Template matching approaches remained virtually unchanged even though two images were now used for each individual. The Geometric-based approach however managed to return the correct individual within the top 3 results every time As the second image used had a neutral expression, query images with less extreme expressions were now closer to being recognised.

**Pose**
The two last test classes focused on variations due to different poses. As with expression variations, two different cases were tested. Firstly, only a single image was used for each individual where faces were rotated in any out-of-plane direction (up, down, right or left). The second training set contained the same images as well as an additional frontally facing image for each person. The query sets in both situations included images of the individuals under different poses and in some cases varying lighting conditions.

As it can be seen from the results for these test classes (sections 8.3.5 and 8.3.6), pose variations greatly affected the performance of all three methods. In the case where only a single image was used, all methods managed to return the correct result for only a single query image not taken from the training set. This was only slightly improved when two images were used and pose variations even in the query set seemed to have an adverse effect in recognition performance. It should also be noted that the face detection algorithm was also affected by the variations in the pose.

The reason for this is simple and gives a good indication on the importance of pose normalisation for a robust face recognition algorithm. An object, such as for instance a car, when looked at from different angles, appears very different(fig. 8.8). The same is true for faces, although more difficult to see due to our familiarity with them. Even small rotation angles cause the overall structure of a face to change. Increasing the angle of rotation, features are lost and distances between certain points on the face are distorted. Such changes may not be obvious from a first look, however when any of the algorithms used compares a rotated and a frontal image of the same person, recognition may fail.
In the case of the Eigenfaces approach, the statistical model created to describe the faces is created using rotated images. As a result, similar variations are expected in the query set in order to recognise the individuals. When that occurs, recognition is usually successful. However, in cases where the pose of the individual is different in the query and the target image, the individual is not correctly recognised. Moreover, as already mentioned, due to the rotation, distances and sizes of features appear distorted. This clearly affects the performance of the Geometric-based method as it is sensitive to such variations (fig. 8.9).

Finally, it is easy to see that the Template matching approach is also affected by variations in the pose. Unlike variations due to expressions, the locations of the features is now changed. Hence, when comparing the two template with an image of the same individual, even if some sections still match, other parts of the face will not align.
Figure 8.8: As it can be seen in the results for this test, none of the methods returned the correct result. However, as it is especially clear through the Eigenfaces results, individuals with similar poses were returned.

Figure 8.9: Distances between the features and the outline of the face are distorted when the face is rotated, making feature-based recognition very difficult.
Chapter 9

Conclusion

9.1 Critical Analysis

The aim of this project was to implement a set of face recognition methods in order to evaluate their performance under a variety of conditions that are likely to occur in real-world situations. Three methods were developed, all based in very different ideas and techniques, giving an indication of the variety of solutions available for the face recognition problem.

The methods implemented were the Eigenfaces algorithm, first suggested by Turk and Pentland [51], a Template matching approach, originally used for face recognition by Baron [11] and finally a Geometric-based approach suggested by Kanade in the early 70s [29]. Through extensive testing of all the methods, a number of interesting observations were made.

The performance of the three recognition methods proved not to be sufficient for robust recognition of different individuals. However, limited normalisation was performed on the images, showing the extent to which recognition is affected by certain variations.

Through the evaluation stage of the project, it was shown that the worst performing algorithm was the Geometric-based approach. Even small variations in the structure of the face seemed to affect its performance and it was observed that it was especially sensitive to expression variations, an aspect typically very hard to control in practical situations.

Generally, all the methods appeared to be highly sensitive to variations in the pose of the individuals, showing the importance of pose normalisation. Through this observation it can be easily understood that controlling the pose of the individuals pictured or ideally adjusting the images to account for such variations would be an essential step for building an effective solution.

Illumination proved to be the second most important cause of variations in images, affecting mainly the Eigenfaces and Template-matching methods. Although a major
problem in recognition and classification tasks, global variations in the lighting of images can be adjusted through several different normalisation techniques as discussed in section 5.1.

Another aspect that became apparent through the evaluation process was the time performance of the different methods. Although strictly not in the focus of this project, it was clear after performing a number of tests that three methods achieved different recognition speeds. Template matching, although it produced satisfactory recognition rates, proved to be a slow method, taking noticeably longer than the other two methods to recognise an individual. This should not come as a surprise as recognition is achieved through direct comparison of each image in the training set with the query image. The testing performance involved small training sets, however in practical situations the size of the database would be considerably larger, making this method impractical if fast recognition is required.

The other two methods on the other hand were able to produce results much faster, however as discussed, their sensitivity to certain variations in images makes them unreliable if such conditions cannot be controlled or adjusted. Concluding, although normalisation can be performed to adjust the size, pose and lighting conditions on a face, facial expressions are harder to control or remove.

Finally, looking at the application that was produced as part of this project, an effective and usable interface was developed. Through the application, the evaluation of the different methods was greatly simplified. Moreover, as the guidelines described in section 3.5.4 were followed throughout the project, the final system proved to be easily extensible and hence possible to use in future research. Different components can be simply integrated with the existing functionality, allowing different aspects of the methods to be reviewed.

### 9.2 Future Work

This section will focus on a number of possible extensions to the system as well as potential research directions to achieve robust and automatic recognition of human faces.

Based on the research undertaken for this project as well as previous research on the subject, the Eigenfaces approach can generally be considered a very promising method for object and more specifically face recognition. It is a global approach, which focuses on the face as a whole and tries to describe its variations statistically. No feature detection is required and effective recognition can be achieved even at relatively low resolutions. Moreover, the time performance of this method gives it a clear advantage.

Combining this method with a real-time face detection system such as the one used throughout this project as well as with effective image preprocessing and normalisation, an efficient solution capable of achieving high recognition rates could be produced. Moreover, due to the nature of the method, such a system would be easily scalable. After creating an Eigenmodel using a sufficient number of images, new individuals
could simply be added to the database by determining the projection of the new image to the eigenspace describing all the faces.

However, despite its limitations, the Geometric-based approach could also be used as part of an effective recognition system. As discussed, the method’s sensitivity to expressions and even small structural variations could make it unsuitable for certain situations. Nevertheless, with a careful selection of features, this effect could be minimised. Furthermore, as it was shown by Brunelli and Poggio [11], recognition through such a method is possible even at very low resolutions, hence reducing storage requirements and making this method useful in situations where higher quality images would not be available. Through automatic feature detection, the method would be able to function without requiring user input. In addition, as sensitivity to lighting conditions is limited, the size of the problem would be greatly reduced. Finally, searching for the best match for a query image is a simple process through this method. Using an efficient data structure, such as a multi-dimensional tree to store the contents of the database, fast recognition could be achieved.

Concluding, a variety of face recognition methods is available and research on the subject has been active for more than 3 decades. Face recognition however is still an open problem due to the large number of parameters that can affect the appearance of a face. The methods implemented and evaluated for this project are only a small subset of the variety of methods available. Despite the fact that each method performs well in certain situations, they are not capable of providing a complete solution. It is thus essential that the possible options are thoroughly reviewed depending on the application. Although the goal is common for all the available methods, they each have certain strengths and limitations that need to be considered. As such, a single, complete solution for the face recognition problem does not exist. A wide variety of tools is available however that when used correctly can greatly improve performance, making automatic recognition of human faces possible.
Bibliography


[27] Identix. Identix faceit frequently asked technical questions, 2005.


Appendix I - Test Results

Following are some screen captures of the application displaying some of the test results.

Class a - Normal Conditions

Figure 1: Class a - Normal Conditions: Query image 1_2

Figure 2: Class a - Normal Conditions: Query image 2_1
Figure 3: Class a - Normal Conditions: Query image 2.5
Class b - Illumination Variations

Figure 4: Class b - Illumination Variations: Query image 1_1

Figure 5: Class b - Illumination Variations: Query image 2_2
Figure 6: Class b - Illumination Variations: Query image 2_4

Figure 7: Class b - Illumination Variations: Query image 2_6
Class c,d - Expression Variations

Figure 8: Class c - Expression Variations I: Query image 1_1

Figure 9: Class c - Expression Variations I: Query image 2_1
Figure 10: Class c - Expression Variations I: Query image 2,3

Figure 11: Class d - Expression Variations II: Query image 1,1
Figure 12: Class d - Expression Variations II: Query image 2_1

Figure 13: Class d - Expression Variations II: Query image 2_4
Figure 14: Class d - Expression Variations II: Query image 2_6
Class e,f - Pose Variations

Figure 15: Class e - Pose Variations I: Query image 1.2

Figure 16: Class e - Pose Variations I: Query image 2.3
Figure 17: Class e - Pose Variations I: Query image 2.6

Figure 18: Class f - Pose Variations II: Query image 1.1
Figure 19: Class f - Pose Variations II: Query image 2.1

Figure 20: Class f - Pose Variations II: Query image 2.3
Appendix II - Project Plan

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Start</th>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Final Year Project</td>
<td>6/19/06</td>
<td>9/19/06</td>
<td>12/19/06</td>
</tr>
<tr>
<td>2</td>
<td>Project Proposal</td>
<td>3/19/06</td>
<td>6/19/06</td>
<td>9/19/06</td>
</tr>
<tr>
<td>3</td>
<td>Literature</td>
<td>6/19/06</td>
<td>9/19/06</td>
<td>12/19/06</td>
</tr>
<tr>
<td>4</td>
<td>First Draft</td>
<td>9/19/06</td>
<td>12/19/06</td>
<td>1/19/07</td>
</tr>
<tr>
<td>5</td>
<td>Literature</td>
<td>12/19/06</td>
<td>2/19/07</td>
<td>4/19/07</td>
</tr>
<tr>
<td>6</td>
<td>Requirements Specification</td>
<td>2/19/07</td>
<td>4/19/07</td>
<td>6/19/07</td>
</tr>
<tr>
<td>7</td>
<td>Report</td>
<td>4/19/07</td>
<td>6/19/07</td>
<td>8/19/07</td>
</tr>
<tr>
<td>8</td>
<td>Design</td>
<td>6/19/07</td>
<td>8/19/07</td>
<td>10/19/07</td>
</tr>
<tr>
<td>9</td>
<td>Code</td>
<td>8/19/07</td>
<td>10/19/07</td>
<td>12/19/07</td>
</tr>
<tr>
<td>10</td>
<td>Implementation</td>
<td>10/19/07</td>
<td>12/19/07</td>
<td>2/19/08</td>
</tr>
<tr>
<td>11</td>
<td>Testing</td>
<td>12/19/07</td>
<td>2/19/08</td>
<td>4/19/08</td>
</tr>
<tr>
<td>12</td>
<td>User Interface</td>
<td>2/19/08</td>
<td>4/19/08</td>
<td>6/19/08</td>
</tr>
<tr>
<td>13</td>
<td>Final draft</td>
<td>4/19/08</td>
<td>6/19/08</td>
<td>8/19/08</td>
</tr>
<tr>
<td>14</td>
<td>Post-presentation</td>
<td>6/19/08</td>
<td>8/19/08</td>
<td>10/19/08</td>
</tr>
<tr>
<td>15</td>
<td>Final presentation</td>
<td>8/19/08</td>
<td>10/19/08</td>
<td>12/19/08</td>
</tr>
<tr>
<td>16</td>
<td>Project hand-in</td>
<td>10/19/08</td>
<td>12/19/08</td>
<td>2/19/09</td>
</tr>
</tbody>
</table>
Appendix III - Code Listing

Eigenfaces
eig_createModel

1 function [eVectors, eValues] = eig_createModel(noOfImages, noOfVectors, imgSize, path, file )
2 % [eVectors, eValues] = eig_getImages( noOfImages, noOfVectors, size, path, file )
3 % Calculates the eigenvectors and their respective eigenvalues for the
4 % selected images. It will only return a number of
eigenvectors specified
5 % by noOfVectors. All the images need to be of the same resolution..
6
7 if nargin == 3
8 for i = 1:noOfImages
9 [file,path] =
uigetfile({'*.jpg;*.bmp;*.gif','Image Files (*.jpg, *.bmp, *.gif); ... 
10 '*.*','All files'},
'Pick a file');
11
12 if nargin == 4
13 f = ls(path);
14 % remove the . and ..
15 f = f(3:end,:);
16 for i = 1:size(f,1)
17 fullPath = [path,f(i,:)];
18 img = fc_resize(imread(fullPath),imgSize);
19 [h,w] = size(img);
20 files(i).name = fullPath;
21 M(1:w*h,i) = reshape(img',w*h,1);
22 end
23 elseif nargin == 5
24 for i = 1:noOfImages
25 fullPath = [path,int2str(i),file];
26 img = fc_resize(imread(fullPath),imgSize);
27 [h,w] = size(img);
28 files(i).name = fullPath;
29 M(1:w*h,i) = reshape(img',w*h,1);
30 end
31 elseif nargin == 5
32 for i = 1:noOfImages
33 fullPath = [path,int2str(i),file];
34
104
33    img = fc_resize(imread(fullPath),imgSize);
34    [h,w] = size(img);
35    files(i).name = fullpath;
36    M(1:w*h,i) = reshape(img',w*h,1);
37   end
38  end
39  save('eig_data.mat','M','files');
40
41  % find the average face
42  meanM = mean(M,2);
43  % align at the origin
44  diffM = double(M) - repmat(meanM,1,size(M,2));
45  % svd of diffM
46  [U,S,V] = svd(diffM,0);
47  % then the the covariance of diffM would be
48  1/noOfImages*diffM*diffM'
49  % if we replace diffM by its svd decomposition: % C = (1/n)*(U*S*V)*(U*S*V)' = U*S^2*U' which is
50  % basically the % eigendecomposition for C
51  eValues = diag(S^2);
52  eVectors = U;
53
54  if noOfImages > noOfVectors
55    eValues = eValues(1:noOfVectors);
56    eVectors = eVectors(:,1:noOfVectors);
57  end
58
59  save('eig_data.mat','eVectors','eValues','meanM','imgSize','-append');

---

eig_createDatabase

1    function DB = eig_createDatabase(noOfImages, path, file)
2
3    load('eig_data.mat','eVectors','eValues','meanM','imgSize');
4    if nargin == 3
5        for i = 1:noOfImages
6            fullPath = [path, int2str(i), file];
7            img = fc_resize(imread(fullPath),imgSize);
8            img = reshape(img', size(img,1)*size(img,2),1);
9            img = img - meanM;
10           DB(i).file = fullPath;
11           DB(i).proj = eVectors'*img;
12        end
13    elseif nargin == 2
14        DB = eig_createDatabase(noOfImages, path, file);
15    end
16
17    return
files = ls(path);
% remove the . and ..
files = files(3:end,:);
for i = 1:size(files,1)
    fullPath = [path, files(i,:)];
    img = fc_resize(imread(fullfile),imgSize);
    img = reshape(img', size(img,1)*size(img,2),1);
    img = img - meanM;
    DB(i).file = fullPath;
    DB(i).proj = eVectors'*img;
end
save('eig_DB.mat','DB');

eig_match

function [ Matches ] = eig_match( img )
% Finds the best matches to the given image using
eigenvalue decomposition
% Returns a structure Matches that has a field
% Matches.file containing the
% full path to the images and another field Matches.dist
% containing the
% actual euclidean distances
if nargin == 0
    [file,path] = uigetfile({'*.jpg;*.bmp;*.gif','Image Files (*.jpg, *.bmp, *.gif)'; ... '*.*','All files'},
        'Pick a file');
    img = imread([path,file]);
end
% Read each image and convert it to a long vector
containing the values
load('eig_data.mat','eVectors','eValues','meanM','imgSize');
img = fc_resize(img,imgSize);
img = reshape(img', size(img,1)*size(img,2), 1);
img = img - meanM;
% project to eigenspace
proj = eVectors'*img;
load('eig_DB', 'DB');

for i = 1:length(DB)
    D(i) = sqrt(sum((DB(i).proj - proj).^2));
end

[Y, I] = sort(D);

for i = 1:10
    Matches(i).file = DB(I(i)).file;
    Matches(i).distance = D(i);
    Matches(i).proj = DB(I(i)).proj;
end
Geometric Features

gm_createDatabase

```matlab
1 function DB = gm_createDatabase( noOfImages, path, file )
2 % m = gm_createDatabase( noOfImages, path, file )
3
4 if nargin == 3
5     for i = 1:noOfImages
6         fullPath = [path, int2str(i), file];
7         img = imread(fullPath);
8         img = fc_resize(img,300);
9         imshow(img,[0 255]);
10        hold on;
11        load('gm_data.mat','Model');
12        for md=1:size(Model,1)
13            plot(Model(md,1),Model(md,2),'r*')
14        end
15        % move points to best match the face
16        interactive_move;
17        pause;
18
19        % get new positions of the points
20        Points = findobj(gca,'Type','line');
21        X = [];
22        Y = [];
23        for pt=1:length(Points)
24            X = [X;get(Points(pt),'XData')];
25            Y = [Y;get(Points(pt),'YData')];
26        end
27
28        DB(i).X = X;
29        DB(i).Y = Y;
30        DB(i).file = fullPath;
31    end
32
33    save('gm_data.mat','DB','-append');
34 elseif nargin == 2
35    files = ls(path);
36    % remove the . and .. files = files(3:end,:);
37    for i = 1:size(files,1)
38        fullPath = [path, files(i,:)];
39        img = imread(fullPath);
40        img = fc_resize(img,300);
41        imshow(img,[0 255]);
42        hold on;
43        load('gm_data.mat','Model');
44    end
45
```

108
for md=1:size(Model,1)
    plot(Model(md,1),Model(md,2),'r*')
end

% move points to best match the face
interactive_move;
pause;

% get new positions of the points
Points = findobj(gca,'Type','line');
X = [];
Y = [];
for pt=1:length(Points)
    X = [X;get(Points(pt),'XData')];
    Y = [Y;get(Points(pt),'YData')];
end

DB(i).X = X;
DB(i).Y = Y;
DB(i).file = fullPath;
end

save('gm_data.mat','DB','-append');

load('gm_data.mat');

% create vector of distances
for im = 1:length(DB)
    for i = 1:size(Distances,1)
        DB(im).D(i) = sqrt((DB(im).X(Distances(i,2))-DB(im).X(Distances(i,1)))^2 +
        (DB(im).Y(Distances(i,2))-DB(im).Y(Distances(i,1)))^2);
    end
end

save('gm_data.mat','DB','-append');

% get new positions of the points
Points = findobj(gca,'Type','line');
X = [];
Y = [];
for pt=1:length(Points)
    X = [X;get(Points(pt),'XData')];
    Y = [Y;get(Points(pt),'YData')];
end

DB(i).X = X;
DB(i).Y = Y;
DB(i).file = fullPath;
end

save('gm_data.mat','DB','-append');

% create vector of distances
for im = 1:length(DB)
    for i = 1:size(Distances,1)
        DB(im).D(i) = sqrt((DB(im).X(Distances(i,2))-DB(im).X(Distances(i,1)))^2 +
        (DB(im).Y(Distances(i,2))-DB(im).Y(Distances(i,1)))^2);
    end
end

save('gm_data.mat','DB','-append');

gm_search

function Matches = gm_search( file, model )
% Gets matches using the geometric based method
load('gm_data.mat','DB','Distances');
load('gm_data.mat','DB','Distances');

% calculate distance vector for model (u idiot)
for i = 1:size(Distances,1)
    dist(i) = sqrt((model(Distances(i,2),1)-model(Distances(i,1),1))^2 +
        (model(Distances(i,2),2)-model(Distances(i,1),2))^2);
end
((model(Distances(i,2),2)-model(Distances(i,1),2))\^2));

9 end

10
11 for im = 1:length(DB)
12    for i = 1:length(DB(im).D)
13        d(im) = sqrt(sum((DB(im).D - dist).^2));
14    end
15 end

16 [Y,I] = sort(d,'ascend');
17
18 for i = 1:10
19    Matches(i).file = DB(I(i)).file;
20    Matches(i).dist = d(I(i));
21 end
Template Matching

temp_insertToDatabase

1 function img = temp_insertToDatabase( file )
2 inImg = imread(file);
3 imgSm = fc_resize(inImg, 100);
4 imgSm = adjcontrast(imgSm, 8, 0.5);
5 imgSm = fc_retinex(imgSm, 10);
6
7 imgL = fc_resize(inImg, 300);
8 imgL = adjcontrast(imgL, 3, 0.5);
9 imgL = fc_retinex(imgL, 10);
10
11 FeaturePts = temp_locateFeatures(imgL);
12 if FeaturePts(1,2) - 40 < 1;
13     hs = 1;
14 else
15     hs = round(FeaturePts(1,2) - 40);
16 end
17 if FeaturePts(1,2) + 40 > size(imgL, 1);
18     he = size(imgL, 1);
19 else
20     he = round(FeaturePts(1,2) + 40);
21 end
22 if FeaturePts(1,1) - 40 < 1;
23     ws = 1;
24 else
25     ws = round(FeaturePts(1,1) - 40);
26 end
27 leftEye = imgL(hs:he, ws:we);
28 if size(leftEye) ~= [81,81]
29     leftEye(size(leftEye,1)+1:81,size(leftEye,2)+1:81) = 1;
30 end
31
32 if FeaturePts(2,2) - 40 < 1;
33     hs = 1;
34 else
35     hs = round(FeaturePts(2,2) - 40);
36 end
37 if FeaturePts(2,2) + 40 > size(imgL, 1);
38     he = size(imgL, 1);
39 else
40     he = round(FeaturePts(2,2) + 40);
41 end
42 if FeaturePts(2,1) - 40 < 1;
43     ws = 1;
44 else
45     ws = round(FeaturePts(2,1) - 40);
46 end
47 rightEye = imgL(hs:he, ws:we);
48 if size(rightEye) ~= [81,81]
49     rightEye(size(rightEye,1)+1:81,size(rightEye,2)+1:81) = 1;
50 end
51
31 if FeaturePts(3,2) - 60 < 1; hs = 1; else; hs = round(FeaturePts(3,2) - 60); end
32 if FeaturePts(3,2) + 60 > size(imgL,1); he = size(imgL,1); else; he = round(FeaturePts(3,2) + 60); end
33 if FeaturePts(3,1) - 45 < 1; ws = 1; else; ws = round(FeaturePts(3,1) - 45); end
34 if FeaturePts(3,1) + 45 > size(imgL,2); we = size(imgL,2); else; we = round(FeaturePts(3,1) + 45); end
35 nose = imgL(hs:he,ws:we);
36 if size(nose) ~= [121,91]
37 nose(size(nose,1)+1:121,size(nose,2)+1:91) = 1;
38 end
39
40 if FeaturePts(4,2) - 25 < 1; hs = 1; else; hs = round(FeaturePts(4,2) - 25); end
41 if FeaturePts(4,2) + 25 > size(imgL,1); he = size(imgL,1); else; he = round(FeaturePts(4,2) + 25); end
42 if FeaturePts(4,1) - 60 < 1; ws = 1; else; ws = round(FeaturePts(4,1) - 40); end
43 if FeaturePts(4,1) + 60 > size(imgL,2); we = size(imgL,2); else; we = round(FeaturePts(4,1) + 45); end
44 mouth = imgL(hs:he,ws:we);
45 if size(mouth) ~= [51,121]
46 mouth(size(mouth,1)+1:51,size(mouth,2)+1:121) = 1;
47 end
48
49 if exist('temp_db.mat')
50 load('temp_db.mat');
51 DB(length(DB)+1).frontal = imgSm;
52 DB(length(DB)).leftEye = leftEye;
53 DB(length(DB)).rightEye = rightEye;
54 DB(length(DB)).nose = nose;
55 DB(length(DB)).mouth = mouth;
56 DB(length(DB)).file = file;
57 save('temp_db.mat','DB','-append');
58 else
59 DB(1).frontal = imgSm;
60 DB(1).leftEye = leftEye;
61 DB(1).rightEye = rightEye;
62 DB(1).nose = nose;
63 DB(1).mouth = mouth;
64 DB(1).file = file;
65 save('temp_db.mat','DB');
66 end
67
function Matches = temp_search( img )

if nargin == 0
    [file,path] = uigetfile({'*.jpg;*.bmp;*.gif','Image Files (*.jpg, *.bmp, *.gif); ... ','.*','All files'}, 'Pick a file');
    img = imread([path,file]);
end

img = fc_resize(img,100);
img = adjcontrast(img,8,0.5);
% img = fc_retinex(img,10);
load('temp_db.mat','DB');
dbImages = [];
for i=1:length(DB)
    dbImages = [dbImages, DB(i).frontal];
    f(i).file = DB(i).file;
end
i = 0;
while i < 10
    i = i+1;
    C = normxcorr2_mex(img,dbImages,'full');
    [val,ind] = max(abs(C(:)));
    [y,x] = ind2sub(size(C),ind(1));
    x = x - size(img,2);
    y = y - size(img,1);
    imgNo = round(x/100)+1;
    Matches(i).file = f(imgNo).file;
    Matches(i).corr = val;
end
% use the openCV version for normalised cross correlation, mex file
% kindly provided by http://www.cs.ubc.ca/~deaton/index.html because i
cant be asked doing it
C = normxcorr2_mex(img,dbImages,'full');
[val,ind] = max(abs(C(:)));
[y,x] = ind2sub(size(C),ind(1));
x = x - size(img,2);
y = y - size(img,1);
imgNo = round(x/100)+1;
% add the current best match in the results
Matches(i).file = f(imgNo).file;
Matches(i).corr = val;
% remove the current match from the 'database' and try to find the next
% best match
if imgNo > 1
dbImages = dbImages(:,[1:(imgNo-1)*100,(imgNo+1)*100:end]);
f = f([1:(imgNo-1),(imgNo+1):end]);
else
dbImages = dbImages(:,101:end);
f = f(2:end);
end
end