Reconstruction and Recognition of 3D Face Models
MSc Report

Name: **Oisín Mac Aodha**

Supervisor: Dr. Simon Prince

Year of Submission: 2007/08

Programme: MSc Intelligent Systems

Abstract

This project is concerned with the capture, prediction and verification of 3D models of human faces. An existing system is modified to capture high resolution 3D models. The system is based on an active stereo model which projects a structured light pattern onto the subject's face and then calculates its 3D structure. A dataset of 320 3D face models is collected from 64 different individuals.

A novel probabilistic approach is developed which predicts the 3D structure of a face from its 2D intensity image. The models are first aligned in 3D space and then the relationship between pixel intensity and depth is learned. A factor analysis model is described which accounts for missing data in the learning stage. The 3D structure prediction results from several different methods are compared and contrasted.

Up until recently the majority of face recognition research has focused on the use of 2D intensity images. It has been shown that 3D face recognition methods can achieve better performance than their 2D counterparts by utilising the 3D structure of the face. Three different experiments are conducted which show that 3D verification outperforms verification using pixel intensity information and verification using only depth information.

Acknowledgements

I would like to thank my supervisor Dr. Simon Prince for his help, enthusiasm and guidance throughout the project. I would also like to thank all the members of the Prince Vision Lab for being very approachable and helpful any time I came to them with questions.
Also I would like to commend my classmates for their friendship and ideas during the last year. Finally I would like to thank my parents and my sister for being so supportive and great proof readers. I would not have been able to complete my MSc without them and am very grateful for all their help.

Contents

1 Introduction **6**

2 Related Work **8**

2.1 3D Capture Methods . . . 8
2.1.1 Laser Range Scanner . . . 8
2.1.2 Stereo Reconstruction . . . 9
2.1.3 Photometric Stereo . . . 10
2.1.4 Shape From Silhouette and Space Carving . . . 10
2.2 3D Face Recognition . . . 11
2.2.1 Background . . . 11
2.2.2 Recognition by Profile Silhouette Extraction . . . 12
2.2.3 Shape Matching . . . 13
2.2.4 Feature Based Recognition . . . 14
2.2.5 Other Approaches . . . 16
2.3 3D Face Models . . . 17
2.3.1 Predefined Geometric Models . . . 17
2.3.2 Learned Models . . . 17
2.4 Summary . . . 18

3 Capturing Data **20**

3.1 Hardware and Calibration . . . 20
3.1.1 Hardware . . . 20
3.1.2 Camera Model . . . 20
3.1.3 Calibration . . . 21
3.2 Data Capture . . . 24
3.3 3D Models . . . 30
3.4 Reconstruction Problems . . . 31
3.5 Collected Dataset . . . 31
3.6 Summary . . . 32

4 Normalising the Dataset **33**
4.1 Aligning Models . . . 33
4.1.1 Storing Aligned Images . . . 35
4.2 Normalising Image Intensities . . . 36
4.3 Limitations and Discussion . . . 36
4.4 Final Dataset . . . 38
4.5 Summary . . . 38

5 Predicting Depth Using a Multivariate Gaussian Distribution **39**
5.1 Modelling Data . . . 39
5.1.1 Multvariate Gaussian Distribution . . . 39
5.1.2 Dimensionality Reduction . . . 40
5.1.3 Choosing the Number of Principal Components . . . 40
5.2 Predicting Depth . . . 41
5.2.1 Aligning 2D Image . . . 42
5.2.2 Conditional Distribution . . . 42
5.3 Results . . . 43
5.4 Summary . . . 44

6 Predicting Depth Using Factor Analysis **45**
6.1 Factor Analysis . . . 45
6.1.1 Expectation Maximisation . . . 46
6.1.2 FA Results . . . 46
6.2 Factor Analysis with Missing Data . . . 47
6.2.1 FA with Missing Data Results . . . 50
6.3 Summary . . . 52

7 3D Face Verification **53**
7.1 Latent Identity Variable Subspace Model . . . 53
7.1.1 Learning the Model . . . 54
7.1.2 Evaluating the Results . . . 54
7.2 Results . . . 55
7.2.1 2D Verification . . . 56
7.2.2 Depth Based Verification . . . 56
7.2.3 3D Verification . . . 57
7.3 Summary . . . 57

8 Discussion and Further Work **58**

Bibliography **60**

Appendices

A Accompanying DVD **63**
A.1 Code 63
A.1.1 C Code 63
A.1.2 MATLAB Code 64
A.2 Demos 64
A.3 Face Dataset 64

Chapter 1

Introduction

The task of creating a 3D model of an object can be a time consuming a difficult process. The equipment needed for capturing high resolution models can be expensive, large and not very portable. A cheap and quick alternative would provide a much easier way of collecting large amounts of data.

This work focuses on the reconstruction of 3D models of human faces. There are many applications for such models from the security to entertainment industries. From an entertainment point of view, it would allow users to create their own life like avatars for a more immersive experience in video games. Users would be able to change their characters appearance to exactly model their own without any extra expensive hardware.

Current 2D face recognition algorithms offer poor performance with images of individuals that contain large changes in pose, facial expression and lighting conditions. 3D face recognition may provide better results. The 3D structure of each individuals face has the potential to contain as much unique identity information as an image of the face. The major advantage of the face's 3D structure is that it is not effected by changes in lighting conditions when capturing data. 3D based recognition also has the advantage of being harder to trick. Some 2D face verification systems can be fooled by simply putting a photograph of the subject in front of the camera. In the 3D verification, the system will be able to tell that the photograph contains no depth and will not be fooled.

Aims and Problem Statement

The goal of this project is to create realistic looking 3D models of human faces. Going from a 2D image to its 3D geometric representation is an ill-posed problem. A 2D image contains no depth information so the geometry has to be some how inferred from the data. The project will attempt to investigate different approaches for predicting 3D face models, with the end result being 3D models which are life like and believable. A system needs to be created which captures 3D models of faces quickly and a large dataset of such models needs to be collected.

Another goal of the project is to investigate if 3D face verification has the ability to outperform standard 2D face recognition. Using 2D images with predicted depth values offers the possibility for greater recognition performance.

Approach and Contributions

An active stereo capture system is setup which reconstructs 3D models using projected structured light. This system allows the capture of high resolution models with near real time performance. A dataset of 320 models is collected. A technique based on the orthogonal Procrustes problem is used to align the models in 3D space.

Several different approaches to predicting depth are described. A novel version of the EM algorithm for factor analysis is introduced which learns the the model parameters while ignoring missing data points. This model creates more realistic predictions for the depth of an image. The model described, predicts depth much faster than of the other methods seen in the literature.

Face verification is performed using a latent identity subspace model, which allows multiple images of the same individual to be attributed to a single hidden variable in identity space. The results of experiments performed show that 3D verification has a better performance than 2D verification on the same data.

Overview

In chapter 2 we review the current work in the areas of data capture, face recognition and the creation of 3D models of faces. This provides a background for the work and gives a sense for what is the current state of the art in the field. Chapter 3 describes the method by which 3D models of human faces are captured. It outlines the set up of an existing active stereo capture system and describes the main contribution of this work to the system. It describes a collect dataset of 3D models which is the basis for the rest of the work in the report. Chapter 4 preprocesses this dataset so that it can be used for inference. This involves aligning the models in 3D space to a template model and normalising the intensity values of the pixels across the images. Chapters 5 and 6 describe three different methods for predicting the depth of a 2D face image. In each case a model is learned for the intensity of pixels and the depth at each of these locations from the collected dataset and then used to infer the depth of a new image. Chapter 7 performs three different face verification experiments of the dataset and compares the performance of 3D versus 2D verification. Finally, chapter 8 provides a discussion of the work and suggests avenues for further work.

All the code used in the report is included on the accompanying DVD, who's contents can be found listed in appendix A. All code is in MATLAB and C. The CD also contains all the face models which were captured in chapter 3. This included the original data and the preprocessed faces used in the inference stage.

Chapter 2

Related Work

The goal of this literature review is to provide a review of related work. This literature review is divided into three sections. Section 2.1 provides an overview of the different methods available for calculating the geometry of a 3D object. In recent years there has been a vast increase in the numbers of papers on the topic of 3D face recognition. Section 2.2 provides a brief introduction to 2D face recognition and then summarises the different 3D approaches to the problem. Taking a 2D image and recreating the 3D geometry of the object is an ill-posed problem. Section 2.3 discusses research into the area and specifically focuses on the creation of 3D models of human faces.

2.1 3D Capture Methods

The goal of 3D data capture is to analyse a real world object or environment and to collect data on its shape and possible appearance. The collected data represents a point cloud of geometric samples on the surface of the object, which can then be used to construct a 3D model of the object. 3D capture is used in security applications, object analysis and extensively in the entertainment industry (film and video games). Like traditional cameras, 3D capture devices have a cone like field of view and can only collect information about surfaces which are not obscured and in this field of view.

2.1.1 Laser Range Scanner

A laser range scanner is a device which uses a laser beam in order to determine the distance to a reflective object. The most common form of laser range scanner operates on the time of flight principle; by sending a laser pulse in a narrow beam towards the object and measuring the time taken by the pulse to be reflected off the target and returned to the sender. Using this technique, the 3D shape of the object can be calculated. The disadvantages of laser scanners is that they can be expensive to purchase. The subject must remain still for extended periods of time while they are being scanned; which increases the likelihood of errors due to movement. Due to the high speed of light, laser scanners are not appropriate for high precision measurements. For sub millimetre measurements triangulation and other techniques are often used.

The Face Recognition Grand Challenge version two 3D [14] dataset was acquired with a Minolta Vivid 910 laser scanner. The scanner uses triangulation with a laser stripe projector to build 3D models. Both texture and depth information are captured, but not simultaneously as it takes the laser scanner a few seconds to cross the face. The resolution of the camera is 640x480 which results in over 300,000 data points.

2.1.2 Stereo Reconstruction

Stereo reconstruction based methods estimate depth by matching points between two images of the same scene. The estimation of the depth is based on triangulation. The basic concept is that if we know the location of a particular feature in each of the two calibrated cameras (each camera is at a different position looking at the same scene), we can work out its depth.

A single camera, which can be described by a perspective projection, maps the point x in the 3D world coordinate systems into the point x_c in the 2D image plane coordinate system. All the points lying on the ray $o_c x_c$ are projected to x_c. In passive stereo, a second camera is used, adding another non collinear view $o'_c x'_c$ that can be used to resolve this ambiguity by triangulation. The location of x is uniquely determined by the intersection point of the two rays $o_c - x_c$ and $o'_c - x'_c$. This procedure is required to determine the corresponding x'_c for every x_c (referred to as the correspondence problem).

In active stereo, the second view is obtained by an active projector rather than a passive camera. A typical active stereo setup is shown in Figure 2.1. The projector is also described by a perspective projection, which

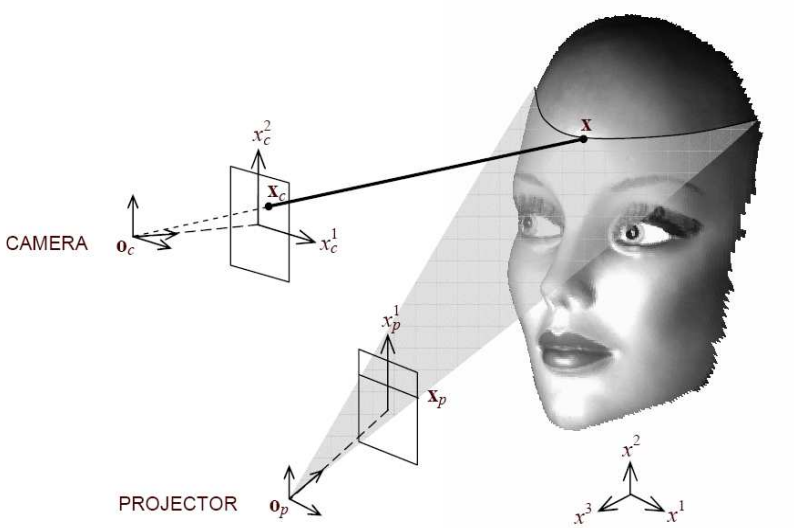

Figure 2.1: From Bronstein et al. [6]. Active stereo model.

maps x into x_p in a 1D projector coordinate system. The projector casts a structured light pattern onto the object, which allows the determination of χ_p corresponding to each s_c in the camera image plane. Different types of illumination patterns have been suggested to encode the projection plane such as grey level intensity [10], time varying [37] and colour patterns [41]. World coordinates x are obtained unambiguously from the intersection of the ray $o_c - x_c$ and the plane $o_p x_p$. This approach can be extended to having multiple cameras viewing the scene.

2.1.3 Photometric Stereo

Woodham introduced a technique called photometric stereo to capture the 3D geometry of an object [48]. The technique works by varying the direction of incident illumination between successive images while holding the viewing the direction constant. Since the imaging geometry does not change between images the correspondence of the points on the image are known. The light sources are ideally point sources some distance away in different directions, so that in each case there is a well-defined light source direction from which to measure surface orientation. Therefore, the change of the intensities in the images depends on both local surface orientation and illumination direction. Surface orientation can then be determined from the image intensities obtained with fixed imaging geometry but with varying lighting conditions. Figure 2.2 shows a typical photometric stereo setup with one camera imaging the surface illuminated by multiplexing several different lighting sources.

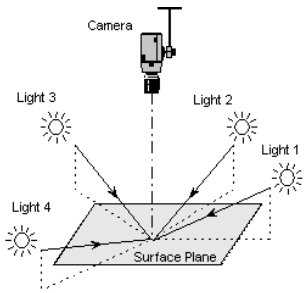

Figure 2.2: From Jerry's Taurus Studio [24]. Typical photometric stereo setup. The surface is imaged by one camera and illuminated from several different positions over time.

Brostow et al. extend this approach to use multispectral photometric stereo [8]. The advantage of using multispectral light photometric stereo is that it can recover a dense normal field from an untextured surface. The algorithm is used to reconstruct the 3D geometry of untextured moving cloth from a video sequence which is filmed under spatially separated red, green and blue light sources. This method allows the acquisition of high resolution models of moving objects.

2.1.4 Shape From Silhouette and Space Carving

Another method is to extract the object's silhouette in many images and use them to estimate the 3D shape. The shape from silhouette algorithm works by taking a set of images captured simultaneously of the scene from a set of calibrated cameras. Then the algorithm divides each image into background and foreground (silhouette). The 3D shape is recovered from the intersection of each of the 3D volumes defined by the silhouettes (see Figure 2.3). Laurentini proposed a visual hull model [28] for reconstructing models of objects which move over time. Matusik et al. describe a real time implementation of this type of model which captures dynamic objects [33]. They surrounded their object of interest with a set of calibrated cameras. The silhouette is calculated by labelling each pixel in the cameras as either background or foreground (belonging to the subject). The advantage of the shape from silhouette approach is that it is fast and can capture the moving shape of an object in real time. The disadvantage is that the resulting depth reconstruction does not always capture the true shape of the subject. This is because it is very difficult to represent some concave regions as silhouettes. It is also necessary to have many cameras imaging the scene.

Another method of capturing the 3D shape of an object is to use shape carving. Shape carving works best on static objects where computations time is not an issue. Kutulakos and Seitz proposed a method called space

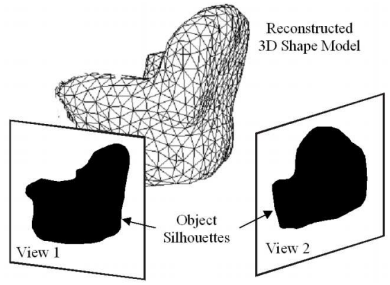

Figure 2.3: From Prince [38]. The shape from silhouette algorithm works by taking several images of the scene simultaneously and then using the silhouettes in each image to reconstruct the 3D shape.

carving to reconstruct the shape of an object [26]. It captures photorealistic models by using a single camera to image an object, multiple times, from different viewpoints. The algorithm can be likened to a sculptor who starts with a big block and removes unwanted pieces until they are left with the desired object.

2.2 3D Face Recognition

This section reviews current research in the area of 3D face recognition. 3D face recognition is a method of face recognition in which the 3D geometry of the face is utilised. Up until recently the majority of face recognition research has focused on the use of 2D intensity images.

2.2.1 Background

Prince et al. define three different groups of face recognition algorithms [40]:
Appearance (distance) based models - in which weighted sums of pixel values are used as features for the recognition decision. Turk and Pentland transforms image data into a feature space based on the principal components of the pixel covariance [45]. These distance based methods fail when there is large changes of pose for the individual.
Geometric based models - a single probe image at one pose is used to create a full 3D head model of the subject based on just one image; parameters representing the pose and illumination are also taken into account. Geometric based methods represent the state of the art in face recognition but can be complex to implement and computationally expensive.
Probabilistic based models - the statistical relationship is used to re-render frontal faces in non-frontal views, or vice versa, and then, standard 2D face recognition methods are used. The features of the face are transformed into a pose-invariant space. Identity is assigned, based on distance is this space. The recognition performance of these methods is not as good as the 3D geometric approaches but they have the advantage of speed and simplicity of implementation.

There is no current method for face recognition that provides good performance for large changes in pose. Current algorithms still have difficulty in performing recognition under varying lighting and pose conditions. Bronstein et al. performed a simple texture mapping experiment to show the limitations of 2D face recognition by wrapping an image of the individual's face is onto a different 3D geometric face model [6]. The experiment reveals the intrinsic weakness of all 2D face recognition approaches; the face is a 3D object, and using only its

2D projection can be misleading. Figure 2.4 shows that individuals are still easily recognisable even when their texture information is placed on a different 3D model.

Figure 2.4: From Bronstein [7]. A visual experiment showing that for the human eye, 2D information plays a more important role in the recognition of a face than 3D information, though such information can be misleading (the faces of Bush and Bin Laden in this example are just drawn on the 3D facial surface of the same person).

A different approach to face recognition is to use 3D models of faces instead of just pixel intensity information. It has been shown that 3D face recognition methods can achieve significantly higher accuracy than their 2D counterparts by utilising the geometry of rigid features on the face during recognition. There are several different methods listed below which attempt to solve the problem. Early research was hampered by the lack of a standard dataset for testing, but the Face Recognition Vendor Test in 2006, provided the first benchmark for 3D face recognition technologies [15]. It provides a dataset which can be used to assess the recognition rates of different algorithms.

2.2.2 Recognition by Profile Silhouette Extraction

Early work in 3D face recognition focused on finding cross sections through faces and using these extracted profiles to perform recognition. One reason for the early adoption of this approach is that it is less computationally expensive then trying to match a whole 3D model of a face.

Cartoux et al. performs recognition by comparing the profiles of two faces [11]. They segment the range image of an individual's face based on its principal curvature and find a plane of bilateral symmetry through the face (see Figure 2.5). This plane is used to normalise the pose. Recognition is performed by calculating the rigid transformation matching the two sets of profiles (the best geometrical transformation fitting the two faces, which is computed using least squares). They report recognition rates of 100% by matching the profiles of faces in their small dataset. The work is important in that it is an early example of recognition using depth information.

Nagamine et al. again use profile and curve matching to perform recognition [36]. They use five manually selected feature points (inner corner of the eyes and the top, bottom and bridge of the nose) on each face to normalise each face pose. The authors note that cross sections which intersect the face vertically through the nose and mouth, contain the most variation in shape between different individuals. Experiments are performed with a dataset of sixteen male subjects, with ten images per subject. The best recognition rates are achieved using vertical profile curves that pass through the central portion of the face. The main contribution of this work

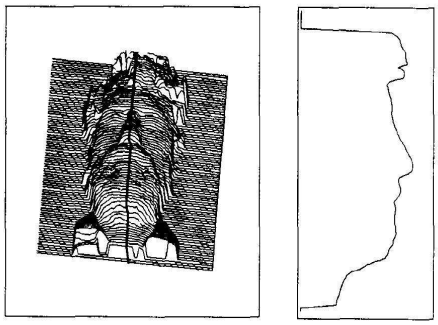

Figure 2.5: From Cartoux et al. [11]. Face surface and profile curve.

is that it shows that the profiles which cross the central region of the face express the face's 3D distinctiveness more effectively.

Tanaka et al. treat face recognition as a 3D shape recognition problem of free-form curved surfaces [43]. The advantage of this approach is that it does not require face feature extraction or surface segmentation. Faces are represented as Extended Gaussian Images (EGI) which are constructed by mapping principal curvatures and their directions at each surface point onto two unit spheres, each of which represents ridge and valley lines respectively. Recognition is then performed using Fisher's spherical correlation on the EGIs [18]. Experiments are reported on a set of 37 images from the National Research Council of Canada range image dataset, with recognition results of 100%.

The advantage to these approaches is that they are invariant to changes in lighting conditions as only the geometry of the face is used. The disadvantage is that they are based on extracting rigid structures from the face and therefore not very good at handling changes in expression and facial shape.

2.2.3 Shape Matching

Recognition based on shape matching attempts to exploit the geometric structure of the face. Most of these algorithms perform some form of pose normalisation on the data and then attempt to iteratively align probe face with a gallery of faces. The matching score (or geometric similarity) is generally used as a metric to score how well the faces match.

Lu et al. propose an approach to 3D face recognition which matches 2.5D scans of faces to 3D models [31]. The authors define a 2.5D image as being a simplified 3D (x, y, z) surface representation that contains, at most, one depth value (z direction) for every point in the (x, y) plane. The method can be broken down into four main steps: Firstly, automatic feature point detection is performed on the 2.5D scans. This step finds a minimum of three corresponding points to calculate the rigid transformation between any two sets of 3D points. Secondly, using the three feature points (inside eye, outside eye and mouth) calculated in the previous step the 2.5D scan is coarsely aligned to a predefined 3D model. This is achieved by applying a rigid transformation using the least square fitting between the triangles formed from the two sets of three points (see Figure 2.6). The next step is a fine alignment of the 2.5D scan and 3D model using a hybrid Iterative Closest Point (ICP) algorithm [2]. ICP iteratively refines the transformation by alternately choosing corresponding control points in the 3D model and finding the translation and rotation that minimises an error function based on the distance between

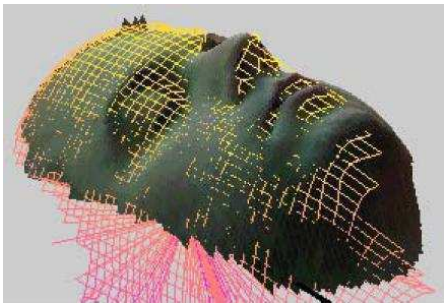

Figure 2.6: From Lu et al. [31]. Result of a coarse alignment of a 2.5D face scan aligned to a 3D textured model from the same person.

them. Regions around the eyes and nose are chosen as control points as they do not change as much as other areas (such as the mouth) because of changes in facial expressions. The final step takes the root mean square distance of the ICP algorithm as the matching score between the scan and the model. The authors abandon the automatic feature point detection algorithm, in the testing phase, in favour of manually selected points due to the presence of outliers. The model is tested using a small dataset of 113 scans of the same 18 subjects.

Bronstein et al. represent one of the first attempts to explicitly tackle the problem of varying facial expression in 3D face recognition [6]. The facial surface is treated as a deformable object and thus finding an expression-invariant representation of the face is essentially equivalent to finding an isometric-invariant representation of the facial surface. A computationally efficient invariant representation of isometric surfaces can be constructed by isometrically embedding the surface into a low dimensional space with convenient geometry. Using this embedding, the task of comparing deformable objects (faces) is transformed into the simpler problem of rigid surface matching. Instead of using the ICP algorithm (the standard choice for surface matching) the authors favour a more computational efficient algorithm based on high-order moments [42]. Experiments are performed on a dataset of 220 faces from 30 different subjects (including a set of identical twins), with facial expressions classified into ten different groups. The algorithm described, significantly outperforms both the straightforward 3D face recognition (rigid facial surface matching) and the classical 2D algorithm (eigenfaces) for neutral expressions.

The results for shape matching algorithms provided are inconclusive but suggest that 3D polygonal matching by itself is not enough to perform 3D recognition, additional information such as colour and texture is needed. This is in line with results from the FRVT 2006 challenge in which shape and texture algorithms outperform shape only algorithms [15]. One reason for this is that the ICP algorithm favours matching large similar areas such as the forehead but for recognition these are the areas which contain the least amount of identity information.

2.2.4 Feature Based Recognition

In contrast to shape based matching approach feature based recognition does not utilise geometric information explicitly. Instead feature points are extracted (automatically or manually) from the data and used to project the data into a feature space where classification is performed. 2D texture information can be used in combination with 3D depth information and can be referred to as multi-modal recognition.

Gordon explorers 3D face recognition, first, by manually labelling features extracted from range images [21]. Then a curvature based segmentation of the face is performed. A set of features are extracted that describe both curvature and metric size properties of the face. Each face is then represented as a point in feature space, and matching is achieved using a nearest neighbour approach. The authors note that the values of the features used are generally similar for different images of the same face, except with cases where there is variation due to expression. Recognition rates of between 80% and 100% are reported.

Lee et al. perform 3D face recognition by forming a feature vector, based on contours, along the face at a sequence of depth values [30]. First, they locate the nose tip and then normalise all the faces by placing them in a standard spacial position, by panning, then rotating and finally tilting. Using a dataset of 70 face models the matching face is included among the best five ranked matches in 94.3% of the cases. The main contribution of the work is to show that recognition can be performed using a minimum amount of parameters and computation if features are chosen which represent the most identity information.

Wang et al describe a method which uses both 3D range data as well as 2D greyscale facial images [47]. Feature points are described by Gabor filter responses in 2D and "point signatures" in 3D to perform multi-modal recognition. The 2D and 3D features are projected into their own PCA subspace and then together form a feature vector. Classification is done on the data of reduced dimensionality using a support vector machine (see Figure 2.7) with a decision directed acyclic graph. The dataset includes models of 50 individuals with

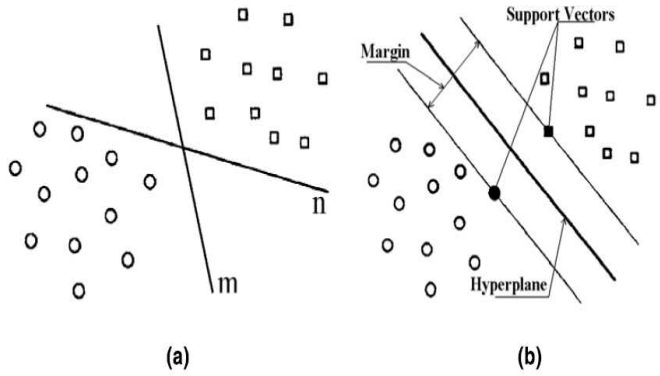

Figure 2.7: From Wang et al. [47]. (a) Arbitrary hyperplanes m and n; (b) the optimal separating hyperplanes with the largest margin.

six images per subject and with pose and expression variations for each model. Results show that the highest recognition rate is achieved by considering both the shape and texture features and choosing less than thirty eigenvectors to construct the subspace.

The advantage of this feature based recognition approach is that the intensity and depth information at each pixel can be concatenated to form one data vector for each individual. Then standard recognition methods such as principal component analysis or factor analysis can be used on the data.

2.2.5 Other Approaches

This section provides a summary of other notable approaches to 3D face recognition which do not easily fall into any of the above categories.

Blanz proposes a morphable model of 3D faces for face recognition [5]. The morphable model is used as a preprocessing tool to generate frontal views from non-frontal images which are then input into a 2D image based recognition system (see Figure 2.8). With this viewpoint normalisation approach the morphable model

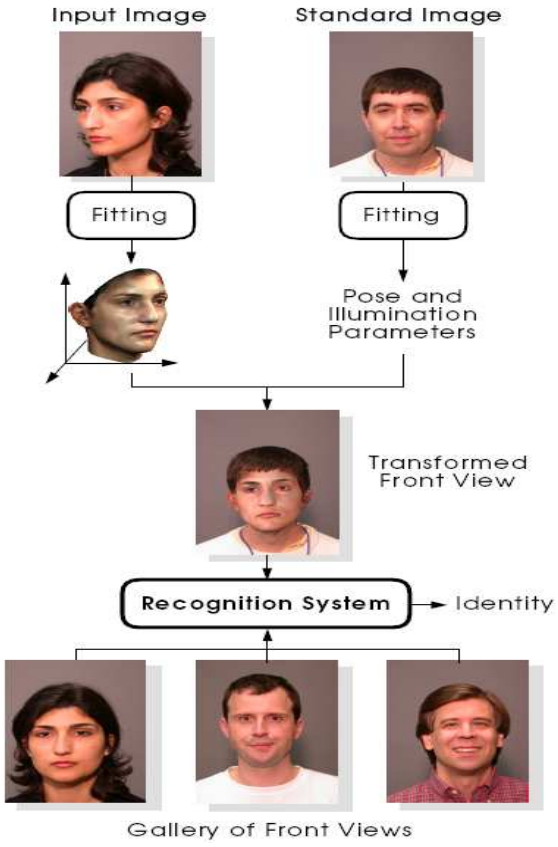

Figure 2.8: From Blanz [5]. Viewpoint normalisation: From a probe image (top left), the morphable model generates a transformed front view. This image is then inputted into a view based face recognition system for comparison with the set of frontal gallery images.

estimates the 3D shape and texture of the face, and then the face is rendered in a frontal pose, at a standard size and illumination. Experimental results using the FERET 02 dataset [16] indicate that not much diagnostic

information on identity is lost when the transformed image is rendered and subsequently analysed in this way.

Faltemier et al. present a method which performs recognition by matching small subregions of the face [17]. Their algorithm exploits subregions (initially 38 or which a subset of the best performing 28 are chosen) of the face that remain relatively consistent in the presence of expression variation. Matching is calculated using the ICP algorithm on each region. Rank one recognition rates of 97.2% and verification rates of 93.2% at a 0.1% false accept rate are reported on the FERET Grand challenge v2 dataset (4007 face scans of 466 unique individuals) [14].

2.3 3D Face Models

This section describes research into reconstructing 3D shape models of human faces. The goal is to take a 2D image of a person's face and reconstruct its 3D shape using only the 2D image and some general 3D model. There are two main approaches: the first is to specify some sort of 3D face model manually and the second is to learn the model from a dataset of faces. These 3D models of faces can be used to generate images of individuals from novel viewpoints with new pose and illumination, or even to create photorealistic 3D models of new human-like faces.

2.3.1 Predefined Geometric Models

The first work in reconstructing 3D models of humans faces, from 2D images, relied on using predefined geometric 3D models of faces and aligning key feature points from the 2D image to the model. Akimoto et al. describe a method which creates a 3D model of an individual, using a front and profile image [1]. The test face is aligned with a generic head model using automatically located feature point matching. Their system does not create a completely accurate 3D facial model as the shape of the generic head model effects the resulting model created from the test image. Due to this constraint, it is difficult to model faces with large variance in appearance. Another limitation of this work is the need for multiple images of the face which need to be taken simultaneously. Tang et al. again use a generic head model to create a 3D model of a face using multiple images [44]. Their major contribution is the use of a multiresolution approach for automatically finding feature points on the 2D face images.

Chowdhury et al. [12] use structure from motion algorithm (SfM) combined with a generic face model to reconstruct 3D models of faces from video sequences. The SfM algorithm obtains a 3D estimate purely from the input video sequence. A Markov chain Monte Carlo (MCMC) framework is used to minimise the energy function when combining the generic model with the 3D estimate. The main contribution of the work is the way the authors incorporate the generic model after the first using SfM algorithm to estimate the structure of the face in order to prevent the solution from converging early.

There is a major disadvantage to using a generic head model; it does not generalise well to different types of face shapes (i.e the solution converges very close to the initial value of the generic model). The model is immediately constrained to the shape of the generic head model.

2.3.2 Learned Models

Current work is focused on creating face models which are learned from a dataset of faces. Vetter and Blanz present a method to derive 3D shape and surface texture for a human face from a single image [46]. The method is based on a general flexible 3D face model which is learned from a dataset of 3D face scans (see Figure 2.9). A modified optical flow algorithm is presented which establishes correspondences between the reference image

Figure 2.9: From Vetter et al. [5]. Top row - Reconstruction of 3D shape and texture from FERET images. Second row - Results are rendered into the original images with pose and illumination recovered by the algorithm. Third row - Novel views.

and the dataset. Using the predicted 3D model new images of the face can be generated from new viewpoints and with differing illumination conditions. There is one simplification in the experiment; all the images are rendered from the test set of faces where both projection parameters and illumination conditions are know.

Jiang et al. [25] use frontal face detection and alignment to locate 83 facial feature points on face images with normal illumination conditions and neutral expression. The 3D shape is reconstructed according to the feature points and a 3D face database. The 2D image is then projected onto the 3D shape to create a textured model; a linear interpolation algorithm is used to fill in areas of unknown texture using the known colours in the vicinity. Using this shape 3D model, virtual 2D images can be generated with different pose, illumination and expression (see Figure 2.10). The algorithm takes approximately four seconds to perform the reconstruction task on a 512×512 image using a Pentium 4 1.3 GHz processor with 256Mb of RAM; this is a major speed improvement over Vetter et al., which takes an average of one minute per face. Another advantage to their approach as opposed to Vetter et al. is that it is fully automatic and does not require any manual initialisation.

2.4 Summary

This literature review has provided an overview of work in the areas of 3D data capture, 3D face recognition and the creation of 3D models of human faces. The rest of this report describes an active stereo system for reconstructing 3D shape models of human faces. A probabilistic factor analysis model is then used to predict the 3D shape of a face from a 2D intensity image. Finally, 3D face verification is then performed using a feature

Figure 2.10: From Jiang et al. [25]. 2D face with feature points marked in red. The 3D reconstructed face, 3D face with texture. Novel view rendered with new pose, illumination and expression.

based approach where the pixel intensity and depth information are combined for each individual.

Chapter 3

Capturing Data

This chapter describes an active stereo system which reconstructs 3D shape models of human faces with texture. The system reconstructs the shape by using a camera to take an image of the subject's face while projecting a known structured light pattern onto it. The depth at each point is then calculated and a 3D shape model is created. The system is a modified version of previous work which uses infra-red light [39]. The main contribution of this work is to; make the system run on new hardware (camera and projector), capture larger resolution models, reconstruct the shape using structured light patterns which are projected at a different orientation and to use visible instead of infra-red light. The finished system reconstructs a model of the subject's face in under two seconds. It is possible to use this system to capture the 3D shape of any object (as long as the surface is not very reflective or transparent) but for the purpose of this report only human faces are captured.

3.1 Hardware and Calibration

3.1.1 Hardware

The system consists of an IDS UI-1645LE-C USB 2.0 camera and a BENQ MP622c projector. The camera is capable of taking images of 1280×1024 pixels at 20 frames per second. The projector operates at a resolution of 1280×1024 pixels and images at 60Hz. Figure 3.1 shows the setup of the camera and projector. By moving the position of the camera it is possible to increase the distance (baseline) between the camera and projector. It is possible to capture faces at a distance of up to one meter from the hardware. The projector is positioned horizontally but the camera is placed on its side so its imaging plane contains more face data and less unnecessary background information (i.e. the long axis of the camera is aligned with the long axis of the face).

3.1.2 Camera Model

In order for the reconstruction to work there must be working models for the projector and camera. These models tell us how points in the real world are projected to points in the camera image plane and vice versa for the projector. There are three types of parameters that are necessary; the intrinsic parameters for both the camera and projector and the extrinsic parameters which explains the relationship between them.

Consider a point (X, Y, Z) in the real world with the camera at the origin of this space and the camera plane perpendicular to its positive z-axis. We want to know to which pixel (x, y) in the image plane this point will

Figure 3.1: Camera and projector used for 3D capture. The camera is positioned on a adjustable mounting so it is possible to alter the distance between the camera and projector if necessary.

appear. The intrinsic matrix of the camera gives us this relationship:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \alpha & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X/Z \\ Y/Z \\ 1 \end{bmatrix} \tag{3.1}$$

where f_x is the focal length of the camera in the x direction, f_y is the focal length in the y direction and o_x and o_y are the pixel offset origins in the x and y directions respectively. The skew term α has no obvious physical interpretation but can be thought of as approximately compensating for cases where the image plane is not precisely perpendicular to the optical axis; its inclusion improves the predictive quality of the camera model.

The intrinsic matrix for the projector is similar but the directionality is reversed. We now imagine a point in the image projecting out into the world in a ray of pixels with constant $(X/Z, Y/Z)$. The projected position of the pixel will be where it intersects with the face.

The camera and the projector are offset from each other a small distance (baseline) and they do not point in the exact same direction. It is unrealistic to assume that they both lie at the origin. We say the camera is at the origin with the camera plane orthogonal to the z-axis and the projector is not and points a different direction. We define a 3D Euclidean transformation called the extrinsic matrix which represents the spatial relationship between the camera and the projector:

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.2}$$

where the transformation takes the form of a 3×3 rotation matrix and a 3×1 translation vector. There are actually only six free parameters due to the dependences between the components of the rotation matrix.

3.1.3 Calibration

In order to find the parameters of Equations 3.1 and 3.2 the system must be calibrated. The camera and the projector must be calibrated to find their relationship between the angular direction in the world and each pixel

in the camera and projector images (their intrinsic matrices). Also the spatial relationship between the camera and the projector needs to be found (the extrinsic matrix).

Camera Calibration

The principal behind calibrating the camera is straightforward; an object with known geometry is placed into the world and the camera is used to take its picture, then known points in the world and image are used to estimate the camera parameters. It is necessary to know the exact position of several points on the object and these points must be identifiable in the camera image. The relationship between points in the object co-ordinate system and the camera co-ordinate system are given by Equation 3.2 and the relationship between points in the camera co-ordinate system and the image position in pixels (x, y) is given by Equation 3.1.

The approach used for calibration is to adjust the parameters $\theta = \{r_{ij}, t_i, f_x, f_y, o_x, o_y, \alpha\}$ so that the predicted camera points (x, y) have minimised square error from measured points in the image (x_m, y_m). It is possible to find a find approximate closed form solutions for the intrinsic parameters $\theta_i = \{f_x, f_y, o_x, o_y, \alpha\}$ [50]. Once the estimated solutions are obtained, the Levenberg-Marquardt algorithm is used to improve the solution by optimising the the mean squared error of the predicted and measured points [29].

Calibration requires a real world object with visually distinctive features at known locations. For this report a checkerboard of known dimensions is used, see Figure 3.2. The corners of the board are marked by the user

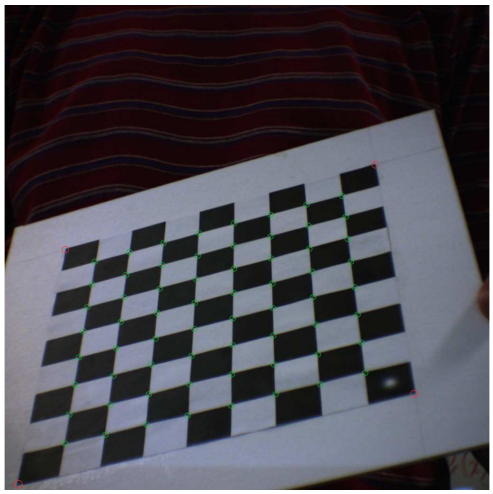

Figure 3.2: To calibrate the camera the relationship between known 3D points in a world co-ordinate system and the camera points is needed. A flat printed planar checkerboard of known size is is often used to accomplish this task. The red circles represent corners marked by the user and the green circles are found automatically using a homography.

and the positions of the intermediate corners are found using an estimated homography transformation. The Harris-Stephens corner detector is then used to accurately find the corner positions [22]. It is necessary to view the checkerboard from several different angles to get enough information to estimate the intrinsic parameters. In practice this means capturing at least ten images of the board at different orientations.

Projector Calibration

The method for calibrating the projector is the same as for the camera but has to be altered slightly as the projector is not an imaging device and is unable to measure where real world points are imaged. The solution to this is to use an image of a known checkerboard and to project this out into the world onto a black planar surface and measure where its corners appear, see Figure 3.3. The camera is used to take pictures of the planar

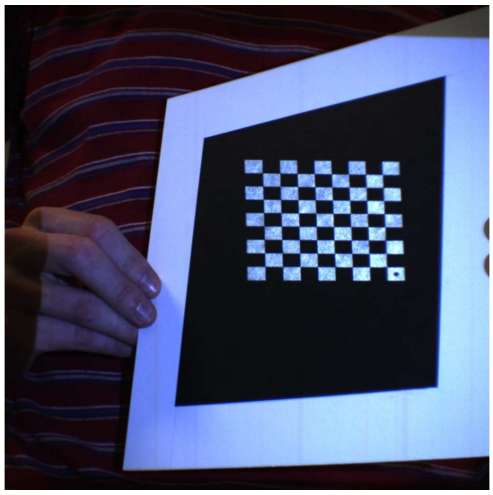

Figure 3.3: A checkerboard image is projected by the projector onto a black planar surface. The corners of the black planar surface and the projected pattern are marked by hand. The point where the corners of the checkerboard project into the world are measured using the camera. Using the known positions of the black surface's corners the position of the checkerboard squares can be calculated using a homography.

surface while the checkerboard pattern is projected on to it. The exact positions of the corners in the world co-ordinate system of the black planar surface and the checkerboard are marked by the user. Again we have the marked corners of the board so we can find the positions of the intermediate corners using an estimated homography transformation. We now have a set of corresponding points in the image and in the real world which are used to calculate the rotation and translation relating the co-ordinate system of the plane to the projector.

Extrinsic Parameters

The final step in calibration is to estimate the Euclidean transformation relating the camera and projector position. By calculating the intrinsic matrices for the camera and the projector it has be shown that it is possible to calculate the transformation from the real world to camera co-ordinate system, $\mathbf{T}_{w\to c}$ and the transformation from the real world to the projector, $\mathbf{T}_{w\to p}$. From this it is possible to find the transformation relating the camera co-ordinate system to the projector co-ordinate system:

$$\mathbf{T}_{c\to p} = \mathbf{T}_{w\to p}\mathbf{T}_{w\to c}^{-1} \tag{3.3}$$

The transformation can be estimated using each frame from the projector calibration stage. Using the average of the estimates as a starting point, the estimate can be improved by minimising the error between the predicted and actual checkerboard corners positions in the camera and projector images.

3.2 Data Capture

Now that the camera and projector are calibrated it is possible to begin capturing data. The camera captures two images in quick succession which are used to reconstruct the 3D shape of the subject's face. The first image is captured under standard lighting conditions and is referred to as the texture image. The second image is captured just after the texture image while the projector is simultaneously projecting a structured light pattern onto the subject's face and is referred to as the structure image. The previous system had the projector lying on its side so now the stimulus image is rotated so that it displays correctly. The structured light pattern has a sinusoidal grating of a specified wavelength and can be seen in Figure 3.4. As the two images are captured

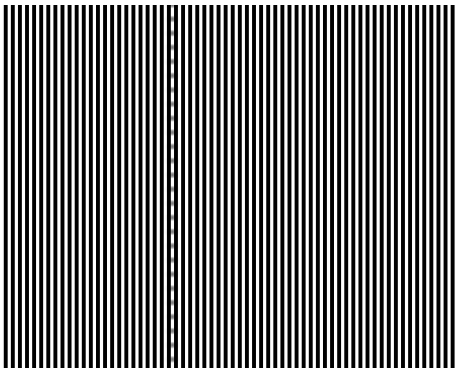

Figure 3.4: A 1280x1024 pixels structured light pattern which is projected onto the subject's face. The pattern has a sinusoidal grating running horizontally with a wavelength of 20 pixels. It also has a marker band at 480 pixels along the x axis. This marker band has a sinusoidal grating running vertically with a wavelength of twice that of the main grating.

in a very short time frame there should be little difference between the subjects head position or expression as long as they do not move very suddenly. In order to extract only the structured light, the texture image is subtracted from the structure image to produce a difference image:

$$\Delta I = R_{str} + G_{str} + B_{str} - R_{tex} - G_{tex} - B_{tex} \tag{3.4}$$

where R, G and B represent the red, green and blue channels of the structure (str) and texture (tex) images. Figure 3.5 shows a typical texture and structure image and their resulting difference image.

Phase Measurement

Using the difference image the next step is to estimate the grating frequency (from the wavelength of the stimulus image) on the face. This is done by taking the Fourier transform of the difference image in the horizontal dimension to find the frequency with the highest mean energy. The next step is to calculate the phase at every point in the difference image. The phase is measured by convolving the difference image with sine and cosine Gabor functions. A Gabor function is the product of a Gaussian and sinusoidal function:

$$G_{sin}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp[-(x-\bar{x})^2)/2\sigma^2] sin(2\pi x/\lambda)$$

$$G_{cos}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp[-(x-\bar{x})^2)/2\sigma^2] cos(2\pi x/\lambda) \tag{3.5}$$

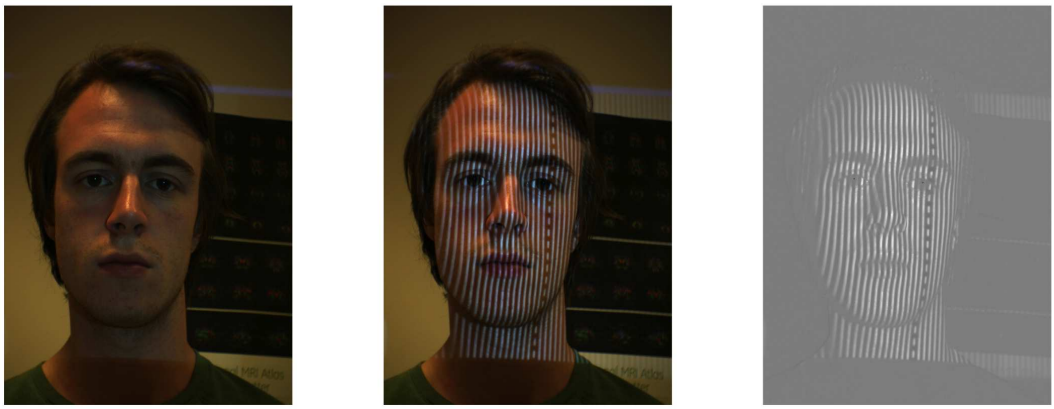

Figure 3.5: (Left) Texture image. (Middle) Structure image. (Right) Difference image; the difference between the first two images.

where $\bar{x}$ is the mean position of the Gabor, σ is the width of its envelope and λ is the wavelength of the underlying sinusoid (known from the stimulus image). The inverse tangent of the ratio between the sine and cosine Gabor functions are taken to calculate the phase:

$$\phi(x) = arctan(G_{sin}(x), G_{cos}(x)) \tag{3.6}$$

Figure 3.6 shows the result of calculating the arc tangent of the convolved sine and cosine Gabor functions at every point in the image. This is referred to as the wrapped phase image and contains the local phase measurements. The next step is to unwrapping this phase.

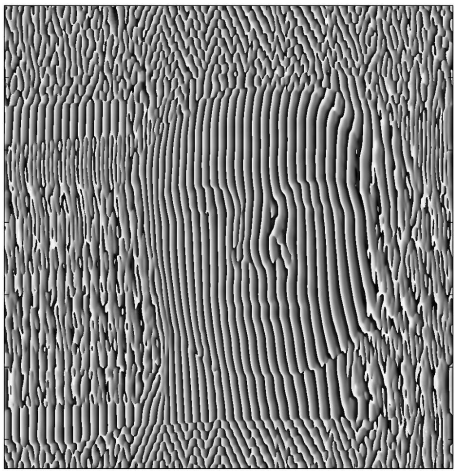

Figure 3.6: On the wrapped phase image, points that are near white have phase near 2π and points near black have phase near 0.

Phase Unwrapping

The wrapped phase image from Figure 3.6 contains only local phase measurements; they are wrapped around 2π. If you take any two points on the image the phase relation is ambiguous by a factor of $N2\pi$. The reason we need to unwrapping the phase is that we wish to known where each vertical line from the stimulus pattern is in the difference image. To do this it is necessary to have an estimate of the absolute phase at every point in the image.

Phase unwrapped in one dimension is accomplished by detecting the wrapped phase jumps and adding an appropriate multiple of 2π radians. Figure 3.7 gives an example of the 1D phase unwrapping problem.

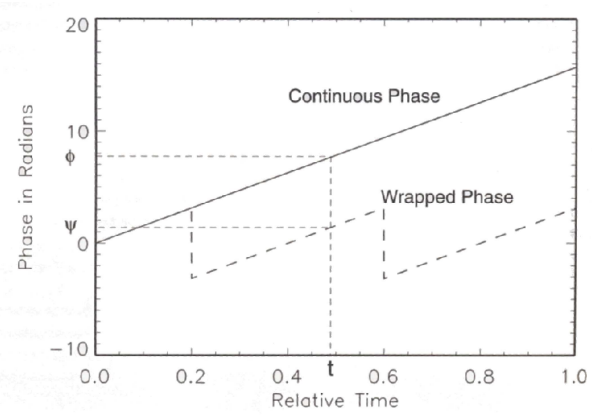

Figure 3.7: From Ghiglia [20]. The true phase is a linear function of t (solid line), whereas the wrapped phase $\psi(t)$ is the sawtooth function (dashed line). Phase unwrapping is accomplished by detecting the wrapped phase jumps and adding an appropriate multiple of 2π radians.

2D phase unwrapping arises most often in interferomic applications such as radar. The 2D phase unwrapping problem is difficult because it is possible that the phase map is inconsistent, the phase at one point relative to another depends on the path taken. This is because of areas which feature sudden changes in phase are attributed to being in the wrong direction. A small negative phase difference between two neighbouring pixels may be due to large positive phase changes. The phase unwrapping algorithm used in this system is a variant of Goldstein's branch cut algorithm [20]. The basic method of the algorithm is to identify areas which are potentially ambiguous and then unwrap the remaining image. The depth is then interpolated over the ambiguous areas to have a complete depth map. Phase residues are defined as four pixels in the image in a 2×2 arrangement that contain phase values so that when they are unwrapped in a clockwise direction the total phase is not 0. The resulting phase is equal to $N2\pi$, but we make the simplifying assumption that $N = \pm 1$; referred to as the positive or negative charge of the residue. It can be shown that the phase image is uniquely unwrappable only when the charges are balanced. Each positive charge is connected to a negative charge by a greedy algorithm which sorts all the residues by their distance from the centre and matches them to the nearest oppositely charged residue. Then if we do not allow the image to unwrap across these connections, the resulting unwrapping will be unique regardless of the order in which we choose to unwrap the pixels. Figure 3.8 shows the wrapped image with its positive and negative residues. As there are much more residues at the edges of the image it is unwrapped from the centre. The centre vertical line is first unwrapped. A modified flood fill algorithm is then used to choose the unwrapping order. To unwrap a pixel with wrapped phase Φ_w from a neighbouring pixel

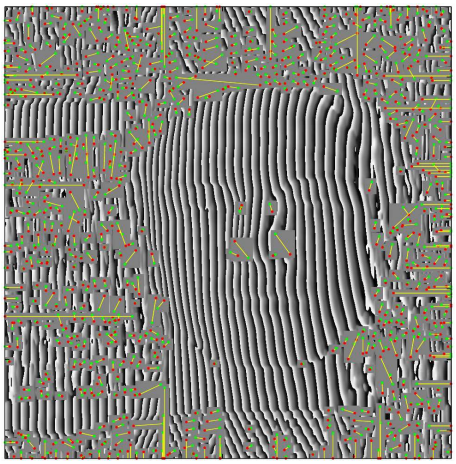

Figure 3.8: The red dots are positive phase residues and the green dots are negative phase residues. These points are greedily matched and points around them are not included in the unwrapping process. It can be seen that that there is a lot of residues in the area that is not on the face.

with unwrapped phase Φ' the following operation is performed:

$$\Phi'_w = \Phi' + wrap\left[\Phi' - \Phi_w\right] \tag{3.7}$$

where the operation $wrap\left[\,\right]$ adds or subtracts 2π until the input falls in the range $[0 \rightarrow 2\pi]$ (i.e. the unwrapping operation adds the wrapped phase change from an unwrapped neighbour to the value of that phase at the unwrapped neighbour). Figure 3.9 shows the result of the unwrapping process. The yellow regions are areas

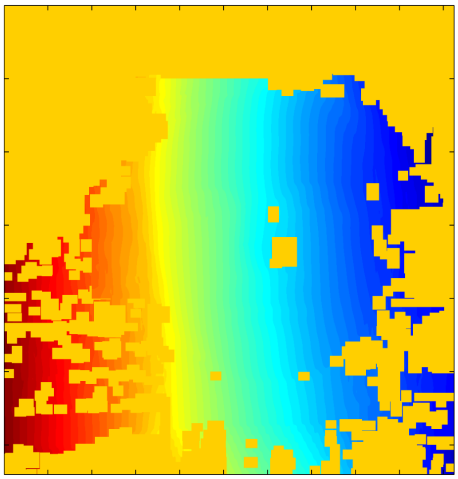

Figure 3.9: In the unwrapped phase image the phase difference between any two points on the image can be calculated. The areas in yellow have bad phase and cannot be calculated.

of bad phase and mostly lie the region off the face. One exception is the area around the nose, this is due to

the nose obscuring some of the structured light. To avoid this problem is advisable for the camera to view the individual from slightly below the nose.

Absolute Phase

The phase image is now unwrapped and this allows the phase difference between any two points in the image to be measured. It is still not possible to determine which bar in the image corresponds to which bar from the structured light pattern. To get the correspondence, we need to find the position of the marker in the image. We already know the approximate wavelength of the marker band (twice that of the main grating). We then convolve the image with horizontal sine and cosine phase Gabors. The local energy is the sum of the squared real and imaginary convolutions:

$$E(x) = (G_{sin}(x))^2 + (G_{cos}(x))^2 \tag{3.8}$$

Each pixel in the image is classified as belonging to discrete cycles of the grating. The marker band is chosen as the cycle with the largest average energy. Figure 3.10 shows the found marker band and the absolute phase image. We now know which bar in the image corresponds to which bar in the structured light pattern.

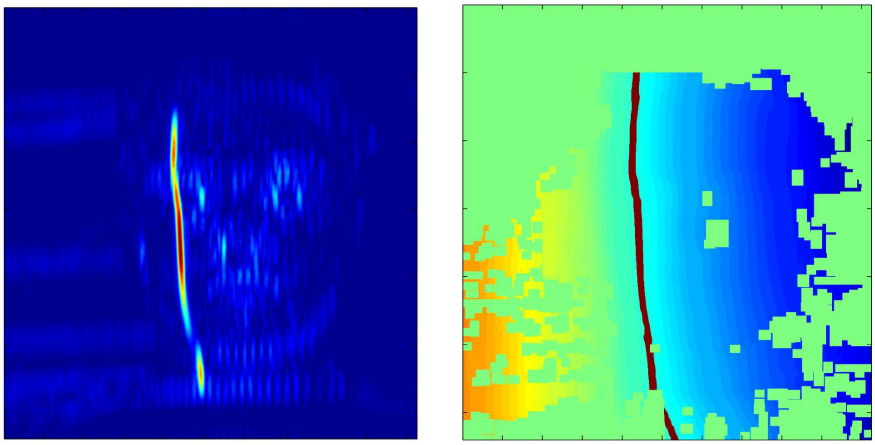

Figure 3.10: (Left) Difference image convolved with horizontal Gabor filters to calculate the filter energy. (Right) The red bar in this image is the bar which has the most energy associate with it (i.e. the grating). This tells us which bar in the structured light pattern corresponds to which in the image.

Calculating Depth

We could use epipolar geometry to triangulate the depth of the image but there is a much faster approximation that can be used instead. By inverting Equation 3.1 we can find expressions for the ratio of the world X and Y coordinates over Z for a point (x, y) in camera co-ordinates such that:

$$\begin{aligned} X/Z &= \frac{x - o_x}{f_x} \\ Y/Z &= \frac{y - o_y}{f_y} \end{aligned} \tag{3.9}$$

The position of a point $\mathbf{P} = (X, Y, Z)$ in the world can be expressed as:

$$\mathbf{P} = \begin{bmatrix} X/Z \\ Y/Z \\ 1 \end{bmatrix} Z \tag{3.10}$$

We can now transfer this point into the co-ordinate frame of the projector:

$$\mathbf{P}' = \mathbf{R}\mathbf{P} + \mathbf{t} \tag{3.11}$$

where $\mathbf{P}' = (X', Y', Z')$. Due to the fact that the old system had the projector on its side we must now find the the depth values Z by looking along the images x axis instead of its y. We can rearrange the previous expression to get the following:

$$X'/Z' = \frac{(r_{11}X/Z + r_{12}Y/Z + r_{13})Z + t_x}{(r_{31}X/Z + r_{32}Y/Z + r_{33})Z + t_z} \tag{3.12}$$

where we know the ratio X'/Z' because we know the corresponding (x', y') in the projector image:

$$X'/Z' = \frac{x' - o'_x}{f'_x} \tag{3.13}$$

We now know all the terms in Equation 3.12 except for Z so we can rearrange to solve for it:

$$Z = \frac{f'_x t_x + t_z o'_x - t_z x'}{(x' - o'_x)(r_{31}X/Z + r_{32}Y/Z + r_{33}) - f'_x(r_{11}X/Z + r_{12}Y/Z + r_{13})} \tag{3.14}$$

Figure 3.11 shows the final calculated depth map which has a value for Z for each location of the texture image. The measurements are in millimetres and areas where depth could not be calculated due to problems with the

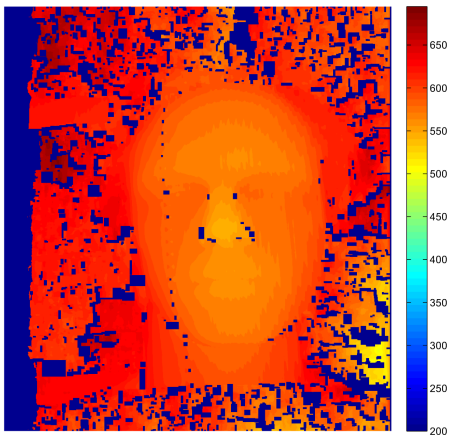

Figure 3.11: This image shows the calculated depth (in millimetres) at each point on the image. Depth of 0 is considered bad depth, i.e. where the algorithm could not calculate the correct answer because of bad phase. It can be seen that these values typically lie off the face where it is hard to determine the phase due to the background being far away.

phase are marked as blue. It is possible to make out the structure of the face in the image where the nose is bright yellow portion in the centre.

3.3 3D Models

It is not possible to calculate depth where there is no phase, so to make realistic looking models the data is interpolated in these missing regions. The approach is to effectively guess the depth in the missing regions by repeatedly blurring a Gaussian over the region of missing data. Figure 3.12 shows an example of the 3D shape model reconstructed from the images in Figure 3.5. The model consists of a set of vertices and triangles. Each vertice contains a (X, Y, Z) position in 3D space and texture co-ordinate (from $[0 \rightarrow 1]$) from the texture map. Triangles are made up from the indices of three vertices in a clockwise order. The texture map is a 512×512 image because graphic cards require textures to be stored with dimensions which are a power of two. It is possible to output the model in VRML format or to display to screen using OpenGL. VRML is a standard text file format for representing 3D graphics. OpenGL allows a much higher resolution model to be displayed. One contribution of this work was to create a viewing application which allows the user to manipulate the position of the model and to view it from any different angle.

Figure 3.12: Reconstructed 3D shape of the face from Figure 3.5. The 3D model is shown viewed from nine different positions.

3.4 Reconstruction Problems

There are several possible problems that can make the reconstruction of the 3D models difficult. Due to the presence of specular reflections it can be difficult to find the structured light pattern on the subject's face. Also some regions of the face such as the eyeballs and hair are not very reflective. The structure light is not reflected at these parts of the face so it is difficult to calculate the phase value at that point. Also parts of the face may be in shadow, in the camera image, due to the distance between the camera and the projector. Figure 3.13 shows the problem of shadowing in the bottom right of the image under the jaw line. Finally, in areas

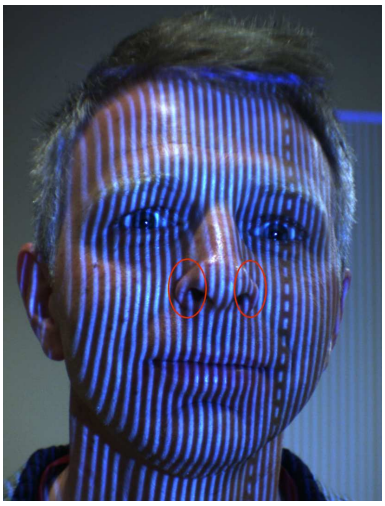

Figure 3.13: This structure image is difficult to unwrap due to the sudden phase change around the nose. There is occlusion because of the nostrils (see the red ellipses) it is very difficult to follow the path of the marker strips down the face.

where there are sudden changes in depth (e.g. around the nose) it can be hard to measure the phase. Figure 3.14 shows an example of a 3D face model which contains errors in reconstruction. The main problem is that the structured light is obscured by the nose (see Figure 3.13). These problems are unavoidable and should be eliminated during data capture.

3.5 Collected Dataset

The final collected dataset consists of 64 different subjects (22 females and 42 males). For each individual the texture image, structure image and resulting depth map is stored. All individuals are between the ages of 20 and 50. Each individual imaged at five different poses; one left, one right and three frontal. For the scope of this project only the frontal images will be used but in future work there is the opportunity to use the two side images to create a more complete 3D model of the head. Figure 4.8 shows the five different texture images from two individuals in the dataset. Due to slightly differing ambient lighting conditions there are small differences between the illumination of the subjects faces (this can be seen by comparing the two rows of Figure 4.8). This is addressed in the intensity normalisation stage. Another problem addressed in normalisation is that each of the subjects is looking at the camera from a slightly different angle. The 3D models must be aligned in three dimensional space.

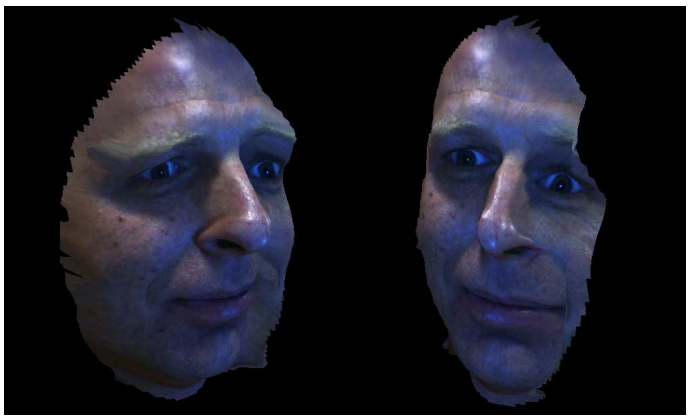

Figure 3.14: Incorrectly reconstructed 3D shape model. Due to problems with the visibility of the marker strip, in the structure image, it is difficult to unwrap the phase correctly and therefore the 3D shape can be misleading. The left hand side of the face is very obviously incorrect.

Figure 3.15: Texture images of two subjects from the collected face dataset. Each individual is viewed from five different poses; one left, one right and three frontal.

3.6 Summary

This chapter has described an active stereo system which reconstructs 3D models of human faces. It has illustrated how the system works and how it goes about reconstructing the 3D models. The system has been acknowledged as being based on previous work and the majority of the contribution of this work is to; understand how the the system works, to get it working again with new hardware, calibrate the hardware and capture data at a different orientation. A dataset of 320 different 3D models has been collected. The collected dataset will be used as the foundation for the inference and recognition performed in the rest of the report.

Chapter 4

Normalising the Dataset

The previous chapter dealt with the reconstruction of the 3D shape of an individual's face using just two images. It also described the collected dataset which consists of the shape models of 64 individuals. Before any inference can be performed on the data it must be normalised. In this case this means aligning the models in three dimensional space so that they all match. A template model is chosen and the rest of the models are aligned to this template. Also each image's pixel intensities must be adjusted to correct for differences in lighting conditions. This is achieved by matching the histogram of the greyscale intensities of the images to that of a template image.

4.1 Aligning Models

When taking images of multiple individuals it is very difficult to ensure; that each individual is looking the exact same direction (there will be variations due to the tilting and angle of the head), that they are the same distance from the camera and in the same position in the x-y plane. To correct for this problem each 3D face model must be aligned in three dimensional space. This is achieved by taking one model as the template and aligning each of the other models to it. The first step is to hand mark landmark points on each of the models in 3D space. Six points are chosen; the corner of the left eye, bridge of the nose, corner of the right eye, left of the mouth, top of the lips and right of the mouth. Figure 4.1 shows the template model and a second model which we wish to align with the landmark points marked. In practice finding the landmark points is achieved by first marking the points on the 2D texture image and then using the following relationship to find where these points are in 3D space:

$$\begin{aligned} X &= \frac{(x - o_x)}{f_x} z \\ Y &= \frac{(y - o_y)}{f_y} z \\ Z &= z \end{aligned} \tag{4.1}$$

Here $[x\, y\, z]$ is the point in 2D image co-ordinates where z is the depth value at the particular $[x\, y]$ position and $[X\, Y\, Z]$ is the point in 3D space. Using the the six landmark points on the model we wish to align we want to find the translation, rotation matrices and scaling factor that best matches them to the template points.

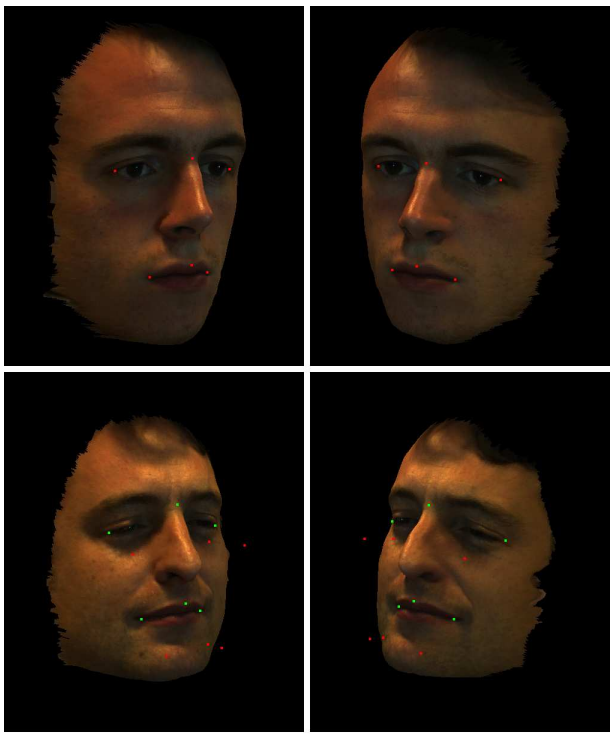

Figure 4.1: (Top row) Template face: all other faces are to be aligned to this model. The red dots represent the six marked landmark points on the face. (Bottom row) The green dots represent the positions of the landmark points on the face for this individual and the red dots are the positions of the landmark points of the template image (the same as the top row). The goal of alignment is to align the green points with the red.

To align a face to the template we need to determine a linear transformation (translation, rotation and scaling) of the face's landmark points B which best conforms them to the landmarks points of the template A. This is achieved by finding the transformation T which minimises the following:

$$|A - T(B)|^2 \tag{4.2}$$

Where each column of A and B is a dimension (x, y and z co-ordinate) and each row is a different landmark point. To find the best transformation we use a form of the orthogonal Procrustes problem. The rotation matrix is found by subtracting the column means ($\bar{A}$ and $\bar{B}$) from A and B and using the singular value decomposition (SVD) to get:

$$\begin{aligned} A_m &= A - \bar{A} \\ B_m &= B - \bar{B} \\ (U\,\Sigma\,V^T) &= SVD(B_m^T A_m) \\ R &= UV' \end{aligned} \tag{4.3}$$

To get the translation we use the mean values of A and B used in the previous step:

$$[t_x\ t_y\ t_z] = \bar{A} - R\bar{B} \tag{4.4}$$

Finally the scaling factor b is found by taking the sums of the main diagonal elements (or Trace) of the following matrices:

$$b = Tr(B_m R A_m^T)/Tr(B_m A_m^T) \tag{4.5}$$

Now we know the transformation that best maps the model to the template and we can simply align the model by finding the new position of each of its co-ordinates with the following relationship:

$$[X'\ Y'\ Z'] = b[X\ Y\ Z]R + [t_x\ t_y\ t_z] \tag{4.6}$$

where $[X\ Y\ Z]$ is the old position of a point in 3D space and $[X'\ Y'\ Z']$ its its new one. Figure 4.2 shows the resulting 2D texture image after alignment to the template. There is some noise visible in the final aligned

Figure 4.2: (Left) The texture image of the template model which the other images are to be aligned to. (Middle) The image of the individual that has to be aligned. (Right) The texture image after alignment. It can be seen that the head pose is different from the middle image also the face is more to the right of the image.

image in Figure 4.2. This is due to the the fact that depth calculation at the edge of the face suddenly changes as the background is reached. So when the model is realigned the noisy estimation of the depth in these regions can obstruct the face.

4.1.1 Storing Aligned Images

Now that the individual's face model is aligned to the template, the depth and the pixel values must be stored. The model is drawn and aligned in OpenGL. The pixel values are read from the rendered model in OpenGL using the *glReadPixels* command which returns a block of pixels of specified size from the frame buffer. This is the new texture image and then saved to the computer as a bitmap file. It is not straightforward to get the new depth value due to the way OpenGL's depth buffer (z buffer) stores data. Figure 4.3 shows the typical OpenGL viewing frustum with its near and far clipping planes. It is advisable to keep the near clipping frame as far from the eye as possible and the far clipping plane as close as possible. The values in the depth buffer are only stored for points lying between the near and far clipping frames. The values are scaled in the range of $[0 \rightarrow 1]$. The depth buffer keeps more precision near the observer, i.e. near the near clipping frame. To get the actual depth in screen co-ordinates of a point from the depth buffer the following equation is used:

$$realDepth = near\ \times\ far/(far - depth\ \times (near\ -\ far)) \tag{4.7}$$

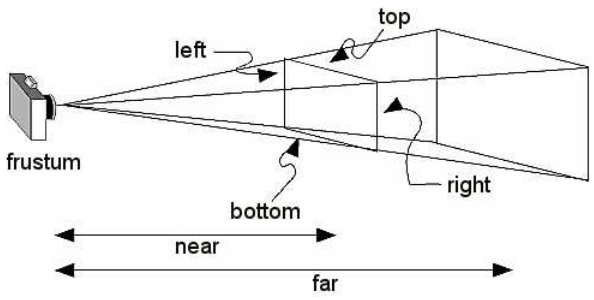

Figure 4.3: From The Official Guide to Learning OpenGL, Version 1.1[49]. OpenGL perspective viewing volume specified by *glFrustum*.

where *depth* refers to the value stored in the depth buffer. The command *glReadPixels* can be used again to get the depth value for the specified pixels. The values are returned from the screen in row order from left to right starting from the lowest to highest row.

4.2 Normalising Image Intensities

In preparing the dataset for the inference stage the images are converted from colour to greyscale to keep the dimensionality of the data low. Unnecessary background information is removed from the texture images by hand marking the background and setting its intensity value's to 0. Currently the data is too large to run any inference on a standard desktop PC in a reasonable time; it needs to be reduced in size. This is achieved by reducing the 1280×1024 texture image and corresponding depth map by half. Then the image is cropped to remove as much of the unnecessary background as possible. This results in a 470×340 texture image and depth map.

Now that the 3D models are aligned the next step is to correct for differences in lighting. Differences can be due to lighting conditions in the room, skin tones or varying light coming out of the projector. Figure 4.4 shows different histograms of intensity images for two individuals from the dataset. The corresponding intensity images of the individuals can be seen in Figure 4.4. To normalise the images for intensity the histograms of the images are matched to that of the template image. This is achieved in MATLAB using the *histeq* function which takes a histogram (the template histogram in this case) and an image as an input and approximately matches the image to the given histogram. Figure 4.4 shows the results of histogram matching.

4.3 Limitations and Discussion

One problem with the method of alignment discussed in section 4.1 is that it can be prone to error. As the 3D shape models contain bad or missing depth values at some points on the face (see Figure 3.11), some selected landmark points may have an incorrect depth value. If this depth value is too large or too small it can have a major impact on alignment. Figure 4.6 shows the result of a bad alignment. One solution to this is to use the interpolated depth value at the landmark point instead of the calculated depth. Another disadvantage to the current alignment method is that it only guarantees that the models are aligned at six points on the models. A more sophisticated alignment algorithm such as the Iterative Closest Point (ICP) algorithm could be used. ICP is a well known registration algorithm which aligns arbitrary 3D shapes. It treats alignment like

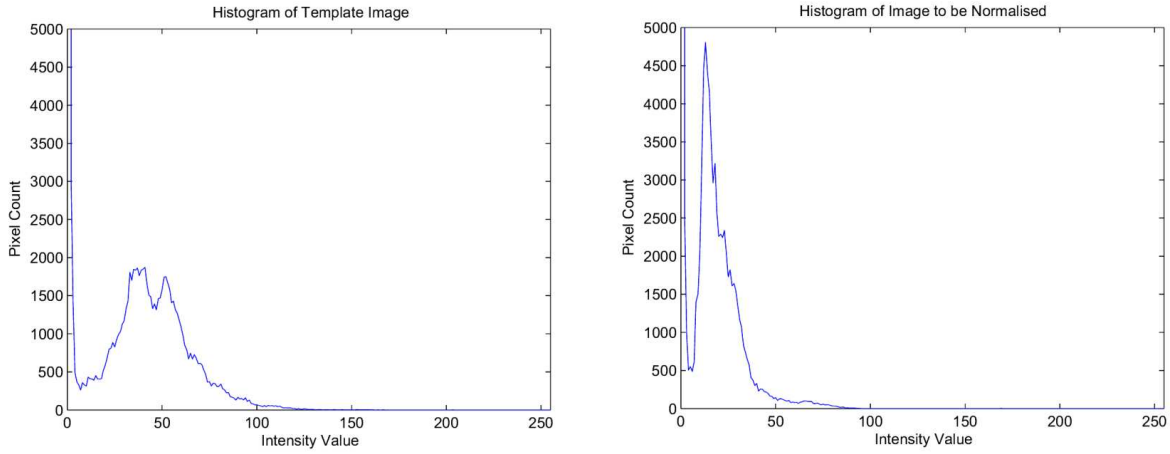

Figure 4.4: In the interest of clarity the y axis is scaled down as the background pixels have a very high count. (Left) Histogram of the template texture image which the other images are to be matched. (Right) Another individual from the dataset with a very different histogram.

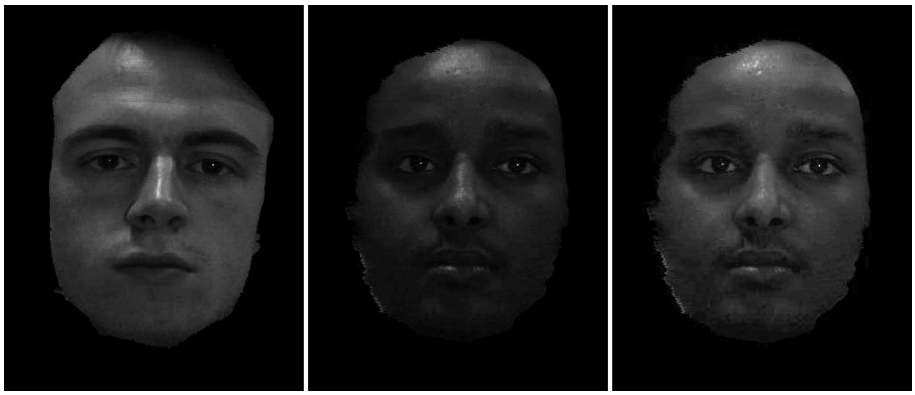

Figure 4.5: (Left) The template image with background removed. (Middle) Another individual before histogram matching. (Right) The same individual after matching the histograms. The intensities values are much closer to the template image.

an optimisation problem and iterates through each point the source model and finds the closest point in the template model. Papatheodorou and Rueckert use the ICP algorithm when aligning their 3D models before performing recognition [34]. Some improvements for future work include the automatic marking of landmark points on the face. Everingham et al. use a Bayesian model of eye and non-eye appearance and a discriminative eye detector trained using AdaBoost to find the position of eyes in a dataset of real world images. [13]. Another improvement would be to use an automatic background removal algorithm on the images. This is a typical example of a colour segmentation problem. In this case it is straightforward, as the background in the images is a plain white wall. Even some form of thresholding whereby pixels of a certain intensity are labelled as background could be used. Friedman et al. use a mixture model for the three different colour channels to segment an image [19]. Also the depth information could be used as the depth values for the background are

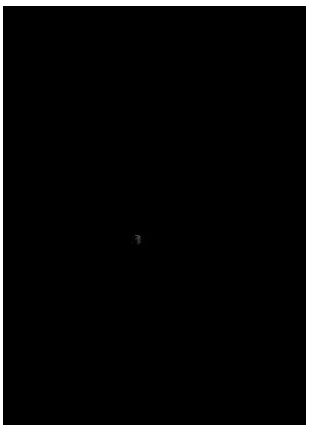

Figure 4.6: Image which has been incorrectly aligned due to bad depth values at the location of the landmark points on the image.

much different that those in the region of the face. Harville et al. utilise depth information for segmenting images [23].

4.4 Final Dataset

As a result of the limitations observed in the previous section some of the individual's are omitted from the dataset. The final dataset $\mathbf{X}$ consists of $n = 60$ observations (individuals) with a dimensionality of $d = 319600$ (the combination of the intensity and depth values):

$$\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \tag{4.8}$$

where each observation $\mathbf{x}_i$ is composed of the intensity ($\mathbf{x}_a$) and depth ($\mathbf{x}_b$) values at every location of the 470×340 image each reshaped to form two row vectors of length 159800:

$$\mathbf{x}_i = [\mathbf{x}_a \, \mathbf{x}_b] \tag{4.9}$$

4.5 Summary

This chapter has taken the texture images and depth maps collected in chapter 3 and aligned the data so that it can now be used for inference. To achieve this the shape models are aligned in three dimensional space to a template model, the background is removed from the images, their image intensities are normalised and the data is resized to make it smaller. The end result is a data matrix of observations which can be be used to learn the relationship between pixel intensity and depth.

Chapter 5

Predicting Depth Using a Multivariate Gaussian Distribution

The goal of this chapter is to be able to take any 2D image of a frontal face, predict its depth and then reconstruct the 3D shape model of the face. This is achieved by taking the dataset from the previous chapter and modelling it using a multivariate Gaussian distribution (MGD). Using the learnt model the conditional distribution of the depth given an image is calculated. Results are provided which show the predicted 3D models for faces of unknown depth.

5.1 Modelling Data

This section describes how the data is modelled as a multivariate Gaussian distribution. It also describes how the parameters of the model are learnt.

5.1.1 Multvariate Gaussian Distribution

The dataset is modelled using a multivariate Gaussian distribution. For a D dimensional vector $\mathbf{x}$ the distribution takes the form:

$$\boldsymbol{\mathcal{G}}_x\left[\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}\right] = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} exp\{\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\} \tag{5.1}$$

where $\boldsymbol{\mu}$ is the mean of the data and $\boldsymbol{\Sigma}$ represents the covariance. Both parameters are learnt from the data using their maximum likelihood estimation:

$$\begin{aligned} \boldsymbol{\mu}_{ML} &= \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \\ \boldsymbol{\Sigma}_{ML} &= \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu}_{ML})(\mathbf{x}_i-\boldsymbol{\mu}_{ML})^T \end{aligned} \tag{5.2}$$

assuming that each observation $\mathbf{x}_i$ is drawn independently from a MVG distribution.

5.1.2 Dimensionality Reduction

Each observation $\mathbf{x}_i$ is of very high dimensionality ($d = 319600$). This is too large to directly model as a multivariate Gaussian, therefore the dimensionality of the data is reduced using principal component analysis (PCA). PCA is defined as the orthogonal projection of the data onto a lower dimensional space, known as the principal subspace, such that the variance of the projected data is maximised. The key idea in PCA is to represent the data as a multivariate Gaussian and find the directions in which the variances of this distribution are largest. These are referred to as the principal components.

The image intensities and depth values are reduced in dimensionality separately. The reason for doing it separately is that later on when we wish to find the depth given an image we will need to be able to find a lower dimensional representation for the image without the depth component. For this example we will only concentrate on dimensionality reduction of the intensity data but it is the exact same procedure for the depth information. Let $\mathbf{I}$ be the matrix which contains each individual as rows and the intensity values for each individual as columns. Standard PCA involves getting the eigen-decomposition of the data's covariance matrix but when the dimensionality of the data is too large this can computationally expensive to compute. To perform PCA on large datasets we first subtract the mean from the data:

$$\bar{\mathbf{I}} = [\mathbf{x}_1 - \boldsymbol{\mu}, ..., \mathbf{x}_n - \boldsymbol{\mu}]$$

Normally PCA involves taking the singular value decomposition of the covariance matrix $\boldsymbol{\Sigma}$ of the data:

$$\boldsymbol{\Sigma} = \bar{\mathbf{I}}\bar{\mathbf{I}}^T \tag{5.3}$$

but in this case the dimensionality of the data is too large. So instead we calculate the following matrix and take its singular value decomposition:

$$\begin{aligned} \boldsymbol{\Psi} &= \bar{\mathbf{I}}^T\bar{\mathbf{I}} \\ &= \mathbf{V}\mathbf{L}^2\mathbf{V}^T \end{aligned} \tag{5.4}$$

Now we can simply calculate the $n \times d$ rotation matrix $\mathbf{U}$ which contains the unit-length orthogonal eigenvectors:

$$\mathbf{U} = \bar{\mathbf{I}}\mathbf{V}\mathbf{L}^{-1} \tag{5.5}$$

where $\mathbf{L}^{-1}$ is the inverse of the square root of the diagonal matrix $\mathbf{L}^2$. We can now reduced the dimensionality of the data by taking the first m columns of $\mathbf{U}$.

$$\mathbf{I}_{RD} = \bar{\mathbf{I}}\mathbf{U}_{1\rightarrow m} \tag{5.6}$$

where $\mathbf{I}_{RD}$ is an $n \times m$ matrix, and $m << d$. To project the data back into the original higher dimensional space we use the following equation:

$$\mathbf{I} = \bar{\mathbf{I}}_{RD}\mathbf{U}_{1\rightarrow m}^T + \mu \tag{5.7}$$

The reduced matrix for the intensities $\mathbf{I}_{RD}$ and depth $\mathbf{D}_{RD}$ values are concatenated together to form the matrix $\mathbf{X}_{RD}$ which can now be used for inference.

5.1.3 Choosing the Number of Principal Components

An important consideration when using PCA is choosing the dimensionality m of the reduced space. The size of m should be much less than d while still maintaining the variance in the data. Figure 5.1 shows the contribution

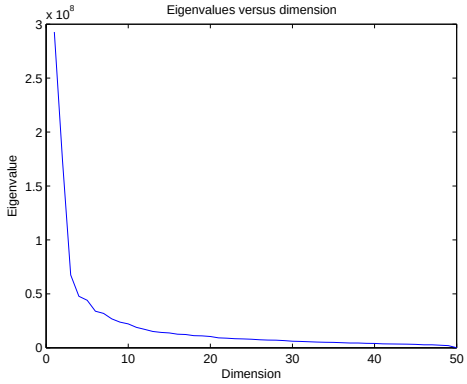

Figure 5.1: The value of the eigenvalues plotted against the number of principal components for the image intensities. It can be seen that over 90% of the variance in the data is contained in the first ten principal components.

of each of the principal components. It is a plot of the eigenvalues (the diagonals of $\mathbf{L}$) against the number of principal components. It can be seen that the first ten principal components (the columns of $\mathbf{U}$) contribute to over 90% of the variance in the data. Figure 5.2 shows the result of representing an image with various different numbers of principal components. It shows that there is not much of a difference in increasing the number beyond 25. It is similar to a form of lossy image compression. As m increases the reconstruction becomes more accurate and would become perfect when $m = d$.

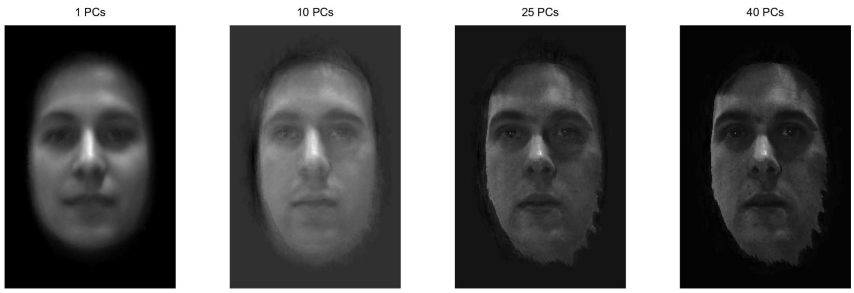

Figure 5.2: Reconstruction of an image using various numbers of principal components. It can be seen that increasing the number of principal components (m) to over 25 does not make much of a difference in reconstruction.

5.2 Predicting Depth

Now that the data is represented as a multivariate Gaussian the next step is to find the depth of a new image that is not in the dataset. This is achieved by finding the conditional distribution of the depth given an image.

5.2.1 Aligning 2D Image

The first step is to align the new image in 2D with the other images in the dataset. This is done by marking three landmark points (left eye, right eye and top of mouth) on the template image and marking the corresponding points on the new image. We then want to find the affine transformation that translates the new points to the ones on the template image. An affine transformation consists of a linear transformation of the x and y co-ordinates plus a translation in the plane:

$$\mathbf{T}_{Aff} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5.8}$$

It has six degrees of freedom, four for the linear transformation and two for the translation. The *maketform* function in Matlab takes the two sets of points as input and returns the affine transformation. Figure 5.3 shows the image we wish to align with the landmark points of the template marked in red and the image after applying the transformation. The image is then greyscaled and reshaped into a column vector of length 159800; let this

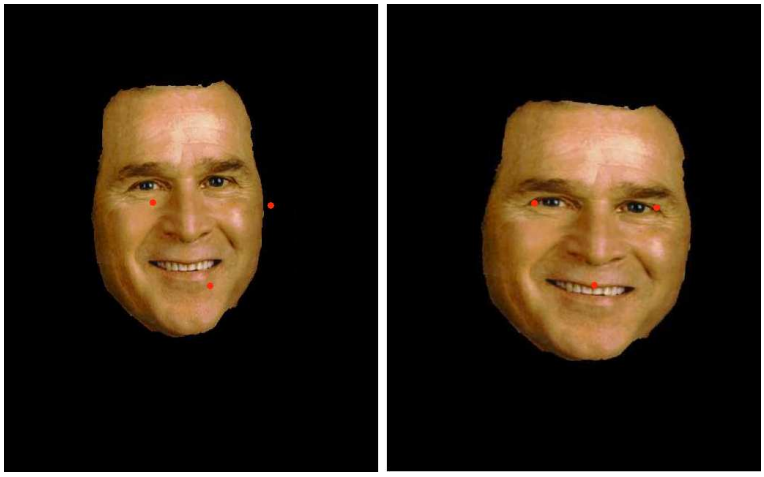

Figure 5.3: (Left) Image of George Bush from [9] with landmark points from the template image marked in red. (Right) The same image aligned to landmark points using an affine transformation.

be called $\mathbf{x}_a$.

5.2.2 Conditional Distribution

An important property of the multivariate Gaussian distribution is that if two sets of variables are jointly normal, then the conditional distribution of one set conditioned on the other is again normal. To find the depth values $\mathbf{x}_b$ for the image $\mathbf{x}_a$ we need to partition the distribution:

$$Pr\left(\begin{bmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{bmatrix}\right) = \boldsymbol{\mathcal{G}}_x\left(\begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}\right) \tag{5.9}$$

Now we find the conditional distribution of the depth $\mathbf{x}_b$ given an image $\mathbf{x}_a$:

$$Pr(\mathbf{x}_b|\mathbf{x}_a) = \boldsymbol{\mathcal{G}}_x(\boldsymbol{\mu}_b + \boldsymbol{\Sigma}_{ba}\boldsymbol{\Sigma}_{aa}^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a), \boldsymbol{\Sigma}_{bb} - \boldsymbol{\Sigma}_{ba}\boldsymbol{\Sigma}_{aa}^{-1}\boldsymbol{\Sigma}_{ab}) \tag{5.10}$$

for the details on how this equation is derived see [3]. The value of the depth will then be the mean of our conditioned normal. We can now take any image and predict a depth value for it using the following equation:

$$\mathbf{x}_b = \boldsymbol{\mu}_b + \boldsymbol{\Sigma}_{ba}\boldsymbol{\Sigma}_{aa}^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a) \tag{5.11}$$

5.3 Results

This section illustrates some results of predicted depth using the method outlined in the previous section. None of the following results are from images of individuals that are present in the the final dataset.

Figure 5.4 shows the resulting 3D model using the predicted depth for the image. There is noticeable noise on the top and the side of the face. This is probably due to missing data values; due to the variation in face shapes, some faces have data values at the edges and others do not. The multivariate Gaussian does not cope well with missing data points. The area around the nose is well rendered and overall the shape looks plausible. Figure 5.5 shows a 3D model for George Bush's face. The main problem with the image is in the area around

Figure 5.4: Noise can be seen on the right and top of the face, this could be due to missing data values.

the nose; the texture does not match well with the shape. The reason for this is that the camera imaging his face is at a much higher angle to his face compared to the previous image and therefore it is not possible to see information for the nostrils. One solution to this problem is to somehow align the input texture image in 3D instead of aligning it in 2D, but this would require a 3D model for the face. Figure 5.6 shows a 3D of a

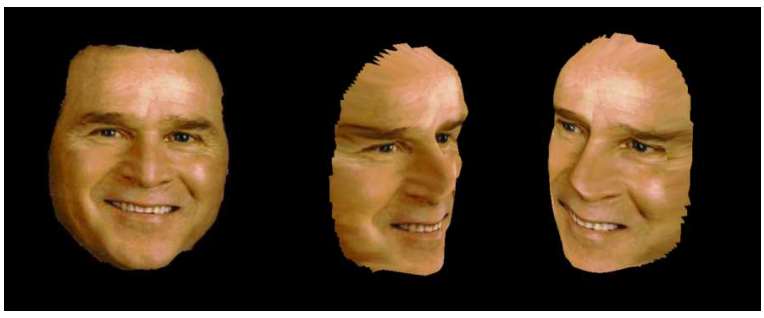

Figure 5.5: 3D model of George Bush's face. The area around the nose does not look realistic.

synthesised photorealistic face from [35]. The image is generated from a dataset of patches of frontal faces and

is not an image from any one individual. The algorithm does a good job of predicting the depth for the face and there is no noticeable problems. The end result is a 3D photorealistic model of a synthesised human face.

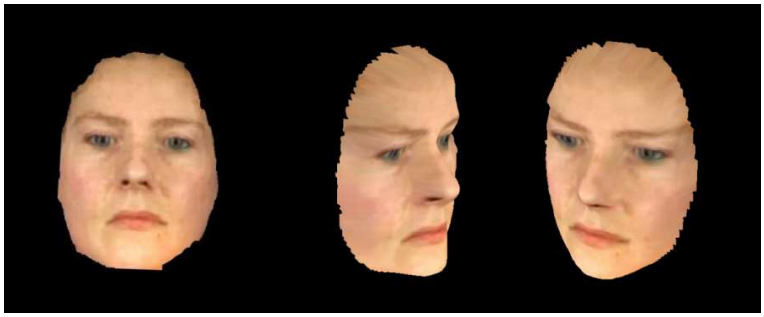

Figure 5.6: Predicted 3D model of synthesised photorealistic face.

5.4 Summary

This chapter has outlined the use of a multivariate Gaussian model to predict the depth for an unknown intensity image. One misleading property of the results is that some data points around the edge of the face are predicted so badly that they are not drawn by the OpenGL viewing application as they are too far away from the face's mean depth. This makes the MVG appear to give better results than it really does. Another disadvantage is that PCA is not a probabilistic model and therefore cannot handle missing data. The next chapter introduces a more sophisticated probabilistic factor analysis model to overcome these problems.

Chapter 6

Predicting Depth Using Factor Analysis

This chapter introduces a fully probabilistic factor analysis model to predict the depth of an image. It also describes a new factor analysis model which takes into account missing data in the learning phase. Both methods are used perform the same depth prediction experiments as the previous chapter.

6.1 Factor Analysis

Factor analysis (FA) is a fully probabilistic method for dimensionality reduction which finds a low-dimensional projection of high-dimensional data that captures the correlation structure of the data. Unlike PCA, FA is measurement scale invariant. In FA a d dimensional multivariate dataset $[\mathbf{x}_1, ..., \mathbf{x}_n]$ is modelled as being a weighted linear sum of k factors $\mathbf{f}_1 ... \mathbf{f}_k$. These weights $\mathbf{h}$ are termed hidden variables and differ for each data point. The rest of the variance is described by a Gaussian noise term. The generative model for FA is given by:

$$\mathbf{x}_i = \boldsymbol{\mu} + \mathbf{F}\mathbf{h}_i + \boldsymbol{\varepsilon}_i \tag{6.1}$$

where $\mathbf{F}$ is a $d \times k$ matrix containing the factors $\mathbf{f}_1 ... \mathbf{f}_k$, the hidden variable $\mathbf{h}_i$ is a $k \times 1$ vector and the term $\boldsymbol{\varepsilon}_i$ is a $d \times 1$ Gaussian noise term with mean 0 and diagonal covariance $\boldsymbol{\Sigma}$. The unknown parameters for this model are $\mathbf{F}$ and $\boldsymbol{\Sigma}$. The probability density function is given by:

$$Pr(\mathbf{x}|\boldsymbol{\mu}, \mathbf{F}, \boldsymbol{\Sigma}) = \boldsymbol{\mathcal{G}}_x \left[\boldsymbol{\mu}, \mathbf{F}\mathbf{F}^T + \boldsymbol{\Sigma}\right] \tag{6.2}$$

To find the the depth values $\mathbf{x}_b$ for the image $\mathbf{x}_a$ we again need to partition the distribution as in Equation 5.9:

$$Pr\left(\begin{bmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{bmatrix}\right) = \boldsymbol{\mathcal{G}}_x \left(\begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix}, \begin{bmatrix} \mathbf{F}_a\mathbf{F}_a^T + \boldsymbol{\Sigma}_a & \mathbf{F}_a\mathbf{F}_b^T \\ \mathbf{F}_b\mathbf{F}_a^T & \mathbf{F}_b\mathbf{F}_b^T + \boldsymbol{\Sigma}_b \end{bmatrix}\right) \tag{6.3}$$

Again we find the conditional distribution of the depth $\mathbf{x}_b$ given an image $\mathbf{x}_a$ and take result for the mean:

$$P(\mathbf{x}_b|\mathbf{x}_a) = \boldsymbol{\mathcal{G}}_x(\boldsymbol{\mu}_b + (\mathbf{F}_b\mathbf{F}_a^T)(\mathbf{F}_a\mathbf{F}_a^T + \boldsymbol{\Sigma}_a)^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a), (\mathbf{F}_b\mathbf{F}_b^T + \boldsymbol{\Sigma}_b) - (\mathbf{F}_b\mathbf{F}_a^T)(\mathbf{F}_a\mathbf{F}_a^T + \boldsymbol{\Sigma}_a)^{-1}(\mathbf{F}_a\mathbf{F}_b^T)) \tag{6.4}$$

$$\mathbf{x}_b = \boldsymbol{\mu}_b + (\mathbf{F}_b\mathbf{F}_a^T)(\mathbf{F}_a\mathbf{F}_a^T + \boldsymbol{\Sigma}_a)^{-1}(\mathbf{x}_a - \boldsymbol{\mu}_a) \tag{6.5}$$

If the dimensionality d of the data is very large it may not be possible to store the matrix $(\mathbf{F}_a\mathbf{F}_a^T + \mathbf{\Sigma}_a)$ in memory as it is of dimension $d \times d$. A solution to this problem is to use the matrix inversion lemma when calculating the inverse:

$$(\mathbf{F}\mathbf{F}^T + \mathbf{\Sigma})^{-1} = \mathbf{\Sigma}^{-1} - \mathbf{\Sigma}^{-1}\mathbf{F}(\mathbf{F}^T\mathbf{\Sigma}^{-1}\mathbf{F} + \mathbf{I})^{-1}\mathbf{F}^T\mathbf{\Sigma}^{-1} \tag{6.6}$$

6.1.1 Expectation Maximisation

For the previous step we need to have values for the parameters $\theta = \{\boldsymbol{\mu}, \mathbf{F}, \mathbf{\Sigma}\}$. The expectation maximisation (EM) algorithm is used to find maximum likelihood estimate of parameters. The EM algorithm alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and a maximization (M) step, which computes the maximum likelihood estimates of the parameters by maximising the expected likelihood found on the E step. The parameters found on the M step are then used to begin another E step, and the process is repeated for a specified number of iterations. The following update equations are used in the M step:

$$\begin{aligned}
\boldsymbol{\mu} &= \frac{1}{N}\sum_n \mathbf{x}_n \\
\mathbf{F} &= \left(\sum_n \mathbf{x}_n E\left[\mathbf{h}_n\right]^T\right)\left(\sum_n E\left[\mathbf{h}_n\mathbf{h}_n^T\right]\right)^{-1} \\
\mathbf{\Sigma} &= \frac{1}{N}\sum_n Diag\left[\mathbf{x}_n\mathbf{x}_n^T - \mathbf{F}E\left[\mathbf{h}_n\right]\mathbf{x}_n^T\right]
\end{aligned} \tag{6.7}$$

as we cannot observe the latent variables $\mathbf{h}_n$ we take their expectated values, $E[\mathbf{h}_n]$. Note the mean μ is calculated in the exact same as for the maximum likelihood method for the multivariate Gaussian in the previous chapter. For more information on the EM algorithm and FA see [3].

6.1.2 FA Results

The EM algorithm was used to learn a FA model for the depth and intensity combined with three factors using the dataset X described in Equation 4.8. The algorithm was run for 100 iterations but in practice much less will still provide adequate results. This is due to the way in which the factors go about aligning to the data, the initial few iterations do most of the work. Figure 6.1 shows the resulting mean face along with the factors and covariance for the portion of the dataset that contains the intensity images. The mean face is the same as the mean face from the normal model in the previous chapter and the factors represent prominent directions of change. An advantage of the FA model over the PCA approach is that it can jointly model the data and does not need to have a separate model for the depth and intensity.

This model is then used to predict depth of an image using equation 6.5. Figures 6.2, 6.3, and 6.4 show the predicted 3D models. The FA model seems to be more sensitive to noise around the edges of the face then the MVG. A comparison between Figure 6.4 and Figure 5.6 suggests that visually the MVG performs better, but an explanation for this is that the FA model provides a more accurate prediction for the noisy data (areas which contain missing depth) where as the MVG's guess is so bad that it is not drawn. The dataset contains lots of observations which have missing data. It does not make sense to model this missing data so the next section introduces a more sophisticated FA model which deals with missing data points.

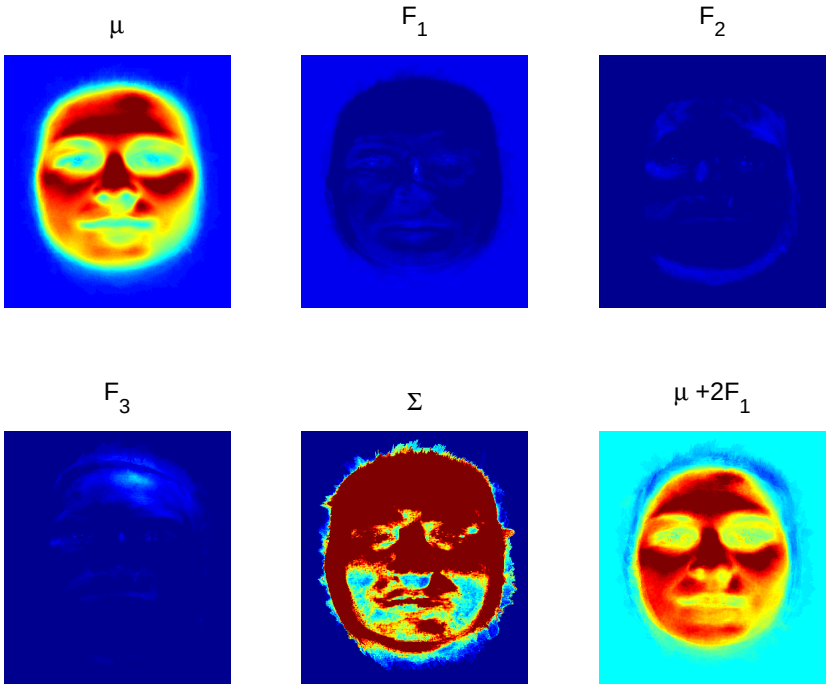

Figure 6.1: Factor Analysis model with three factors showing just the intensity images from the dataset learnt using the EM algorithm.

Figure 6.2: 3D model of subject's face predicted using a FA model. Noise can be seen on the top left of the face.

6.2 Factor Analysis with Missing Data

The data contains many missing data points, due to bad depth calculation and the different shapes of people's heads. It is not possible to statistically model this missing data, so a solution is to only use the observed data in

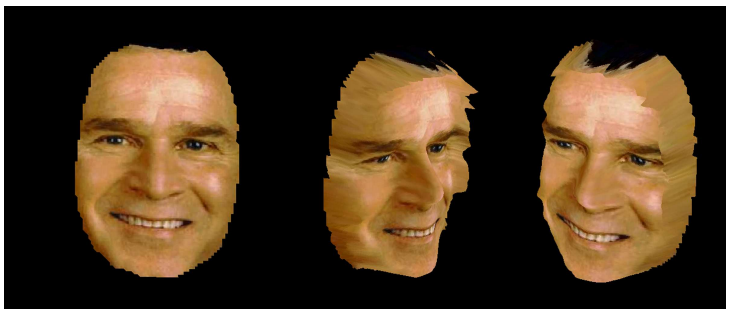

Figure 6.3: 3D model of George Bush's face predicted using a FA model.

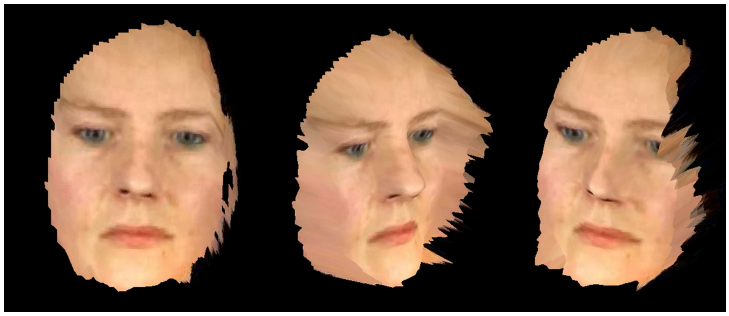

Figure 6.4: Predicted 3D model of synthesised photorealistic face predicted using a FA model. There is a lot of noise around the edges of the face.

the model. A binary label is associated with each data point in $\mathbf{x}$ specifying if the point is missing (a bad depth value at this point) or observed. Figure 6.5 shows the binary labels for two images from the dataset, where the red pixels correspond to missing data and the blue pixels represent observed data. It can be seen that the two

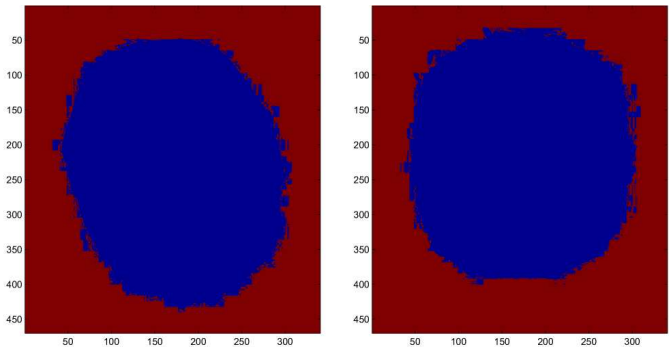

Figure 6.5: Two images from the dataset with red pixels corresponding to missing data and blue pixels representing observed data.

models are missing observations in different places around the face. If the sum at each data point for all the individuals is less than a specified amount we remove that observation from the dataset (i.e. we do not have

enough information to accurately model that data point). We remove all data points which are observed less than m times, where m is a parameter that can be varied.

We now need to reformulate the E and M step in the EM algorithm to use only the observed data. In the E step we want to optimise the free energy $\beta(q(\mathbf{h}), \theta^o)$ with respect to a distribution over the hidden variables $q(\mathbf{h})$ holding the parameters fixed:

$$q(\mathbf{h}) = Pr(\mathbf{h}|\mathbf{x}_n, \mathbf{F}^o, \mathbf{\Sigma}^o, \boldsymbol{\mu}^o) \tag{6.8}$$

where the superscript θ^o denotes that we are only learning the parameters from observed data. We want to fill in the hidden variables according to a posterior given the data. In the M step we want to optimise the lower bound with respect to the parameters θ holding the hidden distribution fixed:

$$\begin{aligned} logPr(\mathbf{x}_n, \mathbf{h}_n|\theta) &= \sum_n logPr(\mathbf{x}_n|\mathbf{h}_n, \theta) + logPr(\mathbf{h}_n) \\ &= \sum_n \sum_d 0.5\, log|\boldsymbol{\sigma}_d^{-1}| - 0.5(\mathbf{x}_n^d - \mathbf{f}^d\mathbf{h}_n)^T\boldsymbol{\sigma}_d^{-1}(\mathbf{x}_n^d - \mathbf{f}^d\mathbf{h}_n) - 0.5\mathbf{h}_n\mathbf{h}_n^T \end{aligned} \tag{6.9}$$

We take the derivative of this expression with respect to the vector $\mathbf{f}^d$ and set the result to zero to get:

$$\begin{aligned} \frac{d}{d\mathbf{f}^d} &\implies \sum_{n\epsilon n^o} \boldsymbol{\sigma}_d^{-1}(\mathbf{x}_n^d\mathbf{h}_n)(\mathbf{f}^d\mathbf{h}_n\mathbf{h}_n^T)^{-1} = 0 \\ \mathbf{f}^d &= \left(\sum_{n\epsilon n^o} \mathbf{x}_n^d\mathbf{h}_n^T\right)^T \left(\sum_{n\epsilon n^o} \mathbf{h}_n\mathbf{h}_n^T\right)^{-1} \end{aligned} \tag{6.10}$$

Taking the derivative with respect to the vector $\boldsymbol{\sigma}_d^{-1}$ gives:

$$\begin{aligned} \frac{d}{d\boldsymbol{\sigma}_d^{-1}} &\implies 0.5\sum_{n\epsilon n^o}\left(\boldsymbol{\sigma}^d + \mathbf{x}_n^d\mathbf{x}_n^{dT} - \mathbf{x}_n^d\mathbf{h}_n^T\mathbf{f}^{dT} + \mathbf{f}^d\mathbf{h}_n\mathbf{x}_n^{dT} + \mathbf{f}^d\mathbf{h}_n\mathbf{h}_n^T\mathbf{f}^{dT}\right) = 0 \\ \boldsymbol{\sigma}^d &= \frac{1}{N^o}\sum_{n\epsilon n^o} Diag\left[\mathbf{x}_n^d\mathbf{x}_n^{dT} - \mathbf{f}^d\mathbf{h}_n\mathbf{x}_n^{dT}\right] \end{aligned} \tag{6.11}$$

where N^o is the number of observations that have data points. We are only using the observed parts of the data (i.e. the rows of $\mathbf{F}$ and the values of $\mathbf{\Sigma}$ where we have associated data). These vectors are then concatenated to create the matrices $\mathbf{F}^o$ and $\mathbf{\Sigma}^o$. We do not know the value of $\mathbf{h}_n$ or $\mathbf{h}_n\mathbf{h}_n^T$ so we must take their expected values. These terms are the mean and second moment about zero for the distribution $Pr(\mathbf{h}_n|\mathbf{x}_n, \theta)$ and are taken during the M-Step:

$$E[\mathbf{h}_n] = (\mathbf{f}_n^T\boldsymbol{\sigma}_n^{-1}\mathbf{f}_n + \mathbf{I})^{-1}\mathbf{f}_n^T\boldsymbol{\sigma}_n^{-1}\mathbf{x}_n \tag{6.12}$$

$$E\left[\mathbf{h}_n\mathbf{h}_n^T\right] = (\mathbf{f}_n^T\boldsymbol{\sigma}_n^{-1}\mathbf{f}_n^T + \mathbf{I})^{-1} + E[\mathbf{h}_n]\,E[\mathbf{h}_n]^T \tag{6.13}$$

where $\mathbf{f}_n$, $\boldsymbol{\sigma}_n$ and $\mathbf{x}_n$ only contain terms where there is data observed. This leaves us with the following equations

for updating the parameters during the M-Step:

$$\begin{aligned}
\boldsymbol{\mu} &= \frac{1}{N^o}\sum_{n \epsilon n^o} \mathbf{x}_n \\
\mathbf{f}^d &= \left(\sum_{n \epsilon n^o} \mathbf{x}_n^d E[\mathbf{h}_n]^T\right)\left(\sum_{n \epsilon n^o} E\left[\mathbf{h}_n \mathbf{h}_n^T\right]\right)^{-1} \\
\boldsymbol{\sigma}^d &= \frac{1}{N^o}\sum_{n \epsilon n^o} Diag\left[\mathbf{x}_n^d \mathbf{x}_n^{dT} - \mathbf{f}^d E[\mathbf{h}_n]\,\mathbf{x}_n^{dT}\right]
\end{aligned} \tag{6.14}$$

6.2.1 FA with Missing Data Results

The same three experiments that were run using the FA and MVG models for depth prediction are conducted using the FA model for missing data. Figure 6.6 shows the resulting mean face and factors learnt using the EM algorithm. In comparison to Figure 6.1 there is much less noise around the edge of the face. The three factors seem to have a very small contribution with the mean face contributing most of the prediction. The predicted

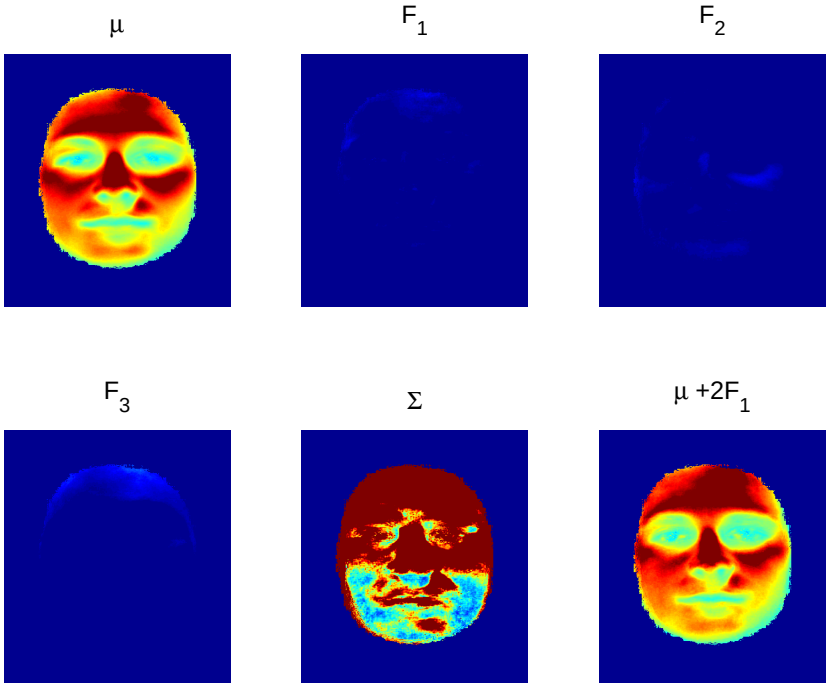

Figure 6.6: Factor Analysis with missing data model with three factors, showing just the intensity images from the dataset learnt using the EM algorithm.

models in Figures 6.7, 6.8 and 6.9 are much more realistic looking than the standard FA results in the previous

section. There is a huge reduction in the amount of noisy data at the edge of the faces. The remaining noise could be explained by the fact that the texture image that we wish to predict depth for may be larger than the mean face. As the model has no data for this point it cannot make a sensible prediction.

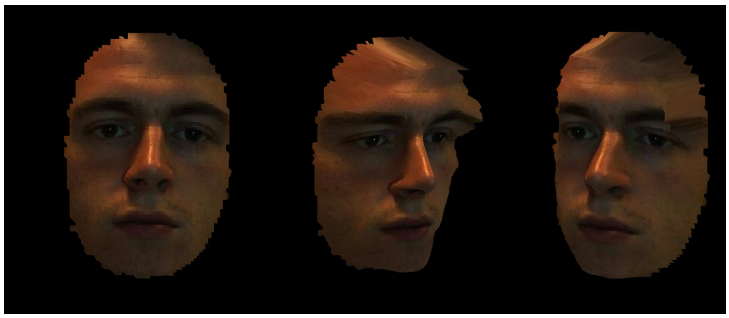

Figure 6.7: 3D model of face with depth predicted using FA with missing data. There still some noisy artefacts at the top and to the left of the model but this is a improvement over the FA model.

Figure 6.8: 3D model of George Bush's face with depth predicted using FA with missing data.

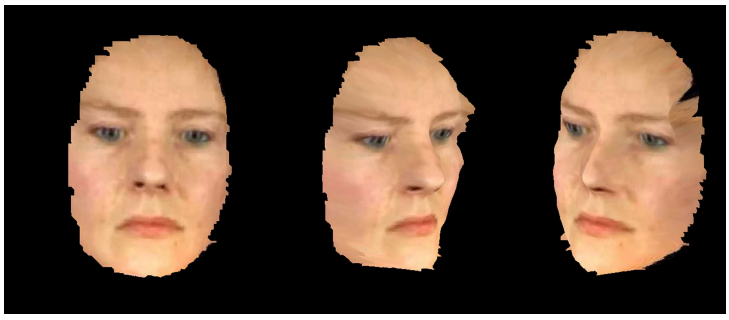

Figure 6.9: 3D model of a synthesised photorealistic face with depth predicted using FA with missing data.

6.3 Summary

Factor analysis is a fully probabilistic generative model and offers many advantages over the previous MVG approach. This chapter has outlined a model for the prediction of the depth of an individual's face based on FA. Results were provided for standard factor analysis and for a new type of factor analysis which takes missing data into account. It does not make sense to try and statistically model missing data and on visual inspection the FA model for missing data generates better depth predictions. An advantage of the FA model as compared to the MVG is that it is a generative method, so we could predict intensity values in places where they are missing in the input image.

Chapter 7

3D Face Verification

This chapter takes the captured dataset and performs 3D face verification. When given two faces, the purpose of face verification is to tell if they came from the same individual or not. The model used here is a latent identity variable subspace model. The main contribution of this work is to use the current model to perform 3D face verification. The results for 3D face verification show increased performance when compared to verification using only the 2D pixel intensity information and verification using only the depth information for the same individuals.

7.1 Latent Identity Variable Subspace Model

Most current face recognition methods are based on distance based approaches (see the literature review in chapter 2), whereby low dimensional feature vectors are extracted from the data and the distance between these features in feature space is used to identify images of the same person. The approach taken here is based on the latent identity variable subspace model from Prince et al. [40]. At the core of this approach is the notion that each individual's identity is represented by a multidimensional variable $\mathbf{h}$, termed the latent identity variable (LIV). This LIV resides in identity space. If two LIVs have the same value then they represent the same individual and if they are different they represent different individuals. Figure 7.1 illustrates the LIV approach. The main property of identity variables is that they remain the same for individuals regardless of changes in pose or illumination. Identity variables can never be observed directly but we can consider the observed faces as being generated by a noisy process. Here we use a factor analysis model which describes the observed face data $\mathbf{x}_{ij}$ in the form:

$$\mathbf{x}_{ij} = \boldsymbol{\mu} + \mathbf{F}\mathbf{h}_i + \varepsilon_{ij} \tag{7.1}$$

where $\mathbf{x}_{ij}$ is the vectorised data (intensity and depth information in this case) from the j'th 3D model of the i'th person, $\boldsymbol{\mu}$ is the mean of the data, $\mathbf{h}_i$ is the LIV and is the same for every image of that person and ε_{ij} is a stochastic noise term with diagonal covariance $\boldsymbol{\Sigma}$. The term $\mathbf{F}$ is the matrix of factors and together $\mathbf{F}\mathbf{h}_i$ account for between individual variance. We can express this model in terms of conditional probabilities:

$$Pr(\mathbf{x}_{ij}|\mathbf{h}_i) = \boldsymbol{\mathcal{G}}_{x_{ij}}\left[\boldsymbol{\mu} + \mathbf{F}\mathbf{h}_i, \boldsymbol{\Sigma}\right] \tag{7.2}$$

$$Pr(\mathbf{h}_i) = \boldsymbol{\mathcal{G}}_{h_i}\left[0, \mathbf{I}\right] \tag{7.3}$$

By integrating out the hidden variable $\mathbf{h}_i$ we get:

$$Pr(\mathbf{x}_{ij}) = \boldsymbol{\mathcal{G}}_{x_{ij}}\left[\boldsymbol{\mu}, \mathbf{F}\mathbf{F}^T + \boldsymbol{\Sigma}\right] \tag{7.4}$$

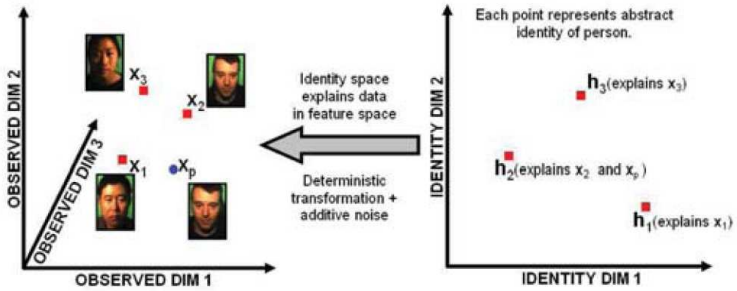

Figure 7.1: From Prince et al. [40]. Observed face data vectors $\mathbf{x}$ in the left image are generated from the underlying identity space $\mathbf{h}$ on the right. The faces $\mathbf{x}_2$ and $\mathbf{x}_p$ are from the same person as they are generated from the same identity variable $\mathbf{h}_2$.

7.1.1 Learning the Model

The EM Algorithm is used to learn the parameters $\theta = [\mathbf{F}, \mathbf{\Sigma}]$ of the model. In the E step we fix the parameters and calculate a posterior distribution for the LIV $\mathbf{h}_i$ given the data $\mathbf{x}_i$:

$$\begin{aligned} Pr(\mathbf{h}_i|\mathbf{x}_{i1...ij}, \theta) &= Pr(\mathbf{x}_{i1...ij}|\mathbf{h}_i, \theta)Pr(\mathbf{h}_i) \\ &= \prod_{j=1}^{J} \boldsymbol{\mathcal{G}}_{x_{ij}}\left[\mathbf{F}\mathbf{h}_i, \mathbf{\Sigma}\right] \boldsymbol{\mathcal{G}}_{h_i}\left[0, \mathbf{I}\right] \end{aligned} \tag{7.5}$$

In this formulation all the face model data for person i contributes to the estimate of the LIV $\mathbf{h}_i$. The update equations for the M step are based on the standard update equations seen in Equation 6.14:

$$\begin{aligned} \mathbf{F} &= \left(\sum_{i,j} \mathbf{x}_{ij} E\left[\mathbf{h}_i\right]^T\right)\left(\sum_{i,j} E\left[\mathbf{h}_i\mathbf{h}_i^T\right]\right)^{-1} \\ \mathbf{\Sigma} &= \frac{1}{IJ}\sum_{i,j} \mathbf{diag}\left[(\mathbf{x}_{ij}\mathbf{x}_{ij}^T - \mathbf{F}E\left[\mathbf{h}_i\right])\mathbf{x}_{ij}^T\right] \end{aligned} \tag{7.6}$$

where the **diag** function retains only the diagonal entries of a matrix. For J face models of each individual the expected values for the LIV are given by:

$$\begin{aligned} E\left[\mathbf{h}_i\right] &= (J\mathbf{F}^T\mathbf{\Sigma}^{-1}\mathbf{F} + \mathbf{I})^{-1}\mathbf{F}^T\mathbf{\Sigma}^{-1}\sum_{j=1}^{J}\mathbf{x}_{ij} \\ E\left[\mathbf{h}_i\mathbf{h}_i^T\right] &= (J\mathbf{F}^T\mathbf{\Sigma}^{-1}\mathbf{F}^T + \mathbf{I})^{-1} + E\left[\mathbf{h}_i\right]E\left[\mathbf{h}_i\right]^T \end{aligned} \tag{7.7}$$

7.1.2 Evaluating the Results

To perform verification on two 3D face models ($\mathbf{x}_1$ and $\mathbf{x}_2$) we calculate the likelihoods that they were created from the same or different identities. The approach is to create two different forms of the generative model for factor analysis; one for the models belonging to the same identity and the other when they do not. The

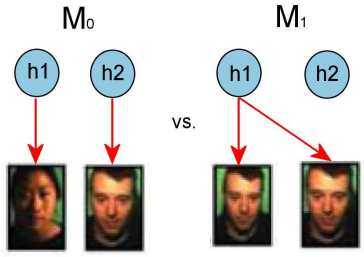

Figure 7.2: Face verification. In model M_0 the two faces do not match but in M_1 they do.

probability of the two different formulations is then evaluated. Figure 7.2 shows the two different cases. If the two 3D face models belong to the same identity, we can write the following joint generative equation:

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix} + \begin{bmatrix} \mathbf{F} \\ \mathbf{F} \end{bmatrix} \mathbf{h} + \begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \end{bmatrix} \tag{7.8}$$

In this case both 3D face models share the same hidden identity variable. If the two 3D face models are from different individuals the following generative model explains their relationship:

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix} + \begin{bmatrix} \mathbf{F} & 0 \\ 0 & \mathbf{F} \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \end{bmatrix} \tag{7.9}$$

or giving names to these composite matrices:

$$\mathbf{x}' = \boldsymbol{\mu}' + \mathbf{F}'\mathbf{h}' + \boldsymbol{\varepsilon}' \tag{7.10}$$

We can rewrite this compound model in terms of probabilities to give:

$$Pr(\mathbf{x}') = \mathcal{G}_{x'}\left[\boldsymbol{\mu}', \mathbf{F}'\mathbf{F}'^{T} + \boldsymbol{\Sigma}'\right] \tag{7.11}$$

where $\boldsymbol{\Sigma}'$ is the composite diagonal covariance matrix for the two 3D models. We evaluate the probability for both types of generating model and the result is the model that returns the largest probability.

If the dimensionality of the data is very large it is very computationally expensive to calculate the determinate of $(\mathbf{F}\mathbf{F}^T + \boldsymbol{\Sigma})$, a solution to this problem is to use the matrix determinate lemma:

$$|\mathbf{F}\mathbf{F}^T + \boldsymbol{\Sigma}| = |\mathbf{I} + \mathbf{F}^T\boldsymbol{\Sigma}^{-1}\mathbf{F}|\,|\boldsymbol{\Sigma}| \tag{7.12}$$

where the determinate of the diagonal matrix $\boldsymbol{\Sigma}$ is the product of its diagonal entries. To evaluate the $Pr(\mathbf{x}')$ we can take the logarithm of the rearranged standard multivariate Gaussian from Equation 7.11:

$$log(Pr(\mathbf{x}')) = -d0.5log(2\pi) - 0.5\left(log(|\mathbf{I} + \mathbf{F}'^T\boldsymbol{\Sigma}'^{-1}\mathbf{F}'|) + log(|\boldsymbol{\Sigma}'|)\right) - 0.5\left[(\mathbf{x}' - \boldsymbol{\mu}')^T\boldsymbol{\Sigma}'^{-1}(\mathbf{x}' - \boldsymbol{\mu}')\right] \tag{7.13}$$

the matrix inversion lemma from Equation 6.6 is also used to reach this formulation.

7.2 Results

Two 3D models from each of 60 individuals in the dataset are used to perform verification (giving a total of 120 3D models). The first 50 individuals are used for the training set and the last ten are used for the test

set. Ten iterations of the EM algorithm is used to learn the parameters of the model. To improve computation speed the dimensionality of the data is reduced by only using every second data point for each individual, also the data is preprocessed to remove data points that have less than m observations across the dataset. For the following experiments the value of m is set to be 30 (i.e. a data point must be present in at least 30 out of 100 individuals to be considered not missing).

Three types of verification experiment are conducted using the LIV subspace model. The first is verification using only the pixel intensity information, the second experiment uses just the depth information and the last uses both the intensity and depth information for full 3D verification. The results are presented on a receiver operating characteristic (ROC) curves, which plots the fraction of true positives versus the fraction of false positives. The best possible prediction method would yield a point in the upper left corner of the ROC space.

7.2.1 2D Verification

For 2D face verification the model is trained only using the pixel intensity information for each individual. The images are normalised and grey scaled (as outlined in chapter 4) before being used. The middle graph in Figure 7.3 shows the ROC curve for 2D face verification. It can be seen from the curve that there is little performance

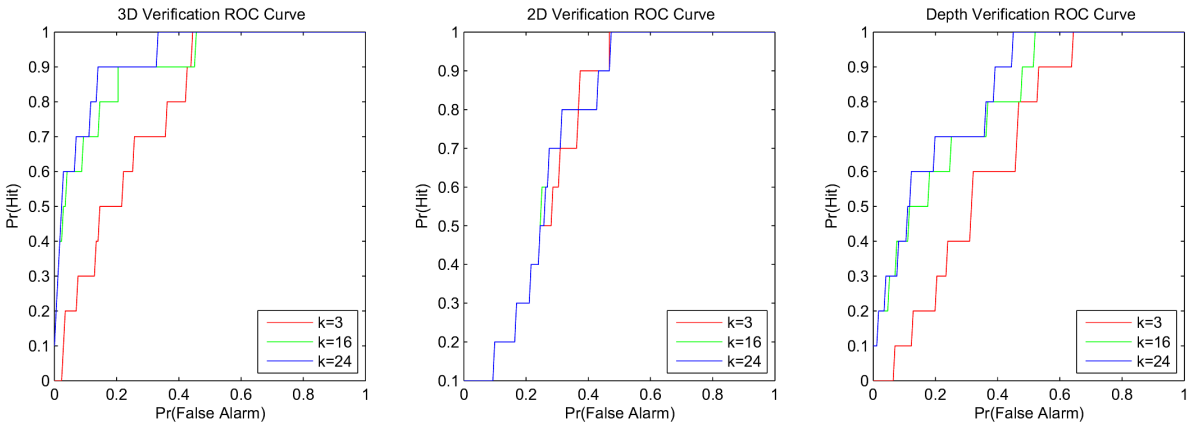

Figure 7.3: Each of the three ROC curves are plotted for three different sizes of subspaces. (Left) ROC curve for 3D verification. (Middle) ROC curve for 2D verification. (Right) ROC curve for depth based verification.

improvement achieved by increasing the number of factors k. The 2D face versification performs the worst of all the methods. One potential reason for this could be due to poor image registration. As a result of the 3D alignment method used the images are sometimes not very well aligned in 2D. A solution to this would be to use a more sophisticated 3D alignment algorithm such as ICP [2]. Also the variations in lighting conditions might not be correctly compensated for during the illumination normalisation stage.

7.2.2 Depth Based Verification

For depth based verification, only the depth information for each individual is used. One advantage to this approach is that the data, if captured correctly, should be invariant to changes in lighting. Like 2D recognition the model is very sensitive to changes in facial expression but for the purpose of these experiments all the subjects have a neutral facial expression. The last ROC curve in Figure 7.3 shows the results for depth based verification. The results are much better than for the standard 2D face verification in the previous experiment.

7.2.3 3D Verification

The final verification experiment uses both the pixel intensity and depth information for each individual to perform 3D verification. It can be seen from the first ROC curve in Figure 7.3 that the 3D method outperforms both the previous two methods. One simple explanation for this is that there is more data for each individual. The results are promising and verify that 3D face recognition has the potential to achieved better results than traditional 2D approach (see Figure 7.4 for a summary of the results).

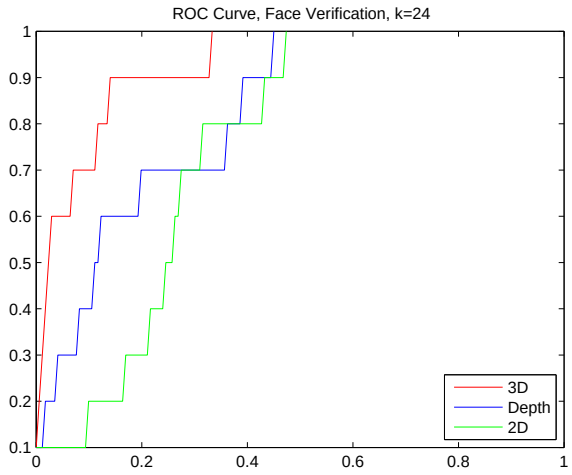

Figure 7.4: ROC curve comparing the performance of 3D, 2D and depth face verification with 24 factors.

One way to to improve performance would be to run more experiments with differing number of factors. Also the parameter m which thresholds the missing data could be varied to see if it has an effect on verification performance.

7.3 Summary

This chapter has outlined a LIV subspace model and performed three different face verification experiments. The advantage of this probabilistic factor analysis approach is that it is invariant to the scale difference between the intensity and depth information for each individual. This is in contrast to traditional distance based approaches which are unable to cope with scale differences without preprocessing the data. This allows each face (intensity and depth) to be modelled as one data vector. The results of the experiments on this dataset show that 3D verification using depth and intensity information performs better than verification using only depth or intensity.

Chapter 8

Discussion and Further Work

This final chapter provides a discussion and criticism of the previous work. It also suggests avenues for further work which were not possible to investigate due to time constraints.

Improvements

This section provides a criticism of the work. It points out problems with the methods used and suggests improvements.

Capture

One improvement that could be done, is the development of a method for automatically rejecting fully or even partially badly reconstructed models. This would allow the system to disregard any models which are not correct and therefore reduce the amount of noise in the dataset. One possible way to achieve this would be to reject images where a certain proportion of the phase could not be unwrapped in the centre of the image.

Currently, to interpolate areas of missing depth a Gaussian is blurred over the region. A better method would be to use a Poisson equation. This is the approach used in image inpainting.

Prediction

A major problem with the prediction phase is the use of badly aligned models. The current approach marks six key points on each face and finds the best rotation and translation to match the model to a template. To improve alignment, the current method could be used for an initial coarse alignment and then the ICP algorithm used to provide a final fine alignment.

The marking of key points and background subtraction by hand is a time consuming process and one which could be easily automated. There are many algorithms which automatically find landmark points on faces. In this case the task is very straightforward as all the images are uniform frontal faces.

When normalising the greyscaled images for intensity, the background values skew the histograms. A more sensible approach would be to first remove the background information from the data vector and then normalise the images. Also it could be possible to build a mixture model (e.g. a mixture of factor analysers) by modelling each colour channel independently.

Like the MVG model, the FA model also assumes the data is jointly normal. A solution to this would be to use a mixture of factor analysers. A more quantitative method is needed to judge if the predicted depth for an image is realistic. The current qualitative visual assessment is not strong enough.

Verification

The experiments reported in the verification chapter are run on a relatively small dataset. With 100 training images from 50 different individuals and 20 test images form ten different individuals. For more conclusive results more data is needed. Also, the results reported are only for frontal faces so it not fair to say that 3D verification outperforms 2D verification in all cases. More experimental results are needed.

Further Work

This report describes the individual components of a system that could be used to capture and verify an individual's identity for access control. By combining all these components together a real time system could be easily engineered.

Further work on verification could include the adaptation of the FA model for missing data to work in the LIV subspace model. This may result in better recognition performance. Experiments could be run to measure if the use of the predicted depth of an image results in higher recognition performance.

More investigation is needed to see how well the methods for predicting depth generalise. Using of a different dataset it may be possible to reconstruct 3D models of various different types of objects. There are many applications of such technology. For example, it may be possible in medical imaging to learn the 3D geometry of a tumour from a single image.

Summary

From this report we can see that 3D face verification offers increased verification performance over 2D approaches. The capture method described shows that it is possible to create a reliable, near real time system for reconstructing 3D models. All the steps outlined could be automated for a fully automatic 3D face recognition system. Also we have seen that it is possible to generate photo realistic 3D models of human faces from 2D intensity images. The techniques used could be applied to any other type of data for automatic 3D model creation.

Bibliography

[1] T. Akimoto and R. Wallace, "Automatic Creation of 3D Facial Models," *IEEE Computer Graphics and Applications*, 1993.

[2] P. Besl and N. McKay, "A Method for Registration of 3-D Shapes," *IEEE Trans PAMI*, Vol. 14, no. 2, pp. 239-256, 1992.

[3] C.M. Bishop, *Pattern Recognition and Machine Learning,* Springer, 2006.

[4] D. Bitouk and S.K. Nayar, "Creating a Speech Enabled Avatar from a Single Photograph," In Proc. *IEEE Virtual Reality Conference*, pp 107-110, 2008.

[5] V. Blanz, "Face Recognition based on a 3D Morphable Model," In Proc. *7th Int. Conference of Automatic Face and Gesture Recognition*, pp. 617-622, 2006.

[6] A.M. Bronstein, M.M. Bronstein, and R. Kimmel, "Three-dimensional face recognition," *International Journal of Computer Vision*, Vol. 64, pp. 5-30, 2005.

[7] "Michael M. Bronstein 3D Face Website", [Online]. Available: http://www.cs.technion.ac.il/ mbron/research_facerec.html. [Accessed: Aug. 15, 2008]

[8] C. Hernandez, G. Vogiatzis, G.J. Brostow, B. Stenger, R. Cipolla, "Non-rigid Photometric Stereo with Colored Lights" In Proc. *ICCV 2007*, pp. 1-8, 2007.

[9] "National Park Service Digital Image Archives", [Online]. Available: http://photo.itc.nps.gov/storage/images/officials/bush.jpg. [Accessed: Aug. 15, 2008]

[10] B. Carrihill and R. Hummel, "Experiments with the intensity radio depth sensor," *Computer Vision, Graphics and Image Processing*, pp. 337-358, 1985.

[11] J. Y. Cartoux, J. T. Lapreste and M. Richetin, "Face authentification or recognition by profile extraction from range images," *In Workshop on Interpretation of 3D Scenes*, pp. 194-199, 1989.

[12] A.R. Chowdhury, R. Chellappa, S. Krishnamurthy and T. Vo, "3D face reconstruction from video using a generic model," In Proc. *ICME*, Vol.1, pp. 449-452, 2002.

[13] M.R. Everingham and A. Zisserman, "Regression and classification approaches to eye localization in face images," In Proc. *7th International Conference on Automatic Face and Gesture Recognition*, pp. 441-446, 2006.

[14] "The Face Recognition Grand Challenge", [Online]. Available: http://www.frvt.org/FRGC. [Accessed: Aug. 15, 2008]

[15] "Face recognition Vendor Test 2006", [Online]. Available: http://www.frvt.org/FRVT2006/docs/FRVT2006andICE2006LargeScaleReport.pdf. [Accessed: Aug. 15, 2008]

[16] "The Facial Recognition Technology (FERET) Database", [Online]. Available: http://www.itl.nist.gov/iad/humanid/feret/feret_master.html.[Accessed: Aug. 15, 2008]

[17] T. Faltemier, K.W. Bowyer and P.J. Flynn, "A Region Ensemble for 3D Face Recognition," *IEEE Transactions on Information Forensics and Security*, pp. 62-73, 2008.

[18] N.I. Fisher, T. Lewis and B.J.J Embletion, "Statistical analysis of spherical data," *The Cambridge University Press*, 1987.

[19] N. Friedman and S. Russel, "Image segmentation in video sequences," In Proc. *Uncertainty in Artificial Intelligence*, pp. 175-181, 1997.

[20] D.C. Ghiglia and M.D. Pritt, *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software,* John Wiley & Sons, 1998.

[21] G.G. Gordon, "Face recognition based on depth and curvature features," In proc. *ICVPR 1992*, pp. 808-810, 1992.

[22] C.J. Harris and M. Stephens, "A combined corner and edge detector," In Proc. *4th Alvey Vision Conference*, pp. 147-151, 1988.

[23] M. Harville, G. Gordon and J. Woodfill, "Foreground segmentation using adaptive mixture models in colour and depth," In Proc. *IEEE Workshop on Detection and Recogntion of Events in in Video*, pp. 3-11, 2001.

[24] "Jerry's Taurus Studio - Photometric Stereo", [Online]. Available: http://www.taurusstudio.net/research/photex/ps /index.htm. [Accessed: Sept. 03, 2008]

[25] D. Jiang, Y. Hu, S. Yan, L. Zhang, H. Zhang and Wen Gao, "Efficient 3D reconstruction for face recognition," *Pattern Recognition*, Vol. 38, pp. 787-798, 2005.

[26] K.N. Kutulakos and S.M. Seitz, "A Theory of Shape by Space Carving," *International Journal of Computer Vision 38*, pp. 199-218, 2000.

[27] "Konica-Minolta USA. minolta Vivid 910 no-contact 3D laser scanner", [Online]. Available: http://www.minolta3d.com/products/vi910-en.asp. [Accessed: Aug. 15, 2008]

[28] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 16(2), pp. 150-162, 1994.

[29] K. Levenberg, "A Method for the Solution of Certain Non-Linear Problems in Least Squares," *The Quarterly of Applied Mathematics 2*, pp. 164168, 1944.

[30] Y. Lee, K. Park, J. Shim and T. Yi, "3D face recognition using statistical multiple features for the local depth information," In Proc. *16th International Conference on Vision Interface*, 2003.

[31] X. Lu, D. Colbry and A.K. Jain, "Three-Dimensional Model Based Face Recognition," In Proc. *ICPR 04*, Vol. 1, pp. 362-366, 2004.

[32] Wan-Chun Ma et al., "Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination," *Eurographics Symposium on Rendering 2007*, 2007.

[33] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan, "Image-based visual hulls," In Proc. *SIGGRAPH 2000*, pp. 369-374, 2000.

[34] T. Papatheodorou and D. Rueckert, "Evaluation of 3D face recognition using registration and PCA," In Proc. *AVBPA 2005*, pp. 9971009, 2005.

[35] U. Mohammed, S.J.D Prince and J. Kautz, "Stochastic generation of realistic image content," presented at 4th European Conference on Visual Media Production, London, UK, 2007.

[36] T. Nagamine, T. Uemura and I. Masuda, "3D facial image analysis for human identification," In Proc. *International Conference Pattern Recognition 1992*, pp. 324-327, 1992.

[37] J.L. Posdamer and M.D. Altschuler, "Surface measurement by space encoded projected beam systems," *Computer Graphics and Image Processing*, Vol. 18, pp. 1-17, 1982.

[38] S.J.D Prince, *Machine Vision Course Notes,* November, 2007.

[39] S.J.D Prince and J.H. Elder, "Three-Dimensional face reconstruction using near infra-red light," Technical Paper, York University Toronto.

[40] S.J.D. Prince, J.H. Elder, J. Warrell and F. Felisberti,"Tied factor analysis for face recognition across large pose differences," *IEEE Pattern Analysis and Machine Intelligence*, Vol. 6, 2008.

[41] J. Tajima and M. Iwakawa, "3D data acquisition by rainbow range finder," In Proc. *International Conference on Pattern Recognition*, 1990.

[42] A. Tal, M. Elad and S. Ar, "Content based retrieval of VRML objects - and iterative and interactive approach," In Proc. *Eurographics Workshop on Multimedia*, 2001.

[43] H.T. Tanaka, M. Ikeda and H. Chiaki, "Curvature-based face surface recognition using spherical correlation. Principal directions for curved object recognition," In Proc. *Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 372-377, 1998.

[44] L. Tang and T.S. Huang, "Automatic construction of 3D human face models based on 2D images," In Proc. *International Conference on Image Processing*, Vol. 3, pp. 467-470, 1996.

[45] M. Turk and A. Pentland,"Face recognition using eigenfaces," In Proc. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586591, 1991.

[46] T. Vetter and V. Blanz, "Estimating Coloured 3D Face Models from Single Images: An Example Based Approach" In Proc. *ECCV'98*, pp. 499-513, 1998.

[47] Y. Wang, C. Chua and Y. Ho, "Facial feature detection and face recognition from 2D and 3D images," *Pattern Recognition Letters*, Vol. 23, pp. 1191-1202, 2002.

[48] R.J. Woodham, "Photometric Method for Determining Surface Orientation from Multiple Images," *Optical Engineering*, Vol. 19 No. 1, pp. 139-144, 1980.

[49] M. Woo, J. Neider and T. Davis, *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.1,* Addison-Wesley, 1997.

[50] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," In Proc. *ICCV 1999*, pp. 666-673, 1999.

Appendix A

Accompanying DVD

The accompanying DVD is broken into three sections; the source code for the project, demos showing the work and the full collected dataset of 3D models. The following appendix lists the contents of the DVD. It is possible for anyone to use the code and data to recreate any of the experiments contained in the body of the report.

A.1 Code

The code section of the DVD contains all the commented source code which is used in the project. The capture code is written in C and the prediction and face verification code is in MATLAB.

A.1.1 C Code

The *C* folder contains all the source code used for calibrating the system, capturing a model, aligning the models and displaying a model to the screen. The *CaptureBuildModelExample* project contains the main capture code. The user must set up the monitor as the secondary display device and the projector as the secondary one. This is because the Glut library will only allow full screen windows on the primary display. When the programme is executed the user must place their head within the circle displayed on the monitor and the model is captured when the operator presses the *space* key. The operator can capture multiple models by repeatedly pressing the *space* key and capture can be terminated by pressing the *esc* key.

The *CaptureCalibationData* folder contains the code to capture the calibration parameters of the camera and projector. The programme collects multiple images of the checkerboard calibration pattern until the user presses the *esc* key. There is also MATLAB code provided to generate the calibration patterns as well as the projected structured light pattern.

The *normaliseImages* programme performs the 3D model alignment. It takes a texture image with associated depth map and rotation matrix as an input and rotates the model. It then takes a screen shot of the newly rotated model and saves the new texture and depth map to disk. There is a batch file provided which allows the alignment of multiple models with just one click.

A.1.2 MATLAB Code

All the necessary MATLAB code is provided to predict depth, perform verification and reprocess the data for alignment. The *faceRecognition* folder contains code for the three different face verification experiments; depth, 2D and 3D verification. It also contains the original data so the user can rerun the experiments to verify the results. Code is provided to create the ROC curves shown in the body of the report.

The *depthPrediction* folder contains the code for the three different types of depth prediction methods; multivariate Gaussian, factor analysis and factor analysis with missing data. Code is included which would allow the user to take any 2D face image, mark the keypoints on the face, warp it to a template image and then predict its depth. The depth file can then be used to display the face model with the 3D viewing application.

Finally the *preProcess* folder contains the code used to mark and store feature points on the faces, output the rotation and translation matrices to align the models with the template, to down sample the data for faster computation and code for background subtraction.

A.2 Demos

The demos section of the DVD contains the 3D models created during the prediction phase of the project. It allows the user to view the predicted standard, George Bush and photorealistic face generated from one of the following three methods: multivariate Gaussian, factor analysis or factor analysis with missing data. Each model can be viewed by running the associated *run.bat* file.

This section also contains the main 3D model viewing application created specifically for this project, *3DFace*. Any texture image and associated depth can be taken from the face dataset and drawn in 3D using OpenGL. By placing the files int the *files* folder and editing the *run.bat* file to point to the files any model can be drawn. The application allows the user to rotate the model in 3D space and when there is no input from the user the model rotates from left to right. The user also has the option to toggle between the texture mapped or wire frame model by pressing the w key.

A.3 Face Dataset

The Face Dataset section contains two folders: *DifferentPoses* and *FrontalFaces*. The *DifferentPoses* contains the texture and structure image, depth map and VRML model for each of the five different head positions for each of the 64 individuals in the dataset. Each individual is placed in a separate folder indexed from one to 64. It also contains a text file called *calibrationConstants.txt* which contains all the camera and projector calibration parameters used when collecting the data.

The *FrontalFaces* folder contains two different frontal texture images and depth map for each of the 64 different individuals in the dataset. There is a folder called *features* which contains the location of each of the six feature points hand marked on each individuals face in MATLAB *.mat* format. It also contains a folder called *mat* which specifies the rotation, translation and scaling needed to align each model to the template model (the first image from the first individual in the dataset). Also the low resolution (470×340) aligned image for each frontal face with the background removed is provided in the folder called *small*. This is the data which is used during the inference and recognition stages of the project.