Human Face Verification by Robust 3D Surface Alignment

By

Dirk Joel Luchini Colbry

A Dissertation

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

PhD

Department of Computer Science

2006

Abstract

Human Face Verification by Robust 3D Surface Alignment

By

Dirk Joel Luchini Colbry

Traditional 2D face recognition systems are not tolerant to changes in pose, lighting and expression. This dissertation explores the use of 3D data to improve face recognition by accounting for these variations. A two step, fully automatic, 3D surface alignment algorithm is developed to correlate the surfaces of two 3D face scans. In the first step, key anchor points such as the tip of the nose are used to coarsely align two face scans. In the second step, the Iterative Closest Point (ICP) algorithm is used to finely align the scans. The quality of the face alignment is studied in depth using a Surface Alignment Measure (SAM). The SAM is the root mean squared error over all the control points used in the ICP algorithm, after trimming to account for noise in the data. This alignment algorithm is fast (<2 seconds on a 3.2GHz P4) and robust to noise in the data (<10% spike noise). Extensive experiments were conducted to show that the alignment algorithm can tolerate up to 15° of variation in pose due to roll and pitch, and 30° of variation in yaw. It is shown that this level of pose tolerance easily covers the normal pose variation of a database of over 275 cooperative subjects. By using the SAM as an initial matching score and automatically rejecting poor quality scans, an equal error rate of 1.2% is achieved on a database of 943 scans from 275 subjects. This surface alignment verification system is fast, fully automatic, and requires no additional training. By using the 3D face surface alignment algorithm, a Canonical Face Depth Map (CFDM) is defined to allow for automatic preprocessing of 3D face scans. The CFDM is shown to help in anchor point localization, reduce model memory requirements, and enables other algorithms, such as PCA or correlation, to be applied.

To Kate, my wife and my best friend.

Acknowledgments

- Thanks to George Stockman for always finding time to provide advice, encouragement and support as I navigated the Ph.D.
- Thanks to Anil Jain for inspiring and funding my initial research in 3D faces, and for his guidance throughout the dissertation process.
- Thanks to Frank Biocca and Hayder Radha for providing feedback and perspective on the dissertation as members of my committee.
- Thanks to all of the members of the PRIP lab for participating in my research projects and for your friendship and support.
- Thanks to the students who helped with programming and data collection, including Brian Hasselbeck, Rayshawn Holbrook, Greg Heil, Charles Otto, Jason Payne, Jermil Sadler, and Thaddeus Walker.
- Thanks to the NSF and the MSU IGERT program for funding my research, and to my fellow IGERT students for their friendship and support.
- Thanks to my extended family in East Lansing - Karl, Angie, Sienna, Krissy, Steve, Teresa and all of the folks at St. John - for the dinners, games and loving friendship over the years.
- Thanks to my wife, Katy, who worked as hard on this dissertation as I did, and who will deliver our first child about the same time I receive this degree.
- Thanks to my entire family for their love and support, especially Alberta, Norm, Tricia, Michael, Alonzo, Zachary, John, Tamara, Katy (and Baby), Mark, Dolly, Nan, Tim, and Frank.

Table of Contents

List of Tables

List of Figures

Images in this dissertation are presented in color.

Chapter 1

Introduction to the Problem

This dissertation describes methods for modeling and matching the human face in three dimensions, and explores how such 3D models can improve the performance of face processing algorithms that have traditionally focused on 2D texture images.

Traditional 2D face recognition systems are used to compare two images and determine whether the faces in the images are from the same person. Such 2D face recognition systems have been used successfully in limited applications such as security access [77], computer login/logout programs [43, 7] and search engines [111]. However, traditional 2D face recognition systems rely on texture (color) images that are not tolerant to changes in pose, lighting or expression (see Figure 1.1).

Figure 1.1: Reasonable variations in face images (lighting, pose, facial hair, hairstyle).

Adding 3D information can improve the performance of face recognition algo-

rithms by accommodating some variations in lighting, pose and expression. A 3D scanner measures the global coordinates on the surface of an object shown in the 2D image. These global coordinates can be used to easily rotate and manipulate the surface in 3D, while in a 2D image the relative locations must be estimated in order to change pose, and this estimation may cause errors [93]. Researchers have successfully used 3D face data to develop algorithms for communications applications [117] and face recognition [65, 2], yet there are a number of situations where 2D and 3D facial processing algorithms are not yet sufficiently robust.

This dissertation explores and improves upon the use of 3D data to make several contributions to face processing research, including:

- Designing and optimizing a fast, fully automatic method for accurately aligning faces from two different scans. This method is shown to be tolerant to changes in pose, lighting and expression.
- Experimental analysis and study of a practical face verification system for use with cooperative subjects.
- Developing a Canonical Face Depth Map (CFDM) to normalize 3D face data by creating a uniform, face-based coordinate system that allows different scans to be readily compared. The CFDM also performs well as a preprocessor for other algorithms.

The first contribution is the design of a fast, fully automatic face alignment algorithm. This automated alignment algorithm fills a gap in existing research, which is dominated by methods that use pre-aligned or manually aligned data. By developing methods for automatically aligning two face scans in real time, this dissertation research provides a foundation for building fully automatic face processing algorithms.

The second contribution is the experimental analysis and study of a practical face verification system. This system achieves $\sim$1% Equal Error Rate

with cooperative subjects and is tested thoroughly on datasets containing 943 scans (from 275 subjects) taken under different pose, lighting and expression conditions. Experiments show that this 3D face verification system is robust to many types of variations that would be expected in a real world application.

A third contribution is the development of the Canonical Face Depth Map (CFDM). The CFDM offers a standard format for all 3D face processing algorithms and can compress the raw 3D data 90% (down to $\approx$ 160 Kilobytes). The face-based coordinate system used by the CFDM can be applied to all types of face data (including 2D texture images and 3D scans) and is designed to be identifiable across different poses and many expressions. The format of the CFDM is easily transformed into a data structure similar to the output of a 3D scanner. This allows the CFDM to be used as a preprocessor to other matching algorithms, such as a 3D version of PCA and a 3D version of correlation.

Table 1.1 summarizes the major contributions of this dissertation and indicates the portions of this dissertation that describe the contributions in detail.

1.1 Background and Motivation for the Face Biometric

The goal of biometrics is to develop methods that can take bodily measurements from two people and return a similarity score indicating whether the data are from the same person or from different people. Normally a similarity score can be used for verification (one-to-one) or for recognition (one-to-many). Examples of human biometrics include fingerprints, iris scans, voice patterns, etc. Current interest in homeland security, such as security surveillance systems [47] and biometrics [94], has rejuvenated an interest in face biometrics. Faces have advantages over other biometric modalities. Some modalities are considered socially intrusive or uncomfortable; for example, some

Table 1.1: Outline of major thesis contributions.

Contribution	Description	Chapter
Anchor Point Detection	• Fully automatic, fast 3D frontal anchor point detection system. • Fully automatic 3D arbitrary pose anchor point detection system.	4
Surface Alignment	• Fully automatic, fast 3D frontal pose alignment system. • Fully automatic 3D arbitrary pose alignment system. • Frontal system tested to be robust to minor pose variations. • Frontal system tested to be robust to lighting variations. • Frontal system tested to be robust to changes in algorithm parameters. • Trimming component makes the alignment robust to noise in the data. • Control point region makes the alignment tolerant to minor expression changes.	5
Canonical Face Depth Map	• Well defined, face-based coordinate system definable under many different poses and inputs. • Utilizes robust surface alignment algorithm. • Reduces 3D model memory requirement by 90% (down to $\approx$ 160 Kilobytes). • Tested as a preprocessor to a standard PCA algorithm and demonstrated improved face recognition performance. • Used as the first step in a correlation algorithm. • Correlation algorithm can improve anchor point localization. • Correlation matching, along with SAM score, produces very successful face verification results.	6
Practical 3D Face Verification System	• Developed Surface Alignment Measure (SAM) as a measurement of how well surfaces are aligned. • Scanning system is optimized for speed ($<$ 2 seconds). • Algorithm is optimized for speed ($<$ 2 seconds). • Developed a symmetry reject option to eliminate bad scans.	7

people are reluctant to enroll in systems that utilize fingerprints because fingerprints are commonly associated with law enforcement [52]. Other systems, such as iris scanners, are highly accurate but physically intrusive - you must place your eye close to a scanner. In contrast, faces are one of the most commonly used (and readily accepted) biometrics in the world. The majority of identification documents (such as passports, drivers licenses, etc.) use face images for identification, and almost everyone uses faces as a primary source of identification in daily life. Humans are often able to accurately identify faces under different pose, expression or lighting conditions. For example, humans have no problem identifying people in distorted caricatures, which is a particularly hard task for computers. A cartoon artist finds distinguishing features and exaggerates them to an extreme (see Figure 1.2). This exaggeration allows people to more easily identify the subject of the cartoon, so it seems reasonable that humans are somehow encoding these differences as part of their identification metric. However, it is possible that people do not actually recognize each other as easily as is assumed. For example, most humans have difficulty recognizing or distinguishing between individuals who are not part of their familiar circle [114]. In other words, most humans find it easier to identify individuals with whom they are already familiar.

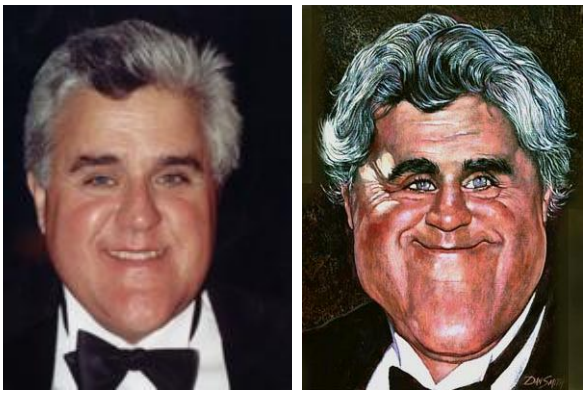


Figure 1.2: People do not have trouble recognizing exaggerated distinguishing features, such as Jay Leno's chin. (Caricature published with permission from [129].)

Another advantage of using faces is that everyone has a face, while not everyone has viable fingerprints (2% of subjects can not be easily fingerprinted using current technology [109]). Also, if a face-based automatic biometric system breaks down due to a power outage or other hazard, security personnel can always be used as a stop-gap replacement to confirm identity by comparing the subject to a photo ID.

Although face-based biometrics have advantages, individual human faces are very similar and are difficult to distinguish via automatic face recognition algorithms. While biometric systems based on fingerprints or iris scans can assume that each subject has a different set of salient minutia (true even for identical twins [80]), the overall structure of the face is largely the same for everyone. Although moles, birthmarks, piercing, tattoos, facial hair, eyeglasses, etc., may be considered distinguishing marks, there is no guarantee that a particular person will have one of these distinguishing marks or that the marks will not change over time. This small between-subject variability makes face recognition a particularly difficult problem in computer vision research.

The Face Recognition Vendor Test (FRVT) was designed to provide a common database for computer vision researchers to compare their face recognition algorithms. One of the findings of FRVT is that 2D face recognition systems can achieve good performance in constrained environments [119]. However, 2D approaches encounter difficulties in handling variations due to changes in head pose, lighting conditions and facial expressions [119]. Since the human face is a three-dimensional object whose 2D projection (image) is sensitive to changes in lighting, pose and expression, utilizing 3D data may improve face recognition performance. Table 1.2 summarizes how this dissertation utilizes 3D information to partially account for changes in pose, lighting and expression. The next section overviews the structure of this dissertation and outlines the key contributions presented in each chapter.

Table 1.2: Utilizing 3D information

	Description	Section
Pose	The 2D projection of a 3D object loses information, while a 3D scan of a 3D object maintains relative position information for each point in 3D space. Pose is corrected for in 3D without loss of information.	5.3.3.
Lighting	2D face images depend entirely on how light reflects off the surface of the face. Thus, changes in ambient lighting can significantly alter the 2D texture (color) image. 3D structured lighting methods project a light source, and only require that the light is seen by the camera to generate valid 3D data. 3D data are demonstrated to be robust to changes in ambient lighting.	5.3.1
Expression	Expression is an issue in both 2D and 3D. However, it is easier to register scans in 3D, which makes it easier to select regions of the face that do not change with expression.	5.2.1

1.2 Project and Document Overview

This dissertation begins with an introduction to 3D notation, detailed in Chapter 2. The discussion of notation includes a description of various coordinate systems and the different data structures used to represent 3D information. A face surface s can be represented by the relationship $f(p) : \mathbb{R}^3 \rightarrow \mathbb{R}$, where $p \in \mathbb{R}^3$ and $s = \{p|f(p) = 0\}$, together with a texture mapping $g(p) : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, where $g(x, y, z) = [R, G, B]$. A new 3D discrete data structure based on cylindrical coordinates is introduced, and Chapter 2 concludes with an overview of the face image databases collected for this dissertation.

Chapter 3 presents the mathematical approaches used in this dissertation to process 3D data. The algorithms presented in Chapter 3 include: curvature calculations, issues with resolution and scale, image cleaning algorithms, basic surface alignment algorithms, orthogonal depth maps, and 3D surface fitting. A fast, linear (with the number of rows in matrix) 2.5D ICP algorithm is also presented in Chapter 3.

Chapter 4 describes methods for detecting anchor points on faces. Anchor point

detection is a key requirement for generic face recognition, calculating the coarse transformation for registration, and transforming a test scan into the Canonical Face Depth Map format. A 99% success rate is achieved in finding anchor points in frontal images, and an 86% success rate is achieved in scans with variations in pose and expression. These results demonstrate the challenges in 3D face recognition under conditions of arbitrary pose and expression.

Chapter 5 discusses the two-step method used to register 3D surfaces. First, the anchor points found in Chapter 4 are used to coarsely align the face scans. Then, the Iterative Closest Point (ICP) algorithm [17] is used to finely register the two scans. Mathematically, this alignment algorithm is searching for a rigid transform R that aligns two face surfaces (s_1 and s_2) so that they are equivalent ($s_1 \sim s_2$ if $\exists R \in \mathcal{R} | \forall p \in s_1, f_2(R(p)) = 0$). Detailed analysis of this two-step alignment algorithm shows it to be tolerant to variations in pose and lighting, as well as robust to changes in the algorithm parameters. The Surface Alignment Measure (SAM) is introduced as a measurement of the quality of the two-step alignment process, and experimental results demonstrate that the SAM score is a viable solution for a practical face verification system used with cooperative subjects.

Chapter 6 introduces face-based coordinate systems and defines the Canonical Face Depth Map (CFDM). The CFDM is defined with a face-based coordinate system that can be identified easily using different detection methods, and that generalizes over all types of human faces. Once the face-based coordinate system is identified in the 3D scan, the data are normalized (for scale and resolution) using an orthogonal depth map. This normalization enables faster and more accurate alignment, a reduction in memory requirements, and the CFDM conversion can function as a preprocessing step to a 3D version of PCA and a 3D version of correlation (note, in this dissertation the words "correlation" and "cross-correlation" are used interchangeably). The correlation algorithm is experimentally shown to achieve viable matching

performance and anchor point localization.

Chapter 7 (and the Appendix) give an overview of the software research platform developed and used in this dissertation. The research platform includes a real time demo of a practical 3D face verification system, as well as a toolbox of 3D processing algorithms that can be easily used on multiple operating systems and platforms. The research platform presented in Chapter 7 is designed as a foundation for future research in 3D face recognition.

Chapter 8 discusses the conclusions of this dissertation and presents directions for future work.

Chapter 2

Background

Chapter 1 introduced face recognition and verification using three dimensional data. This chapter provides additional background information on face recognition research and applications, and discusses how three dimensional data are collected and represented in this dissertation. Section 2.1 overviews existing work, and Section 2.2 outlines the basic notation used in this dissertation to represent three dimensional coordinate systems. Section 2.3 describes the various sensors used to generate three dimensional data, while Section 2.4 describes the different data structures used to represent and store three dimensional data. Finally, Section 2.5 describes the datasets used in this dissertation.

2.1 Existing Work

Biometrics is literally the measurement and analysis of biological data, and in medicine the term is used to describe an analysis of biological systems [140]. In computer science, particularly in security applications, the term biometrics is commonly used to refer to a semi or fully-automatic computer measurement of the human body for authentication [125]. The most common application of biometrics in computer science is the measurement of human data specifically for the purpose of person

recognition. There are generally two modes of recognition:

- Verification - Comparing a person's biometric to a single "claimed" identity.
- Recognition - Comparing a person's biometric to a [large] database of biometrics to find the closest match.

Both of these modes generally use the same underlying algorithms. However, there are some subtle differences. In verification mode, it is common to have a threshold that determines whether the two biometric measurements are similar enough to be from the same person. A database of training examples is generally used to establish this threshold, and in some cases a different threshold may be calculated for every person in the database [112]. The recognition mode, on the other hand, does not necessarily require a threshold or a "claimed" identity. Instead, the best match is selected from the entire database. This process can be quite time consuming, depending on the size of the database and the speed of the recognition algorithm.

There are a number of different biometrics that researchers have explored for recognition. Many of these are direct measurements of the human body and include fingerprints [79], faces [94], iris scans [38], palm prints [56], ears [145] and hand geometry [82]. Another class of measurements are indirect ones based on a subject's actions, such as speech [81], signature [85], or gait [95]. Biometrics are used to control access to resources [135], for security surveillance [77] and encryption [83], and for postmortem identification [36].

2.1.1 Face Recognition

The key goal for face recognition algorithms is to determine whether two facial images are from the same individual. Normally, there are two main steps in a face recognition system: the first step is to detect the face in the image, and the second step is to process the image for a particular application. Face recognition systems are used

to improve public safety; for example, current researchers are developing systems to detect human faces in video [146], to identify pedestrians at crosswalks [115], and to monitor and care for the elderly and impaired [122]. Human face recognition systems are also of interest in communications and entertainment [113], where models of the face are used to drive remote avatars and to improve video compression. All of these applications benefit from algorithms that are designed to understand and process the face efficiently.

Table 2.1 outlines the design space for 2D and 3D face recognition systems. Currently, the most common approach to face recognition uses a database of 2D faces to recognize 2D test images (upper left box in Table 2.1). Advances in 3D imaging technology make it possible to consider face recognition in three dimensions as well. 3D information inherently makes a face recognition system more robust to changes in pose and expression, and this dissertation focuses on systems that match 3D input to 3D data (lower right box in Table 2.1). Because 2D cameras are so affordable, future research may consider matching 2D surveillance images to a 3D database (upper right box in Table 2.1). Another, more limited, application is to have a database of 2D images and use a 3D scan to help verify a person within the images [3] (lower left box in Table 2.1). For example, the Mitsubishi Electric Research Laboratories [89] used 3D scanner technology to try to identify the subject of the famous image of a sailor celebrating his return from World War II with a kiss (see Figure 2.1). In this example, each person who claimed to be the sailor had a 3D scan taken, which was then measured against the 2D photo.

There are three general approaches to doing face recognition in both 2D and 3D [94]:

1. Analytic methods
2. Holistic methods
3. Hybrid of analytic and holistic methods

Table 2.1: Design space for 2D and 3D face recognition systems.

		Testing (Verification / Identification)	
		2D	3D
Training (Enrollment)	2D	Solution 1 Most common	Solution 3
	3D	Solution 2	Solution 4 This Dissertation

Figure 2.1: A database of 3D scans was used to try to clarify the identity of the sailor in this photo.

Analytic methods use knowledge about the face to measure important features on the face. For example, there is a long history of face recognition research based on the distance between sets of known anchor points on the face [102]. Researchers have explored face measurements in work that goes back to Leonardo Da Vinci (see Figure 2.2), and anthropometric face proportions are commonly studied in medicine [57, 53]. In anthropometric research, measurements are made on the surface of the face (see Figure 2.3) and compared across, age, gender and ethnicity.

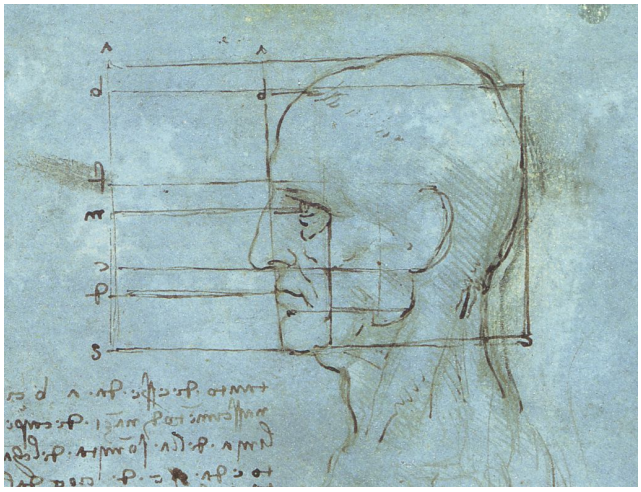


Figure 2.2: Early anthropometric work by Leonardo Da Vinci.

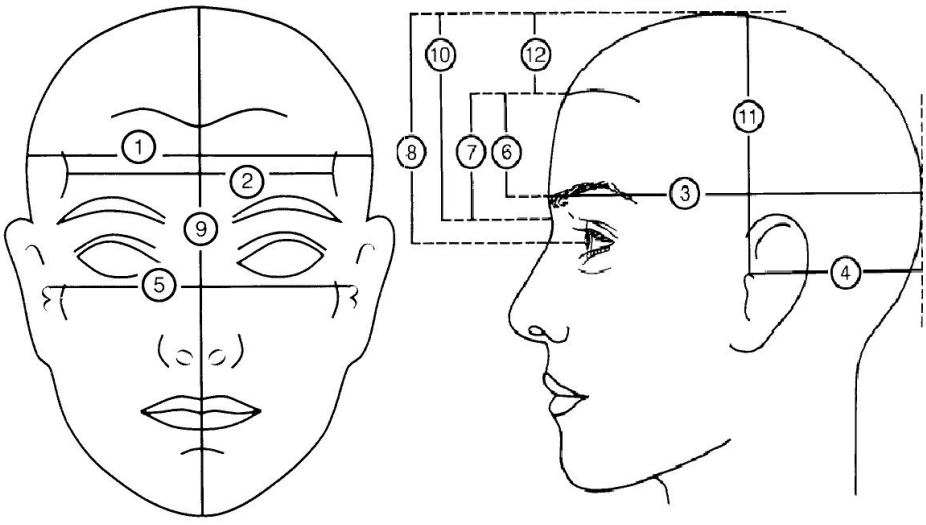


Figure 2.3: Example locations for anthropometric face measurements (Figure from [16]).

Early computer science research used anchor points for face recognition by manually selecting anchor points on 2D images, with various levels of successful recognition [86], and some recent 3D work is also based on manually selected anchor points [124]. Current research is exploring the use of automatically detected anchor points, but this work is limited to small datasets and the recognition results are susceptible to changes in pose and to noise in the data [91].

Another example of an analytic method for face recognition takes advantage of features that can be found in profile images. The measurement of profile faces for identification started with Sir Francis Galton in 1888 [63]. Galton studied different component measurements of the face and defined key feature points on the profile. Mathematically, Galton developed a method for describing irregular curves. Early applications of profile images in face recognition research relied on images taken with 2D cameras and compared using various techniques [126]. However, the addition of 3D scans allows the profile to be extracted from a wide range of scan poses. Results of these 3D experiments are mixed and many of the existing approaches still require near frontal images in order to accurately extract the profile pose [19].

Holistic methods treat the input vector as an unknown entity and apply learning algorithms to the entire vector, regardless of any heuristic knowledge of the objects being recognized. For example, Eigenfaces [133] is a holistic approach that generates the principal components of a set of training face images and maps new face images onto these vectors to identify individuals in real time. The Eigenface algorithm is frequently cited because of its simplicity and speed. However, this algorithm requires a high level of correlation between the faces in the database in order to achieve good performance, and any changes from the training conditions (including pose, lighting and expression) will hinder performance [93].

Analytic methods are powerful because they have the potential to be pose invariant as well as flexible to minor global changes (such as lighting, etc.). However, analytic

methods will break down when features are inaccurately detected. A hybrid approach combines analytic and holistic features. For example, Manjunath, *et al.* [102] use Gabor filters to identify non-specific features. These non-specific features tend to relate to higher-level features, such as the eyes, nose and mouth, but this result is due to the fact that these higher-level features are also good Gabor features. Hybrid methods are more trainable than analytic methods, making the hybrid methods a good compromise for face recognition research.

2.1.2 3D Face Processing

Much recent research has focused on the development of 3D face recognition technology. Some investigators have used the 3D processing component to normalize the pose and lighting of the input image in order to match the pose and lighting variations in the 2D gallery images. These mappings include simple 3D rotations, as well as more advanced morphing models that include variations in expression [21].

One of the findings of the Face Recognition Vendor Test (FRVT 2002) is that 2D face recognition systems can achieve good performance in constrained environments; however, both analytic and holistic approaches still encounter difficulties in handling variations due to head pose, lighting conditions and facial expressions [119]. Because the human face is a three-dimensional (3D) object whose 2D projection (image) is sensitive to changes in lighting, pose and expression, utilizing 3D information can improve face recognition performance [21, 33, 34, 35]. Range images captured explicitly by a 3D sensor [51, 108, 1] present both the face surface shape and texture. The 3D shape of facial surfaces represents the face structure, which is a function of anatomy rather than an artifact of external factors such as lighting, pose or makeup.

The Face Recognition Grand Challenge (FRGC) [118] was developed in part to introduce 3D face data to researchers to explore possible improvements over FRVT results. The 3D database for the FRGC was collected using structured lighting tech-

nology and has two channels of information. The first channel is a normal color image, and the second channel is a depth map with an x, y and z value for every pixel in the color image. The FRGC database contains only frontal scans taken in a controlled lighting situation and does not present enouph variations to adequately test for tolerance to lighting and pose changes. This database does, however, offer some indication of the pose variation that may be expected in a real world application.

In order to take advantage of the additional information provided by 3D scans for face recognition, researchers generally focus on four tasks:

1. 3D data collection methods.
2. Face normalization and input correction.
3. Develop features for 3D space face recognition.
4. Explore methods for merging 2D and 3D face recognition algorithms.

The first approach focuses on methods for collecting 3D data. Most 3D research relies on commercially available 3D scanners that use either structured lighting or stereo imaging. However, researchers are also looking at fitting models to image pairs. For example, Ansari and Abdel-Mottaleb [8] and Ip and Yin [72] use two orthogonal 2D images to generate a 3D model from common anchor points found in both images. Another method for generating a 3D model is to use two orthogonal 2D images [148], where one image is frontal and the second image is profile. This approach allows the xy plane to be defined by the frontal image and the zy plane to be defined by the profile image. The key to this 3D modeling technique is that the exact image point location is needed in order to fit the general 3D model onto the specific high resolution data. Recent methods have tried to generate a 3D model from a single 2D image by using a generic 3D model of the face and fitting the model to the 2D image. Blanz and Vetter [21] iteratively fit a 2D image to a complex 3D face model that accounts for lighting, expression and pose but takes a long time to

process. These 2D-to-3D methods are also available in commercial products, such as Animetrics [7] and CyberExtruder [50].

The second approach is to use 3D information to normalize the face data; this normalization includes pose, lighting/color and expression correction. In some algorithms [14, 124], the pose correction is determined by a set of manually selected anchor points and the pose-corrected 2D texture and shape images are fed into an off-the-shelf recognition algorithm. These anchor points are used to put the face into a partially canonical format. Current research in the use of automatically detected anchor points is limited to small datasets, and the results are susceptible to changes in pose and to noise in the data [91]. One problem with using anchor points is that even minor errors in their detection can cause large errors in the transformations. Cartoux *et al.* [31] used facial profiles extracted from 3D range images for face recognition. Malassiotis and Strintzis [101] used the 3D channel to compensate for illumination in the color channel and show some improvement on a small dataset. This algorithm is fully automatic, but assumes that the nose tip is the closest point to the scan - an assumption that cannot reliably be made in a real world application. Lu and Jain [98] and Bronstein *et al.* [27] present methods for modeling and normalizing expressions. However, methods that correct for expression are currently too slow for real world verification problems.

A third approach to using 3D information is to rely solely on the 3D channel and recognize the face without using the color component. This is desirable because it avoids problems with changes in lighting and color (e.g., makeup). Some feature spaces, such as curvature [131] and 3D Gabor filters [139], are at least partially pose tolerant. Other approaches assume normalized frontal scans and then apply existing 2D face detection algorithms, such as PCA [144, 25]. Xu *et al.* [144] use a standard sized mesh grid over the nose point to even out the feature space. Points are evenly sampled along the vertices of the grid so that there is a standard-sized feature vector,

and 3D PCA is used to identify the faces in a 120 subject dataset, achieving at best around 93% accuracy. The following list describes different feature spaces and approaches that use 3D data:

- Lee and Milios [90] segmented the range image to obtain the convex regions based on the sign of the mean and Gaussian curvatures at each point. These convex regions correspond to distinct facial features. The Extended Gaussian Image (EGI) is used to represent each convex region. A similarity metric between two regions is defined to match the facial features of the two face images.
- Gordon [67] explored face feature extraction for recognition based on depth and curvature features.
- Achermann and Bunke [4] introduced a 3-D version of the partial Hausdorff distance. Experiments on a database with 240 images of 24 persons yielded nearly perfect results.
- Beumier and Acheroy [19] extracted the profiles (curves) both from depth and gray-scale images for face verification.
- Hesher *et al.* [70] applied PCA to the range image, and estimated probability models for the coefficients.
- Bronstein *et al.* [26, 27] proposed an algorithm based on geometric invariants in an attempt to deal with facial expression variations for face recognition.
- Bellon *et al.* [13] proposed a Surface Inter-penetration Measure (SIM). For every point in the scan, a tangent plane is determined and a local neighborhood is analyzed. If there are local points both above and below the plane, then the point has a high SIM score. This measurement avoids many of the problems that are encountered by RMS errors. However, this calculation can take time and these experiments were just a proof of concept, using a small number of subjects.

Each of these feature spaces have different advantages and disadvantages, and no one feature space has been shown to significantly outperform the others. Boywer *et al.* [24] present a good overview of current approaches and challenges in 3D face recognition. However, it is difficult to directly compare these results since different systems are evaluated under different conditions and use datasets of varying quality.

The fourth approach to utilizing 3D data is to combine 2D and 3D recognition channels into a single classifier. Combining 2D and 3D face recognition systems has been shown to achieve better performance than each individual algorithm alone [25]. However, much of the testing of these multimodal systems does not consider the variations in pose and lighting that were the original motivation for utilizing the 3D information. Work by Chang *et al.* [35] demonstrated that face recognition systems based on either two-dimensional texture information or 2.5D range information have similar performance characteristics. However, they went on to show that significant improvements can be made if a system uses a combination of texture and shape information, such as applying PCA to both 2D and 3D face data.

Although some of these approaches to 3D face recognition applications are promising, none develop a complete, viable recognition system and all fail in at least one of the following areas:

1. The system is not fully automatic and/or is not practical for real world applications.

2. Testing is only done with small datasets and results do not generalize.

3. Testing is done on a dataset with only minor variations in lighting and pose and results do not generalize.

4. Extensive training sets are required to achieve good performance, which makes the system impractical and difficult to maintain because new training sets are required every time the application environment changes.

2.2 Coordinate System Transformations

This dissertation deals with points in 3D space ($p \in \mathbb{R}^3$) where p is a triple with x, y, z coordinates ($x, y, z \in \mathbb{R}$). $\mathcal{T}$ is a set of all $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ transformations and is defined as follows:

$$\mathcal{T} \equiv \{T | T : \mathbb{R}^3 \rightarrow \mathbb{R}^3\} \tag{2.1}$$

$\mathcal{R}$ is a set of rigid transforms that is a subset of $\mathcal{T}$ ($\mathcal{R} \subset \mathcal{T}$). A rigid transform has two properties. First, it conserves distances over the standard Euclidean metric d:

$$D \equiv \{T \in \mathcal{T} \mid \forall p_1, p_2 \; d(p_1, p_2) = d(T(p_1), T(p_2))\} \tag{2.2}$$

A rigid transform also maintains the relative handedness of the points. This means that the rigid transform can be represented by a translation and a rotation about a single axis. The class of rigid transforms does not include reflections. This second property can be described using the following cross product constraint:

$$\begin{aligned} H \equiv \{&T \in \mathcal{T} \mid \forall p_1, p_2, p_3, \\ &\overline{(p_2 - p_1)} \times \overline{(p_3 - p_1)} = \overline{(T(p_2) - T(p_1))} \times \overline{(T(p_3) - T(p_1))}\} \end{aligned} \tag{2.3}$$

All rigid transformations ($\mathcal{R}$) are the combination of both constraints represented by the sets D and H.

$$\mathcal{R} \equiv D \cap H \tag{2.4}$$

An object in $\mathbb{R}^3$ can be defined by a relationship of the form $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ where

a point p is a part of object b if the following surface relationship holds:

$$b \equiv \{p | f(p) = 0\} \tag{2.5}$$

This definition could represent any set of points in $\mathbb{R}^3$, however for this dissertation it will be assumed that this set of points falls on a surface $s \in S$ that obeys the following constraint:

$$S \equiv \{s | \exists f,\ f : \mathbb{R}^3 \rightarrow \mathbb{R}\} \tag{2.6}$$

Two surfaces (s_1, s_2 with relationships f_1, f_2) are equivalent if there exists a rigid transformation $R \in \mathcal{R}$ such that for every point in s_1 the transformed point is also in s_2:

$$s_1 \sim s_2 \text{ if } \exists R_1 \in \mathcal{R} | \forall p \in s_1, f_2(R_1(p)) = 0 \tag{2.7}$$

Every rigid transformation has an inverse denoted by R^{-1}. Since s_1 and s_2 are equivalent, the inverse is also equivalent, i.e., $s_2 \sim s_1$ and $R_2 \in \mathcal{R}$ is the same as R_1^{-1}.

$$\text{if } s_1 \sim s_2 \text{ then } R_1 = R_2^{-1} \text{ and } R_2 = R_1^{-1} \tag{2.8}$$

This equivalence relationship will break down in real world situations. For example, faces are not rigid objects - people can grow and change their expression. Also, sensors do not provide a continuous ideal surface. Instead, 3D scanners provide a 2.5D discrete depth map estimation of the ideal face surface and may have noise or holes in the data. However, this relationship does provide some insight as to the paradigm of surface alignment explored in this dissertation. The goal is to find a

rigid transformation $R \in \mathcal{R}$ such that the two surfaces are aligned as closely as possible. The surface relationship constraint f is represented as a distance measure. If a point is on the surface then its distance is zero; if a point is off the surface then the shortest distance from the point to the surface is returned by the surface relationship constraint. Ideally, if the two surfaces are the same then all distances will be zero. However, perfect alignment in a real world system is unlikely because of surface changes, discretization errors, sensor errors, etc.

Any three non-collinear points in $\mathbb{R}^3$ can define a coordinate system. Two types of coordinate systems are considered in this dissertation: sensor-based and object-based. Any coordinate system that is based on the location of the camera and is independent of objects in the camera's field of view is a sensor-based coordinate system. For example, the scans produced by the Minolta Vivid 910 have sensor-based coordinates. The origin of the scanner-based coordinate system is located on the front plane of the scanner, with the coordinates shown in Figure 2.4.

Figure 2.4: Scanner-based coordinate system.

Within any object-based coordinate system, the coordinates are invariant with object motion and are grounded in features related to the object. For example, the coordinate system shown in Figure 2.5 is attached to the face.

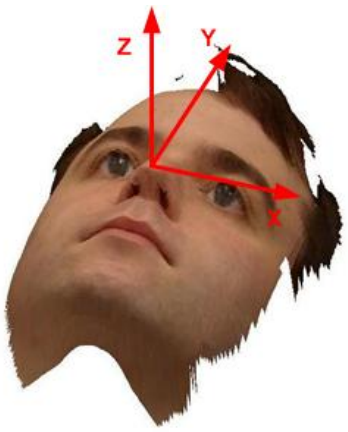


Figure 2.5: Object-based coordinate system.

Normally, 3D data produced by a sensor are presented in a sensor-based coordinate system because sensors are designed independently of the objects being sensed. This object independence allows the sensors to be used with a wide range of objects. However, object-based coordinates make assumptions about the object being represented. These assumptions can limit the variety of objects that can use a specific object-based coordinate system; however, the object-based coordinate system also enables correlation between objects of the same type. Good correlation is one of the most important prerequisites for many pattern recognition algorithms.

The mathematical representation of a surface assumes that the surface is continuous almost everywhere. However, the 3D sensor only uses a discretized sampling of this "true" surface. This sampling can have errors due to sensor noise or quantization in the data collection. A discretized 3D object is represented by a $4 \times n$ matrix, where n is the number of points, and each point has a location (x, y, z) and a flag (with a value of one or zero). The flag value is a place holder denoting points that are deemed invalid due to scanner error, occlusion, or any other form of noise or error. Any rigid

transformation $R \in \mathcal{R}$ is represented by a 4×4 transformation matrix:

$$P = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ z_1 & z_2 & \cdots & z_n \\ f_1 & f_2 & \cdots & f_n \end{bmatrix} \qquad R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} & t_x \\ r_{yx} & r_{yy} & r_{yz} & t_y \\ r_{zx} & r_{zy} & r_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

By multiplying the point matrix P by the transformation matrix R, the points are transformed into the new coordinate frame:

$$P' = RP \tag{2.9}$$

A 4×4 transformation matrix is used because it is simple to calculate and manipulate, and because it can represent any rigid transformation. The upper left 3×3 matrix R is the standard rotation matrix, and the vector $t = [t_x, t_y, t_z]^T$ is a translation vector. The rotation matrix R is an orthogonal rotation matrix where each row represents the new vector axes in the new coordinate system.

$$R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix}$$

The following equations give the simple rotations of angle α about the x, y and z axes:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & sin(\alpha) \\ 0 & -sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{2.10}$$

$$R_y = \begin{bmatrix} cos(\alpha) & 0 & -sin(\alpha) \\ 0 & 1 & 0 \\ sin(\alpha) & 0 & cos(\alpha) \end{bmatrix} \tag{2.11}$$

$$R_z = \begin{bmatrix} cos(\alpha) & sin(\alpha) & 0 \\ -sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

Any rotation can be represented as a single rotation angle α about a specific axis A. Given an arbitrary rotation matrix, this axis and rotation angle can be calculated as follows:

$$\alpha = acos(\frac{1}{2}(r_{xx} + (r_{yy} + (r_{zz} - 1)))) \tag{2.13}$$

$$A = \frac{1}{2sin(\alpha)} \begin{bmatrix} r_{zy} - r_{yz} \\ r_{xz} - r_{zx} \\ r_{yx} - r_{xy} \end{bmatrix}$$

Also, the Euler angles (roll, pitch and yaw) can be calculated directly from a rotation matrix. These calculations are dependent on the order in which the roll, pitch and yaw are defined. For this dissertation, the angles are defined as in Table 2.2.

Table 2.2: Calculating the Euler angles from a rotation matrix.

If $R_{xx} = 0$ and $Ryx = 0$ then:
$roll = 0.0$
$pitch = tan^{-1}(R_{xy}, R_{yy})$
$yaw = pi/2$
else
$roll = tan^{-1}(R_{yx}, R_{xx})$
$pitch = tan^{-1}(R_{zy}, R_{zz})$
$yaw = tan^{-1}(-R_{zx}, sqrt(R_{xx}^2 + R_{xy}^2))$

2.3 Sensor Selection

The most common sensor used in face recognition is a standard color camera, which captures a 2D (x, y) projection of an illuminated object residing in 3D space. During this 2D projection, the 3D information of the surface is mostly lost. However, 3D sensors have the potential to retain the 3D surface information (x, y, and z) of the face, which has been shown to improve recognition rates [35]. 3D systems can compare the depths of face anchor points in different scans, and the presence of depth information also allows for calculation of surface curvatures and surface normals that assist in face anchor point detection [46]. In addition to enhanced anchor point detection, 3D systems are more tolerant to adverse scanning conditions such as changes in lighting, pose, makeup, etc. There are four major sensor technologies for calculating 3D information:

- **Shape from Shading** uses a standard color camera to observe the natural shading of an object. Shading provides surface normal constraints and ultimately can be used to interpolate the 3D surface. Few systems take this approach because it is difficult to generalize and has low accuracy. The most successful applications of this method use artificial lighting to simplify the lighting model that is used to determine the surface shape [151]. Section 2.5.3 discusses a shape from shading system that has been used in connection with the algorithms described in this dissertation.

- **Stereo** systems use two cameras that are calibrated so that depth can be calculated by triangulating disparities between the images from the two cameras [28]. Correspondence between pixels is required to perform the triangulation. Calculating this disparity can be time consuming, and the results may be ambiguous if a reference point in one image matches more than one point in the counterpart image. In the most successful stereo systems, the cameras are rigidly aligned to

each other and the distance between the cameras is accurately calculated.

- **Structure from Motion** is similar to stereo methods. However, in structure from motion the disparity difference is calculated between two different frames of a video input from a single camera [132]. This method requires that there be motion between frames in order to create a disparity in the images, but the motion must be small enough for the system to find the correct correspondence between points in the frames [40]. If too much of the scene changes during motion, then the corresponding between points on a moving object may be inaccurate.

- **Structured Lighting** is similar to stereo, but replaces one of the cameras in the stereo rig with a known light source. The light source is projected into the scene and the camera picks up the location of the light to establish correspondence. The light can be structured so that correspondence calculations are simple and unambiguous. Most industrial scanning hardware uses some sort of structured lighting because it allows the system to be more accurate and more robust to environmental changes. The disadvantage of a structured lighting system is that it is not passive - it requires a light source that alters the environment. An example of a structured lighting system is the Minolta Vivid 910 scanner, which is the primary scanner used for data collection in this dissertation (see Section 2.5).

2.4 Existing 3D Modeling Methods

Many data structures have been developed to represent 3D information. This section examines some of the most common 3D data representations and categorizes their advantages and disadvantages using the following criteria:

- **Surface / Volume Representation** - This dissertation deals primarily with surface representations of the face. However, some data representations are volumetric and can represent more than just the surface of an object, such as scanners used in medical imaging (e.g., CAT scan).

- **Occlusion and Blind Spots** - Some surface representations are limited to specific directions or structures. These limitations can make algorithms more efficient, but can also cause holes in the data due to occlusions.

- **Neighborhood and Localized Registration** - Most 3D scanning hardware deals with unordered raw surface data. However, the neighborhood relationships between the points can also be useful. The neighbors of a point are the other points in the representation that are nearby in space. A representation that has access to this extra information can improve the speed of many algorithms, including correlation (see Section 6.6.1) and surface fitting (see Section 3.1).

- **Face Registration / Orientation** - Most 3D scanning hardware deals with raw surface data and does not include any contextual knowledge of the objects being modeled. However, by adding contextual information about the face (such as anchor points or axis registration) to the data structure, the processing can be specialized for dealing with this contextual data.

- **Memory Allocation** - Data formats can also be compared based on their required memory capacity. Many modern 3D scanners measure more points than are necessary for most applications. These extra points are very close together in 3D space and may not add additional information. Storing these extra points can cause problems in applications where bandwidth or memory storage limitations are an issue.

- **Extra, invalid data storage** - Another problem with some data formats is

that they can have a rigid structure that requires information even when none is available. These data formats require extra flags to show when data are invalid.

Table 2.3 lists some common 3D data structures and outlines their pros and cons using the criteria just described. The remainder of this section describes each of these data structures in more detail, with the exception of the Canonical Face Depth Map (CFDM), which is described in Chapter 6.

All of the data formats described in this section have advantages and disadvantages depending on the intended application of the data. For communications, a polygonal model is usually adequate to display a face in a realistic way. In addition, a structured polygonal model can be used to greatly reduce the size of the model and enable transmission over low bandwidths. For identification and recognition, it is important for a data format to be small enough to fit on a smart-card (512 KB [66]), and contain enough information to disambiguate individuals within a database.

Table 2.3: 3D model data structure comparison.

	Data Type	**Blind Spots**	**Neighbor Info**	**Face Reg**	**Memory Allocation**	**Invalid Data**
Point Cloud	Volumetric	No	No	No	Small	No
Polygonal Models	Surface	No	Yes	No	Small	No
2.5D Point Matrix	Surface	Yes	Yes	No	Large	Yes
Depth Map	Surface	Yes	Yes	No	Small	Yes
Structured Polygonal Models	Surface	No	Yes	Yes	Very Small	No
Face Mesh	Surface	Yes	Yes	Some	Small	Yes
Cylindrical Point Matrix	Surface	Some	Yes	Some	Large	Yes
CFDM (Chap. 6)	Surface	Yes	Yes	Yes	Small	Yes

2.4.1 Point Clouds

A point cloud is an unordered set of n points in space. Each point p_i is a vector that contains information about the point, such as location (x, y, z) and color (r, g, b).

$$\mathrm{p}_i = \{x_i, y_i, z_i, r_i, g_i, b_i\} \text{ where } 1 \leq i \leq n$$

Each point p_i is a sampling from an object in 3D space (see Figure 2.6). This set of sampled points is normally assumed to be dense enough to adequately represent the surface information, even the volumetric information. The disadvantages of point clouds are a lack of neighborhood information and a lack of information needed to determine which points are connected.

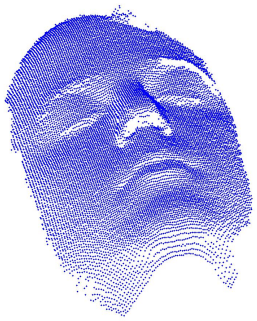


Figure 2.6: Point cloud (surface only).

2.4.2 Polygonal Models

The polygonal model is an extension of a simple point cloud that also includes neighborhood information (see Figure 2.7). This representation of 3D objects includes two lists: a list of points and a list of polygons.

$$p_i = \{x_i, y_i, z_i, r_i, g_i, b_i\} \text{ where } 1 \leq i \leq n$$
$$poly_j = \{v_1, \cdots, v_{c_j}\} \text{ where } 1 \leq j \leq m \text{ and } c_j = \text{number of vertices}$$

The list of points p_i is the same as in the point cloud representation. In addition to this list, the polygon list of size m stores all of the polygons within the model. Each polygon ($poly_j$) is a list of vertices that correspond back to the original point list. The number of vertexes (c_j) in a polygon can vary to represent polygons with different numbers of sides. The advantage of this structure is that the polygons represent the entire surface between the vertexes. Piecewise planes can be calculated for each polygon ($poly_j$) by decomposing the polygon into triangles. Each triangle represents a plane with coefficients (a, b, c, d) and an infinite number of points (x, y, z) can be sampled from these planes, which allows the entire surface to be represented at any level of detail desired:

$$ax + by + cz + d = 0.$$

Many graphics cards are optimized for data in a polygonal format and algorithms such as texture mapping and ray tracing have been optimized to work with this data format [110, 78, 44, 62]. Polygonal models require fewer points than in the point cloud format because the surface planes can represent many coplanar surface points instead of needing a large sampling of points.

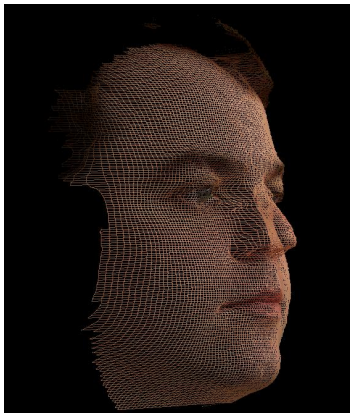


Figure 2.7: Polygonal mesh.

2.4.3 2.5D Point Matrix

The 2.5D point matrix is commonly used by 3D scanner manufacturers, such as in the Minolta Vivid 910. The 2.5D point matrix is similar to a point cloud. However, instead of storing the points as an unordered vector, the points are stored as a set of 2D matrices with n rows and m columns.

$$\text{point}_{(i,j)} = \{x_{(i,j)}, y_{(i,j)}, z_{(i,j)}, r_{(i,j)}, g_{(i,j)}, b_{(i,j)}, flag_{(i,j)}\},$$
$$\text{where } 1 \leq i \leq n \text{ and } 1 \leq j \leq nm.$$

The advantage of a 2.5D point matrix over a normal point cloud is that the point matrix is ordered such that the neighboring points in the matrix are neighboring points in 3D space. The points are a 2.5D projection of the surface of the object in 3D space. For example, the Minolta Vivid 910 outputs a standard 2D 320×240 pixel color image. In addition, the range data is stored in a set of 320×240 x, y and z matrices that allow exact registration with the original color image (see Figure 2.8). One limitation of the 2.5D format is that the system only represents surface information and must include a $(n \times m)$ flag matrix to account for missing data. A flag is zero if the point is invalid and one if the point is valid. Another problem with any 2.5D representation is that the format does not allow full 3D representation of the surface due to occlusions.

(a) Color Image (r, g, b) (b) x matrix (c) y matrix (d) z matrix

Figure 2.8: 2.5D point matrix representation. Lighter colors represent larger values of x, y and z and darker colors represent smaller values.

2.4.4 Depth Map

Instead of requiring three matrices to encode the 3D coordinates of a point cloud, the information can also be stored in a single matrix with a known scale factor. In this model, only the z matrix is used but is re-sampled as an orthogonal projection of the data onto a flat plane. In this way, the index values of the points encode the x and y information. The x and y resolution $(step_x, step_y)$ and starting point $(xmin, ymin)$ must also be stored, but if the x and y resolutions are known then the x, y, z location of a point can be determined using only a single matrix. In addition, an extremely large or extremely small value $(zthresh)$ can be used to represent the flag.

$$
\begin{aligned}
x_{(i,j)} &= i \cdot step_x + xmin \\
y_{(i,j)} &= j \cdot step_y + ymin \\
z_{(i,j)} &= z_{(i,j)} \\
flag_{(i,j)} &= z_{(i,j)} > zthresh
\end{aligned}
$$

So, information that requires four matrices in the 2.5D point matrix format can be reduced to a single matrix in the Depth Map format (see Figure 2.9). However, information will be lost because the Depth Map is even more rigid than a 2.5D Point Matrix and will only represent an orthogonal projection of one side of the surface. This is because the orthogonal projection requires re-sampling of the data. This re-sampling is not invertible and may lose some information from the original scan.

2.4.5 Structured Polygonal Models

Modifications to the standard polygonal model can also be made. One limitation of all of the previously described models is that none of them take into consideration the natural shape and pose of the human head. In fact, each of the representations can represent any 3D object and the same face can be represented in an infinite number

Figure 2.9: Single matrix depth map.

of ways, making comparisons of two different faces difficult. In computer graphics, structured polygonal models of the face are used to standardize the location of the polygons on the face (see Figure 2.10). This allows for simpler models that make it easier to map a texture onto the surface. The only information that needs to be stored is the location of the vertex points in the model. The polygonal nature of this representation makes rendering face images extremely fast in modern graphics processors. However, automatically extracting this information from an arbitrary scan can be difficult (see Chapter 4). The structured polygonal model may also provide less information about the surface structure of the face, therefore less information is available when disambiguating between subjects for identification.

2.4.6 Face Mesh

The face mesh [144] is specifically designed for registering raw scanner data. The face mesh assumes that the scans are frontal poses and registers the face using the nose tip. After registration, an evenly spaced mesh is applied to the face and at each intersection of the mesh the surface distance is re-sampled in such a way that the result is an evenly spaced data structure. The mesh can be adjusted for different levels of detail (see Figure 2.11). An advantage of this mesh is that it standardizes

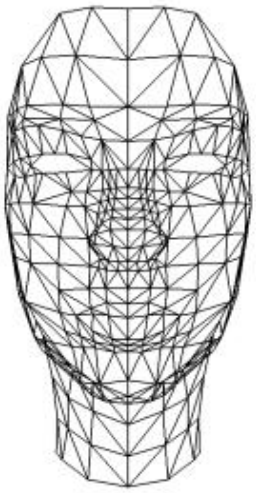


Figure 2.10: Structured polygonal model (image from [92]).

the face feature vectors and closely aligns all the faces into a face-based coordinate system.

Figure 2.11: Mesh models in different levels of detail (Figure from [144]).

2.4.7 Cylindrical Point Matrix

When dealing with full 3D models of the face, the data are transformed into a Cylindrical Point Matrix to represent the entire 3D surface of a face model. This format stores the X_C, Y_C, Z_C information in three 300×1000 matrices (see Figure 2.12), similar to the 2.5D format. However, these new matrices are indexed based on cylindrical coordinates instead of Cartesian coordinates and the points are re-sampled based on the centroid of the face model (see Figure 2.13). The cylindrical coordinate system allows for a richer modeling of the face structure than the 2.5D representation because all viewpoints of the face can be encoded.

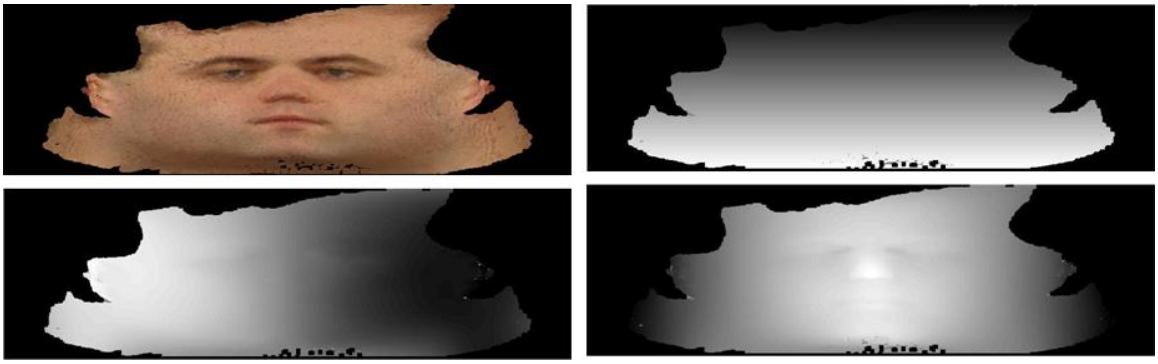


Figure 2.12: Example cylindrical data structure.

Figure 2.13: Example cylindrical data structure.

A major limitation of many 3D data structures is their inability to account for occlusions. The 2.5D scans generated by many 3D scanners only look at one side of a 3D object, and cannot represent a full 3D model of the head's surface. One notable exception is a scanner developed by Cyberware [51]. This scanner moves a vertically mounted laser around the subject to generate a full model of the subject in a format similar to the Cylindrical point matrix. The Cyberware scanner still produces occlusions around regions like the ears, nostrils and hair but these regions are much smaller than the regions that are occluded in a 2.5D scan.

In the approach to arbitrary pose anchor point detection shown in Section 4.4, a full 3D model of the head is used as a reference frame in order to align any arbitrary pose scan with a face. Various methods were considered for producing a 3D model from 2.5D scans. Five test scans, generated with a Minolta Vivid 910 scanner, were combined. For each subject, the scans were stitched together using a commercial software package, called Geomagic Studio [130]. The models were also cleaned up by filling holes, smoothing the surface, and deleting noisy points associated with hair and clothing. The resulting models had thousands of redundant points and polygons, so a decimate function was used to reduce the number of polygons in the model to a reasonable level (approximately 50,000). The end result was a smooth, full view of the face for each of the subjects (see Figure 2.14).

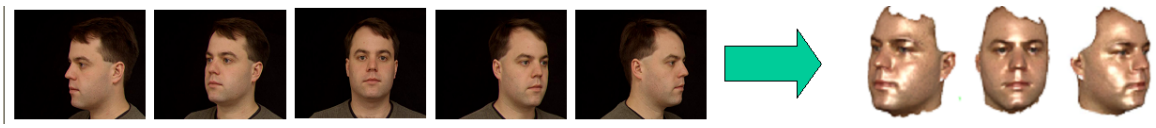


Figure 2.14: Model construction. Five 2.5D test scans used to construct a full 3D model.

Once the full 3D model of the face was generated, the models were output in 3D VRML (Virtual Reality Modeling Language) format. The 3D VRML format was chosen because most 3D editing programs can generate VRML output, so the system can be tested on face models generated using various types of scanners, software and

algorithms. The VRML format is an unordered polygonal model of the face. The unordered nature of the data makes it difficult to estimate the local structure or to calculate the local surface curvature and the surface parameters (such as surface normals) at each point. The cylindrical coordinate system was developed to resample the 3D model and store the neighborhood information in a matrix similar to a 2.5D representation. However, since this representation stores the points in cylindrical coordinates, it can represent all viewpoints of the 3D model. There are places where occlusions are still a problem; however, these locations are much smaller than in the 2.5D format.

In order to generate the cylindrical model, the points in the polygonal (VRML) model need to be resampled. The polygonal model is represented as a set of k points that are vectors $p_i = x_i, y_i, z_i$. The transformation is made using the algorithm shown in Figure 2.15. It is assumed that the orientation of the model is vertical, and that there is a vertical cylindrical axis passing though the centroid of the model (x', y', z'). After the vertical axis is determined, a rotational orientation about the axis is needed. The tip of the nose was chosen to be located at an angle π. Now, the matrix can be sampled such that there are $nrows$ rows and $ncols$ columns. For this research, $nrows = 300$ and $ncols = 1000$ were chosen, with the vertical range approximately equal to the vertical range calculated by taking the minimum and maximum y values from the model.

// Initialize Variables
$ncols = 1000$, $nrows = 300$;
$distanceLimit = 6$;
$min_y = min_{i=1}^{k}(y_i)$;
$max_y = max_{i=1}^{k}(y_i)$;
$size_y = max_y - min_y$;

// Calculate center of model.
$x' = \frac{\sum_{i=1}^{k} x_i}{k}$, $y' = \frac{\sum_{i=1}^{k} y_i}{k}$, $z' = \frac{\sum_{i=1}^{k} z_i}{k}$

// Transform all model points onto Cylindrical Matrix
Loop from $i = 1$ to k
    $x_i = x_i - x'$, $y_i = y_i - y'$, $z_i = z_i - z'$;
    $\theta = atan2(x_i, -z_i)$;
    $h = \frac{y - min_y}{s(2)} \cdot ncols$;
    $a = \frac{\theta}{2*\pi} \cdot nrows$;
    $X_C(h, a) = x_i$, $Y_C(h, a) = y_i$, $Z_C(h, a) = z_i$
    $im_C(h, a) = im_i$
    $Flag_C(h, a) = true$
end Loop

// Fill in the missing data in the Cylindrical Matrix
Loop from $h = 1$ to $nrows$
    Loop from $a = 1$ to $ncols$
    If $Flag_C(r, c) == false$
        $\theta = 2\pi \frac{c}{ncols}$;
        $rayvector = [sin(\theta), \frac{r}{nrows*size_y} + min_y, -cos(\theta)]$;
        If $rayvector$ is valid
            $triangleIndexes = findIntersectingTriangle(rayvector)$;
            $[X_C(h, a), Y_C(h, a), Z_C(h, a)] = findIntersection(triangleIndexes, rayvector)$;
            $im_C(h, a) = InterpolateColor(triangleIndexes, X_C(h, a), Y_C(h, a), Z_C(h, a))$;
            $Flag_C = true$
        end If
    end If
end Loop

Figure 2.15: Algorithm for computing the cylindrical coordinates from an arbitrary point cloud.

2.5 Data Collection

Much of this dissertation focuses on the study of 3D models produced by the Minolta Vivid 910 3D scanner (see Figure 2.16). This scanner can generate a 2.5D image of the face in less than 2 seconds. A 2.5D image is a simplified 3D (x, y, z) surface representation that contains at most one depth value (z direction) for every point in some (x, y) view plane (see Figure 2.17). The Minolta Vivid 910 produces measurements accurate to less than 0.2mm [22]; however, the cost ($\approx \$50,000$) and slow speed of the Minolta Vivid 910 scanner make it an unrealistic solution for many applications that could otherwise take advantage of 3D information.

Figure 2.16: Minolta Vivid 910 2.5D scanner.

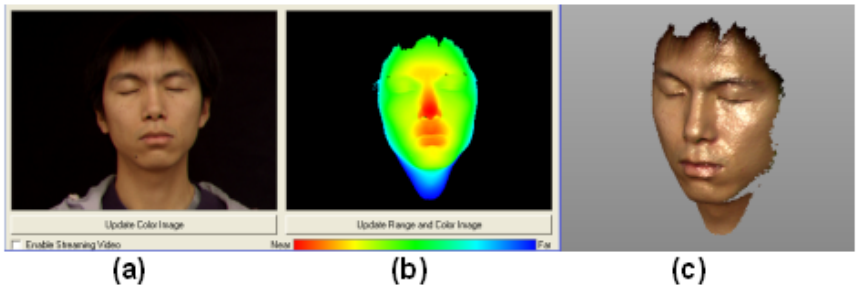


Figure 2.17: Example 2.5D image (a) 2D texture image (b) depth map (c) rendered 3D model.

The Vivid 910 uses structure from lighting by passing a horizontal plane of laser light onto the surface of an object, and then a 640×480 pixel color camera picks up

the curve produced by the interaction of the laser with the object, and triangulates the depth of the illuminated surface points. An advantage of this type of sensor is that it is well calibrated and will produce scan depth values accurate to less than a millimeter ($\pm 0.10mm$ in fine mode).

The Minolta Vivid 910 has several different modes, each with different speeds and resolutions. The fast mode is used in this dissertation because it is faster and still produces enough data for recognition. The fine mode is used by the FRGC and has more pixel information but it is harder for subjects to hold still. Table 2.4 and Figure 2.18 give a detailed list of the amount of time each camera mode requires for each scan. For every pixel, the camera outputs the Cartesian coordinates of the pixel (x, y, z), the color of the pixel (r, g, b), and a flag value indicating whether or not the laser was detected. The Minolta Vivid 910 scanner is commonly used in face detection and is currently the primary 3D scanner used to gather 3D face data for the Face Recognition Grand Challenge [118].

Table 2.4: Operation of the Vivid 910 scanner (time in seconds).

	Fast	Fine	Description
Camera set up	14.2		Camera calibration and setup. This should only happen when the system starts up.
Focus	3.4		Minor Focus Adjustment if the subject moves closer or farther from the camera.
Lead	0.072		Scanner setup time.
Scan	0.87	3.74	Scanning time.
Color	0.85		Record 3 picture color image.
Release	1.72	4.59	Total time Subject needs to hold still.
Download	1.04	1.92	Download image from scanner to computer.
Total	2.83	6.58	Total time for processing (without setup or focus).

2.5.1 Michigan State Dataset

For the experiments described in this dissertation, the Vivid 910 was set to record a 320×240 pixel 3D scan in less than a second. At Michigan State there have been

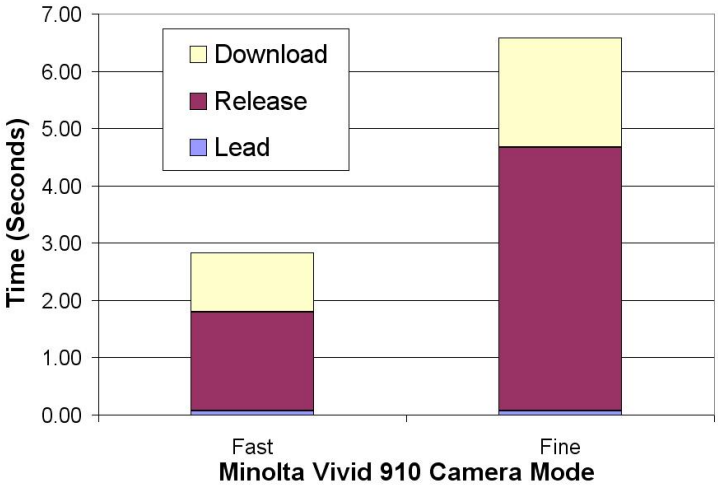


Figure 2.18: Scanning time.

three phases of data collection:

- Phase I (arbitrary pose database) - In the first phase, about 11 scans were taken for each of 111 subjects. These scans included the following poses and expressions: neutral expression (2 frontals, 30°, 45°, 60° to the right, and 30°, 45°, 60° to the left) and smiling expression (frontal, 45° to the right, and 45° to the left).

- Phase II (frontal pose database) - In the second phase, subjects were scanned using a less rigid procedure. Many of the 80 or so subjects in this database only have two scans (one frontal neutral expression and one frontal smiling expression).

- Phase III (additional scans) - In addition to the first two databases, a number of test scans were made using different expressions, occlusions, facial hair, etc. These scans are used to analyze special cases in this dissertation.

A database of scans of a mannequin head was also generated to establish a testing foundation for the registration algorithm.

2.5.2 Face Recognition Grand Challenge Datasets

In addition to generating data at MSU, this dissertation research also uses data provided by the Face Recognition Grand Challenge (FRGC) [118]. The aim of FRGC is to encourage the development of new algorithms for face identification. One of the goals of FRGC is to introduce a 3D dataset to the research community in order to encourage algorithm development. The FRGC dataset is the largest publicly available dataset of 3D faces and currently has two versions. Table 2.5, Figure 2.19 and Figure 2.20 show some of the statistics associated with the FRGC data. Table 2.6 outlines the attributes of the FRGC and MSU datasets.

Table 2.5: FRGC database.

Version	Release Date	Subjects	Subject Sessions	Recordings	Size
1.0a	May 2004	275	943	943	n/a
2.0	Sept 2004	625	n/a	4,007	70 Gbytes

2.5.3 Ongoing Work in 3D Scanners

New scanners have recently made 3D scanning more cost effective, although not always with the accuracy of the Minolta. The Canesta scanner is more affordable ($\approx$ \$200) than the Minolta; however, it is limited to an accuracy of $2cm$ at $1m$ distance from the scanner [30]. PhotonX [120] is also developing a scanner that may be available in quantities from around \$5000 and may generate 10 frames per second of high resolution 2.5D scans. The scanner uses a multi-image, shape from shading technique to generate surface normals and then uses local interpolation to calculate the depth values.

Currently, only a preliminary proof of concept prototype (see Figure 2.21) of the PhotonX scanner has been tested with the frontal face detection algorithm described in Chapter 7. The prototype system still requires a controlled lighting situation and some manual input to seed the depth values. However, as this technology matures, it

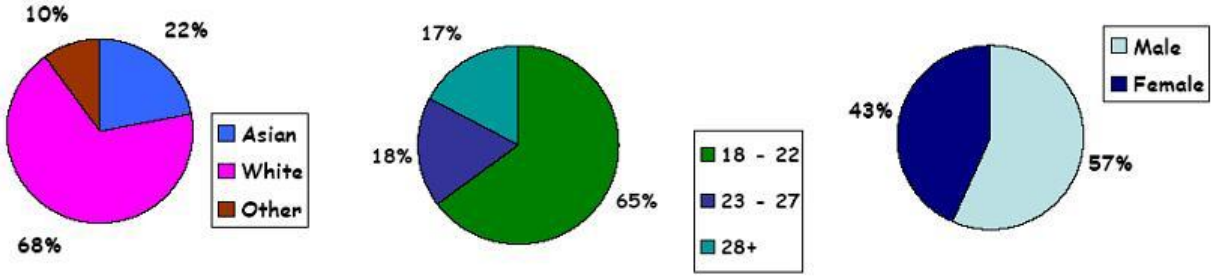


Figure 2.19: FRGC version 2.0 demographics (ethnicity, age, gender). (Figure from [118]).

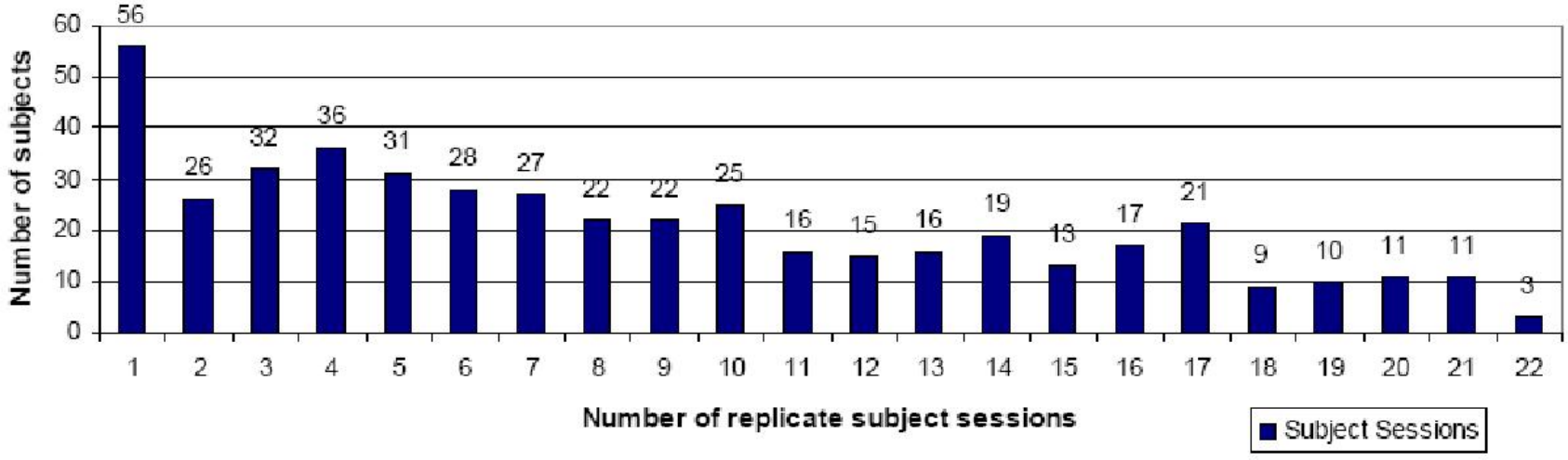


Figure 2.20: FRGC version 2.0 sittings. (Figure from [118].)

Table 2.6: Comparison between the FRGC and MSU datasets.

Issue	FRGC	MSU	Discussion
File Size	640×480	320×240	The larger FRGC files have more information
Download Time	1.9 (Sec)	1.0 (Sec)	The larger file size causes slower download times.
Scanning Time	4.6 (Sec)	1.7 (Sec)	Even though the FRGC data has more information, the slower scanning time allows for subject movement and causes more errors in the FRGC dataset.
Pose	Frontal Only	Tested on non-frontal	Even though one of the goals of 3D data is to make face recognition tolerant to pose variations, the FRGC dataset only contains minor pose variation. However, these small variations allow accurate measurement of pose variations in a cooperative subject pool.
Lighting	Minimal Light Variation	Range of Lighting	Another goal of using 3D data is to make face recognition more tolerant to lighting. However, most of the FRGC data was collected with constant lighting so there is little opportunity to test 3D algorithms under varying lighting conditions using the FRGC dataset.
Expression	Extreme Expression-	Minor Variations-	The FRGC dataset has neutral expressions or extreme "uncooperative" expressions, which are particularly difficult to record using the slower scanning time. About 2/3 of the MSU dataset is neutral expression and the other 1/3 with minor smiles (no teeth). This makes the MSU dataset a more reasonable representation of real world cooperative subjects.
Anchor Points	Manual	Automatic	The FRGC provides manual anchor points to help facilitate the exploration of feature spaces. However, any practical system would need automatic selection of anchor points. The automatic anchor points may not be as good as the manual points, and the recognition system needs to take anchor point errors into consideration.

could feasibly work at near real time and at long distances. The greatest challenge is the vast amount of data that needs to be processed. PixelVelocity [121] has worked with MSU and PhotonX to implement a high data bandwidth solution that uses highly parallel Xilinx processes directly attached to the image charge-coupled device (CCD).

(a) PhotonX Camera Setup (b) Example Image

Figure 2.21: Example prototype setup for the PhotonX multi-image shape from shading camera.

Chapter 3

Solution Approach

Chapter 2 introduced coordinate systems and mathematical notations, described different scanning methods, and introduced 3D data formats. This chapter presents methods and algorithms for working with three dimensional data; concepts that are used throughout this dissertation. Section 3.1 describes an algorithm for fitting a continuous surface to a discrete set of 3D points. Section 3.2 describes the shape index, a pose-invariant feature space that helps deal with variations in input data due to pose, resolution and scale. Section 3.3 describes issues that need to be considered when dealing with scanner data at different camera offsets. Section 3.4 presents an algorithm to address common types of noise in 3D scanner data. Section 3.5 describes methods for aligning scans taken in different sensor-based coordinate systems. Section 3.6 describes methods for reprojecting arbitrary 3D data in order to produce artificial 3D scans.

3.1 Surface Fitting

This section describes a method for fitting a smooth parametric surface to a set of discrete points. This surface can be used to interpolate unavailable points or to calculate the surface curvature [58].

Assume that the local surface structure of a face scan is smooth. This is a reasonable assumption since most faces are smooth with some minor exceptions, such as occlusion from the nose or the small "step functions" at the eyelid and mouth. The surface properties of each point on the scan can be calculated by fitting a small cubic surface function to the neighborhood around the point. A cubic polynomial (see Equation 3.1) was chosen because it is locally smooth, can fit the several structures of the face, and is simple to calculate.

$$z = ax^3 + bx^2y + cxy^2 + dy^3 + ex^2 + fxy + gy^2 + hx + iy + j \tag{3.1}$$

The first step in fitting the surface in Equation 3.1 is to select a local neighborhood of n points around a particular point p' in the scan. Then, the coordinates of each of the n points are adjusted such that they are shifted to make p' the origin.

$$\overline{p_i} = p_i - p' \tag{3.2}$$

$$P = [\overline{p_1}, \overline{p_2}, \cdots, \overline{p_n}] \tag{3.3}$$

The second step in fitting the surface is to calculate the covariance matrix of these points and transform them into a principal component coordinate system:

$$B = eigenvectors(\sum_{i=1}^{n} \overline{p_i} \cdot \overline{p_i}^T) \tag{3.4}$$

$$P_n = B \cdot P \tag{3.5}$$

Once the points have been normalized into the principal space, the coefficients for Equation 3.1 are solved using least squares fit of the following equation:

$$B = AX \tag{3.6}$$

where:

$$A = \begin{bmatrix} x_1^3 & x_1^2 y_1 & x_1 y_1^2 & y_1^3 & x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2^3 & x_2^2 y_2 & x_2 y_2^2 & y_2^3 & x_2^2 & x_2 y_1 & y_2^2 & x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^3 & x_n^2 y_n & x_1 y_n^2 & y_n^3 & x_n^2 & x_n y_n & y_n^2 & x_n & y_n & 1 \end{bmatrix} \tag{3.7}$$

$$B = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \qquad X = \begin{bmatrix} a \\ b \\ \vdots \\ j \end{bmatrix} \tag{3.8}$$

Solving for X gives the coefficients of equation 3.1. This solution is a parametric local representation of the sampled points. This new continuous surface is also differentiable, and therefore can be used to calculate the surface curvature.

3.1.1 Curvature Calculation

The curvature of a surface is defined as the second derivative of the surface, and on a 3D surface there are different curvatures for each direction along the surface. The maximum and minimum curvatures are orthogonal to each other and are calculated using the Hessian matrix, similar to [127, 55]. The Hessian matrix is defined from the second partial derivatives of a function:

$$Hessian = \begin{bmatrix} \frac{d^2 z}{dx^2} & \frac{d^2 z}{dxdy} \\ \frac{d^2 z}{dydx} & \frac{d^2 z}{dy^2} \end{bmatrix} \tag{3.9}$$

The first derivatives of Equation 3.1 are:

$$\frac{dz}{dx} = 3ax^2 + 2bxy + cy^2 + 2ex + fy + h \tag{3.10}$$

$$\frac{dz}{dy} = bx^2 + 2cxy + 3dy^2 + fx + 2gy + i \tag{3.11}$$

Similarly, the second derivatives are:

$$\frac{d^2z}{dxdy} = 2bx + 2cy + f \tag{3.12}$$

$$\frac{d^2z}{dx^2} = 6ax + 2by + 2e \tag{3.13}$$

$$\frac{d^2z}{dy^2} = 6dy + 2cx + 2g \tag{3.14}$$

When calculating the curvature at the origin, $x = y = z = 0$. Combining Equations 3.9 with 3.12, 3.13, and 3.14 gives:

$$Hessian = \begin{bmatrix} 2e & f \\ f & 2g \end{bmatrix} \tag{3.15}$$

Solve for the Eigenvalues of Equation 3.15 returns the maximum and minimum curvatures:

$$k_{1,2} = e + g \pm \sqrt{e^2 + 2eg + g^2 - 4eg + f^2} \tag{3.16}$$

This solution is equivalent to the solution presented in [58] which solves the following equation:

$$k_{1,2} = H \pm \sqrt{H^2 - K} \tag{3.17}$$

where the Gaussian curvature K, which is the determinant of Equation 3.15, and the mean curvatures H, which is one-half the trace of Equation 3.15:

$$k_{1,2} = e + g \pm \sqrt{(e-g)^2 + f^2} \tag{3.18}$$

These eigenvalues represent the minimum and maximum curvatures on the surface:

$$k_{max} = max(\lambda_1, \lambda_2) \tag{3.19}$$

$$k_{min} = min(\lambda_1, \lambda_2) \tag{3.20}$$

3.1.2 Surface Curvature and Normal Directions

The direction of the maximum eigenvector is in the principal plane and is calculated from the Hessian Matrix in Equation 3.9:

$$\begin{aligned} eig_1[x] &= \begin{cases} e - g + \sqrt{(e-g)^2 + f^2} & \text{if } e > g \\ f & \text{otherwise} \end{cases} \\ eig_1[y] &= \begin{cases} f & \text{if } e < g \\ -(e - g - \sqrt{(e-g)^2 + f^2}) & \text{otherwise} \end{cases} \\ eig_1[z] &= 0 \end{aligned} \tag{3.21}$$

Choose the first solution if $e > g$ and the second solution if $e < g$ [58]:

$$eig_1[x] = \begin{bmatrix} \frac{eig_1[x]}{eig_1[x]^2 + eig_1[y]^2} \\ \frac{eig_1[y]}{eig_1[x]^2 + eig_1[y]^2} \\ 0 \end{bmatrix} \tag{3.22}$$

The minimum eigenvector direction is orthogonal to the maximum eigenvector:

$$eig_2 = \begin{bmatrix} -eig_1[y] \\ eig_1[x] \\ 0 \end{bmatrix} \tag{3.23}$$

The direction of curvature in the original coordinate space is:

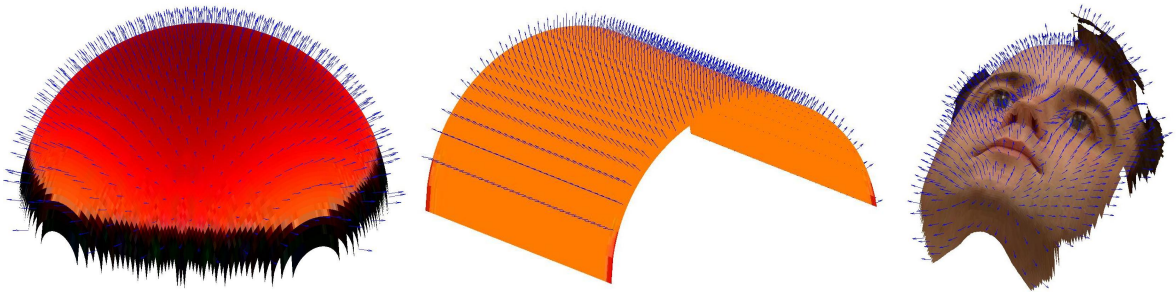
$$\vec{k_{max}} = B^{-1} eig_1 \tag{3.24}$$

$$\vec{k_{min}} = B^{-1} eig_2 \tag{3.25}$$

The surface normal is calculated by taking the cross product of the minimum and maximum curvature directions:

$$normal = \vec{k_{min}} \times \vec{k_{max}}; \tag{3.26}$$

There may be some (180 deg) ambiguity as to the direction of the surface normal. However, for 2.5D scans the z component of the surface normal should always be in the positive direction (if it is not, then the solution is flipped 180 deg). Figure 3.1 shows the surface normals calculated using the method described in this section for three surfaces.



(a) Sphere (b) Cylinder (c) Face

Figure 3.1: Example surfaces with surface normals represented by arrows.

3.2 Shape Index

The local curvature information about a point is independent of the coordinate system. Dorai and Jain [54] proposed local shape information, called the Shape Index, to represent the surface curvature at each point within a 2.5D image. The Shape Index at point p is calculated using the maximum (k_{max}) and minimum (k_{min}) local curvature. This calculation produces a shape scale between zero and one (see Figure 3.2).

$$S(p) = \frac{1}{2} - \frac{1}{\pi} tan^{-1} \frac{k_{max}(p) + k_{min}(p)}{k_{max}(p) - k_{min}(p)} \tag{3.27}$$

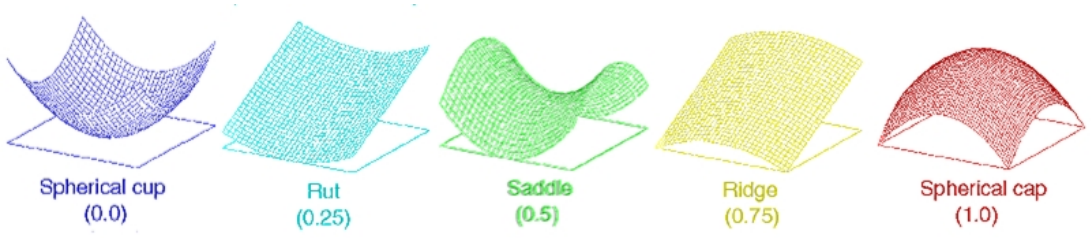


Figure 3.2: Shape index scale. A value of zero represents a spherical cup, while a value of one represents a spherical cap. This index is independent of the coordinate system.

Since the shape index is independent of the coordinate system, it is useful for finding similar points between scans from different poses. The faces shown in Figure 3.3 are examples of 2.5D face images with intensity representing the shape index. Notice that there are several consistencies between these scans of different faces. For example, the area between the eyes and the bridge of the nose is consistently *rut* shaped, as is the area around the mouth. These consistencies are used to help locate prominent anchor points within the face regardless of the orientation of the face in the scan.

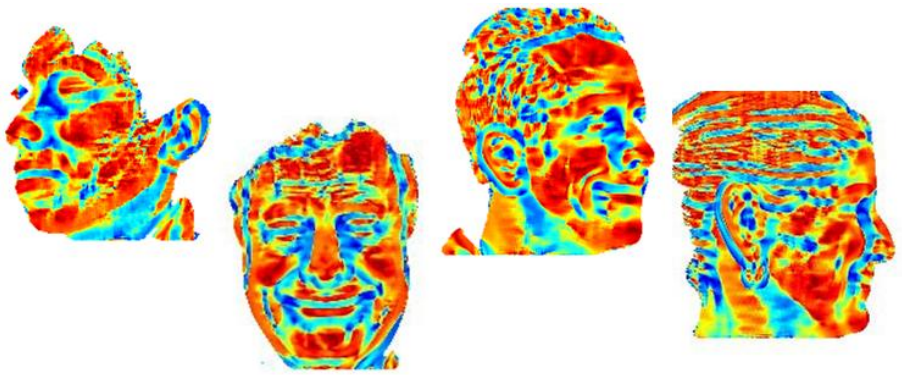


Figure 3.3: Example of the shape index over faces in different poses. Note the consistency of the values around areas such as the eyes and nose regardless of pose.

3.3 Scanner Resolution and Scale

The resolution of the scan is the number of pixels per image, while the scale can be defined as the number of pixels per unit distance. For example, the Minolta Vivid 910 scanner produces images with a resolution of 320×240, while the scale depends on how close the subject is to the scanner. In the scan shown in Figure 3.4a, there are only 31 pixels between the center of the eyes, yet the scan shown in Figure 3.4b has 70 pixels between the center of the eyes.

(a) Far Image (b) Close Image

Figure 3.4: Two scans taken with different camera offsets resulting in different scales for the face.

While the resolution of a scan is constant, the scale varies depending on the distance between the subject and the scanner. The scale may not be constant within a particular scan; for example, the distance between two neighboring pixels may differ

depending on how close points represented by the pixels are to the scanner.

Define the change in x between two horizontally neighboring points in a 2.5D depth map (see Section 2.4) to be the column step, and define the change in y between two vertical neighboring pixels to be the row step. Regions of the scan that have a larger offset from the scanner will have larger step values than regions that are closer to the scanner. The extra x and y coordinate matrices in 2.5D scans are needed in order to account for the variation in step within a particular scan. However, in an orthogonal depth map (see Section 3.6.1), the step size can be made constant so there is no need to maintain the redundant data. In the continuous form of the surface the $step_x$ is the the partial derivative of the function $x(i, j)$ in the x direction and $step_y$ is the partial derivative of the function $y(i, j)$ in the y direction. In the discrete from, the step value is calculated as a matrix with size one less than the x and y matrices, and also requires a flag value ($step_xf$ and $step_yf$) for areas where it is not possible to calculate the step. The step in the x direction is the difference between any two horizontally neighboring points on the 2.5D depth map in the x direction:

$$1 \leq u \leq cols \tag{3.28}$$

$$1 \leq v < rows \tag{3.29}$$

$$step_x(u, v) = x_{(u,v)} - y_{(u,v+1)} \tag{3.30}$$

$$step_{xf}(u, v) = flag_{(u,v)} | flag_{(u,v+1)} \tag{3.31}$$

The step in the y direction is the difference between any two vertically neighboring points on the 2.5D depth map in the y direction:

$$1 \leq u < cols \tag{3.32}$$

$$1 \leq v \leq rows \tag{3.33}$$

$$step_y(u, v) = x_{(u,v)} - y_{(u+1,v)} \tag{3.34}$$

$$step_{yf}(u,v) = flag_{(u,v)} | flag_{(u+1,v)} \tag{3.35}$$

The average step size (both column and row) represents a general scaling factor that can be used to estimate the size of different regions in the scan. This scaling factor can speed up distance calculations on a scan when only an estimate is required. For example, the bounding box regions generated for anchor point detection in Section 4 are generated by using the minimum and maximum step values to ensure that the bounding boxes properly encompass the regions of interest. If the step value is not used, then all pixels would have to be searched to determine where the bounding boxes lie (which consumes processing time).

To get an idea of the ranges of step values that are generated, the average step values for the MSU frontal dataset and the FRGC1.0 dataset were calculated. Note that the MSU dataset was taken at half the resolution of the FRGC data, which accounts for the differences in the step values. The minimum, average, and maximum step differences represent the range of step values within a particular scan. The step difference for a scan is the difference between the minimum and maximum step values over an entire scan. The "step diff" values shown in Table 3.1 represent the minimum, average and maximum step difference in the entire database.

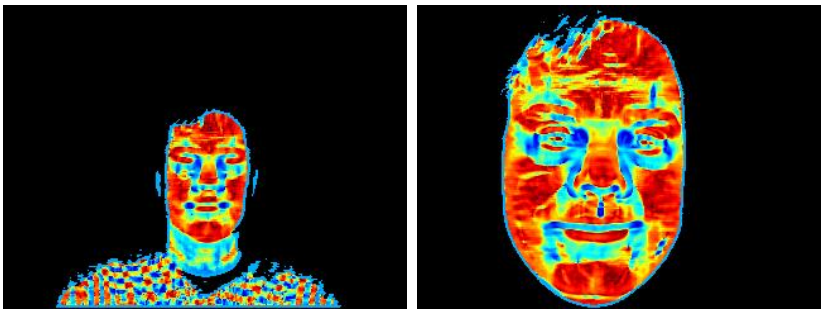
Table 3.1: Step values for standard databases.

	Cols	Rows	Minimum	Average	Max
MSU step	320	240	1.28 mm	1.29 mm	1.29 mm
step diff			0.00 mm	0.01 mm	0.04 mm
FRGC1.0 step	640	480	0.43 mm	0.80 mm	0.98 mm
step diff			0.00 mm	0.03 mm	0.06 mm

Scale is also a challenge when working with sliding window algorithms. When using a sliding window on images with different scales, it is important to note that the results may be different for different scans. An example is a sliding window algorithm used to calculate surface curvatures. In this algorithm, a cubic surface is

fit to a set of data points in the window. In one version of the algorithm, the data points are separated by a fixed window size. In the second algorithm, the data is selected based on the relative scale of the images using the step value. This relative scale allows the fitting algorithm to be applied to points at the same scale for any image.

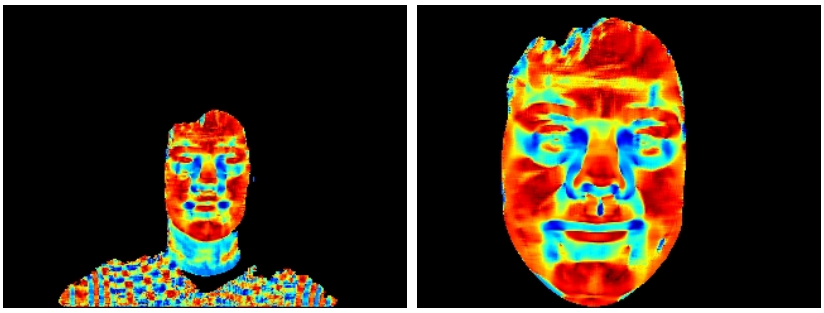
For example, consider a scan of a subject with a large camera offset (Figure 3.5a). In this case, the between-pixel distance will also be large, and a fixed window of 9×9 could represent $14mm^2$ of space. Now, consider a scan taken closer to the camera (Figure 3.5b). The same 9×9 window could now represent less than $9mm^2$ of space. This discrepancy in scale will cause variations in the curvature calculation and shape index, as can be seen in Figures 3.5 and 3.6.



(a) Far Image (b) Close Image

Figure 3.5: Fixed window (9×9) shape index calculation window. Note that the shape index values are dependent on the resolution of the scan. The detail of the curvature in (a) is much more prominent than the detail shown in (b).

Even though, Figures 3.5a/3.6a and Figures 3.5b/3.6b are of the same scan, their curvature regions differ because the curvature calculation varies at different scales. When a fixed-size window is estimated using the average image step, the curvatures look smoother. Note that using the average image step is fairly reasonable given the results of Table 3.1; however, there will still be some slight variations in regions with abnormal step values. This is especially true in regions with high drop-off values, such as the sides of the face.



(a) Far Image (b) Close Image

Figure 3.6: Fixed distance shape index calculation. Notice that the values are more similar across scale than in the fixed window mode (i.e., notice the smoother regions on the larger face).

3.4 Image Cleaning

Two types of errors are common to scanners such as the Vivid 910. The first type of error is holes in the scan (i.e., areas were the laser light is not seen by the camera). The Minolta handles holes gracefully with a flag matrix, which records zero if a hole is found and records one otherwise. Another approach to dealing with holes is to fill in the missing data. For example, a surface can be fit to a neighborhood of points around a hole as shown in Section 3.1. Ahn, *et al.* [6] present a hole filling algorithm that uses a dynamically changing window size. This allows the algorithm to reduce the smoothing effects of the algorithm by using a smaller window in high density areas and larger windows in low density areas.

The second type of scanner error is spike noise, which is common around the eyes and hair due to the scanner's inability to properly pick up the laser light reflection from these surfaces (see Figure 3.7). The light gets reflected twice causing inaccurate locations. Spike noise creates large outlying values that can cause problems in ICP algorithms, anchor point detectors, and correlation. A gradient filter can be used to eliminate spike noise by comparing the difference in z value for each point with all of its neighbors. If any of the gradients are above a certain threshold, then the flags for these noisy points are set to zero. A gradient filter set to around $5mm/pixel$ will

remove most of the spike noise in data collected with the Vivid 910.

Figure 3.7: Profile view of a frontal scan from the FRGC1.0 database. Notice the large spike noise errors around the eye.

3.5 Surface Alignment

When a scan is taken, all values are in a scanner-based coordinate system. Thus, two scans of the same object can have vastly different values if they are positioned differently relative to the camera. Surface alignment (also called registration) is defined as the calculation of a rigid transformation ($R \in \mathcal{R}$) that best transforms two scans of the same object taken at different poses or locations onto one another. The first scan is denoted by a set of points P_s and the second scan by the points P_m. The transformation is rigid and has six degrees of freedom: roll, pitch, yaw, and changes in x, y and z.

3.5.1 Single Point Rigid Alignment

If both the scan and the model are known to have similar poses relative to the scanner (i.e., both are frontal face scans with no rotation), then the transformation between the test scan and the model will only have a translation component and no rotation component. Thus, coarse alignment can be calculated using the difference between a single corresponding anchor point that is the same in the scan and the model. For

example, the 3D location of the nose tip is found in both the test scan (s_x, s_y, s_z) and the model (t_x, t_y, t_z). These anchor points can be used to generate the transformation matrix (T) to transform the test scan to the model:

$$T = \begin{bmatrix} 1 & 0 & 0 & (t_x - s_x) \\ 0 & 1 & 0 & (t_y - s_y) \\ 0 & 0 & 1 & (t_z - s_z) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The problem with a single point transformation is that it is difficult to assume that there are no variations in the pose angles. In fact, it is quite difficult to maintain exactly the same pose between two scans.

3.5.2 Three Point Rigid Alignment

If the relative pose between the scan and the model is unknown, then a single common anchor point is not enough. Instead, a minimum of three corresponding anchor points are required to calculate the transformation from the test scan to the model. For example, assume that three anchor points (inside right eye, outside right eye, and nose tip) have been identified on both the scan and the model. Then, a rigid transform $(R \in \mathcal{R})$ can be easily calculated between the sets of anchor points [141] by a least squares fitting between the triangles formed from the two sets of three points. The set of three anchor points on the test scan (a) is transformed into the same location as the second set of anchor points on the model template (p). The rigid transformation is composed of a series of simple transformations, as shown in Figure 3.8.

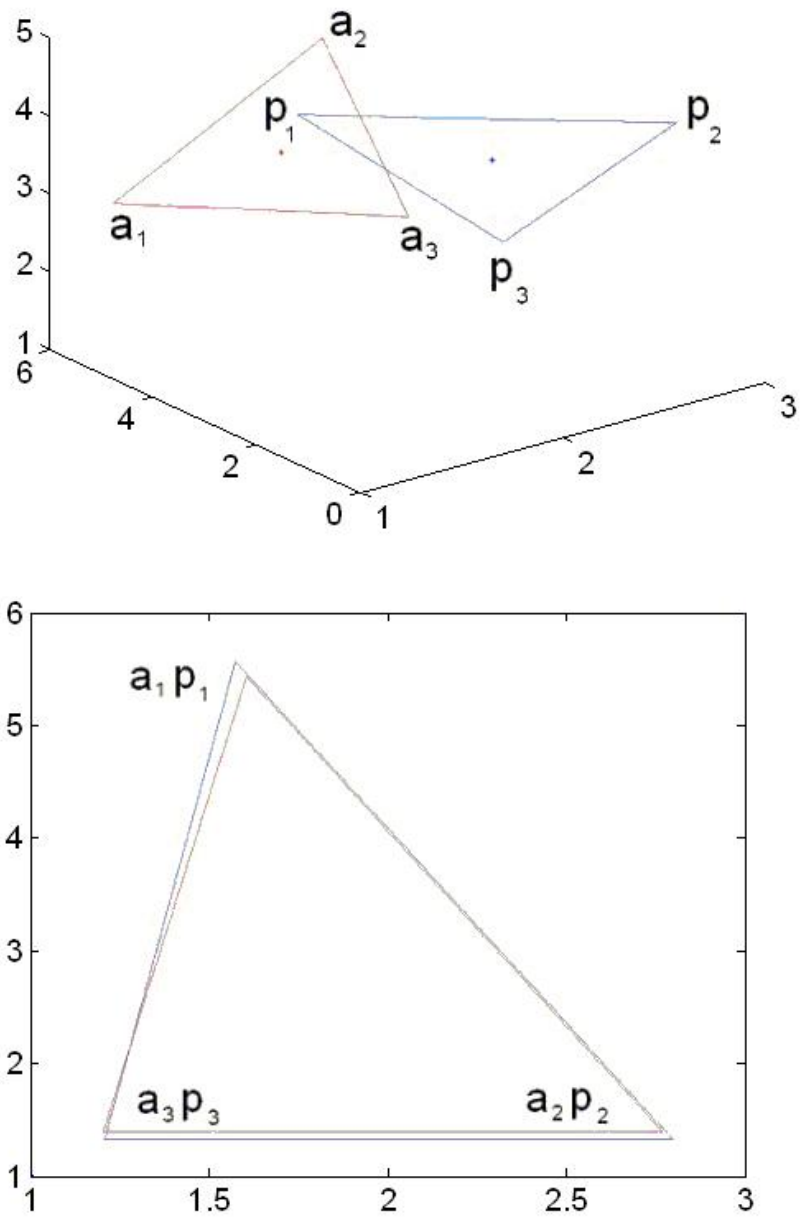


Figure 3.8: Rigid transformation between two sets of three corresponding points; (a) the original set of points (the red triangle is constructed from the a points, the blue triangle is constructed from the p points); (b) The set of points after the rigid transformation of points a onto points p.

$$R = T_{c-p} \cdot B_p \cdot \theta \cdot B_A \cdot T_{c-a}$$

where:

R	Total Rigid Transformation from anchor points t to anchor points m.
T_{c-a}	Translate points a to an origin at the center of the triangle formed by the vertices of the triangle.
B_A	Transform world coordinates into coordinates based on basis vectors of set a.
θ	Optimum rotation to align point vertices into basis vectors of set p.
B_p	Transform from p basis vectors back into original world coordinates.
T_{c-p}	Translate from the center of p vertices into the original world origin.

3.5.3 Iterative Methods

The single point and the three point alignment algorithms are closed-form alignment methods. However, these methods are very sensitive to the accuracy of the points used in the alignment. Even small errors in the detected point location or variations in pose can cause the resulting alignment to have errors. These errors tend to get larger for points that are farther away from the anchor points used in alignment.

The Iterative Closest Point (ICP) algorithm [17, 37, 152] is a fine alignment method that minimizes surface distance errors. Starting with an initial estimate of the rigid transformation, ICP iteratively refines the transformation by choosing corresponding (control) points on the 3D surface of a probe object and calculating the best translation and rotation (that minimize an error function) based on the distance to the surface of the base object.

This dissertation primarily uses the Besl ICP algorithm [17], which uses point-to-point distance and a closed-form solution when calculating the transformation matrix during each iteration. The point-to-point distances will have errors proportional to the distance between the sample points on the objects. The closer together the points are, the smaller the error. If the points are far apart, then the algorithm has difficulty

making small incremental changes and there is a higher probability of getting stuck in a local minima.

The point-to-plane distance used by Chen [37] makes the ICP algorithm less susceptible to local minima, compared to the point-to-point metric [64]. Instead of finding the closest point in a discrete sample, the point-to-plane algorithm interpolates a plane between the points on the surface. The problem with the point-to-plane algorithm is that it is more processor intensive.

These two classical ICP algorithms [17, 37] can also be run in a interleaved style, called the hybrid ICP algorithm [100, 99]. Each iteration consists of two steps, starting with Besl's scheme [17] to compute an estimation of the alignment, followed by Chen's scheme [37] for a refinement. The two different distance metrics are utilized together, which has the potential for a better registration result.

The iterative nature of all of these ICP algorithms can make them slow. The most time consuming component of the algorithms is the search for the closest point on the surface of the base object for each of the control points. If every possible point on the base object is searched, then this is an $O(np)$ algorithm, where p is the number of points in the base object (normally $p = rc$, where r and c are the number of rows and columns in a 2.5D scan) and n is the number of control points sampled from the probe object.

A KD-tree based [15, 61] ICP algorithm can improve the time it takes to calculate the point distances by ordering the points in a lookup tree. This will have a worst case running time of $O(n\ ln(rc))$, which is better than ICP alone; however, the construction of the KD-tree can take time. Another possibility to speed up point distance calculation is to use the matrix structure of the 2.5D scan to help guide the search. The ordering of the points in xy space is related to the ordering of the point index in the matrix. This relationship can help speed up the point-to-point ICP algorithm by using row and column lookup tables. For each row, the maximum

and minimum y values are recorded. Similarly, for each column the maximum and minimum x values are also stored. Then the search for the nearest point (x, y, z) involves searching these maximum and minimum tables for all possible rows and columns that could have this point, within an error distance threshold. This search table loop-up takes $O(r + c)$ processing time to execute and these rows and columns represent a much smaller search window. The search speed depends on the size of the window, which depends on the size of the resolution error threshold.

For this dissertation, an error threshold of $2mm$ is chosen (see Section 5.2.3). Since the standard resolution between pixels is around $0.5mm$, it is possible to have a search window with a radius as large as 4 pixels for every row or column (9 x 9 window). Typically, the window's radius ranges from zero (i.e., the point is off the scan) to 8 pixels (a 17 x 17 window), with an average window size around 5 x 5 pixels. The size of the window depends on the chosen error threshold and on the resolution of the scan. In any case, a search window is much smaller than searching every point in the scan and is similar to the number of comparisons required for the KD tree. The overall speed of this algorithm is estimated to be $O(n(r+n)w)$, where w is the window size. In these scans, w is much smaller than any of the other components, and can be considered constant. Thus this algorithm has a speed of $O(n(r + n))$. In practice, the speed increase can be quite significant, especially as the xy deviations of the rows and columns become smaller (such as with an orthogonal projection). However, an obvious limitation of this method is that it will only work on matrix-style 2.5D depth maps.

3.6 Orthogonal Depth Maps

The scans produced by the Vivid 910 require three matrices (x, y, and z) to describe the location of the 3D points. As described in Section 3.3, the x and y matrices are

required to account for the variations of scale within a scan. With three matrices any point on the surface of a sphere can be stored; however, due to occlusion, the 2.5D matrix is only able to show one side of an object. An orthogonal depth map is a re-projection of the 3D scan data with a constant step value. This constant step value eliminates the need for the x and y matrices. In other words, an orthogonal depth map is a re-projection of the surface data to be a function of z in terms of x and y:

$$z = f(x, y) \tag{3.36}$$

Given any 3D object in space, a depth map is produced by projecting z values onto a projection plane defined by a matrix. For each cell in the matrix, a ray is traced onto the object perpendicular to the image plane (see Figure 3.9). The distance between the cell and the first point at which the ray intersects the object is recorded in the depth matrix. The depth map cannot represent all the points on a sphere, but rather only a hemisphere.

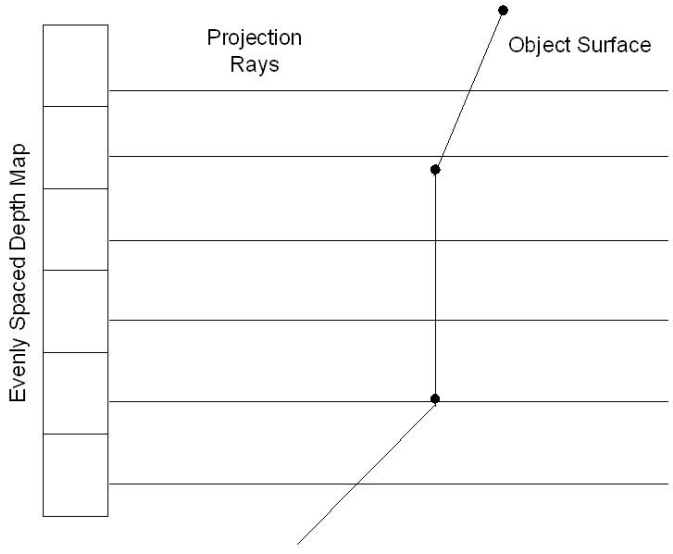


Figure 3.9: Visual representation of an orthogonal projection.

In computer graphics the z-buffer algorithm is commonly used to generate an

orthogonal depth map. For each polygon in a mesh, the coordinates of the polygon are projected onto the orthogonal matrix. If a new projection overlaps an existing point on the matrix, then the minimum of the two points is stored (this overlap happens during occlusion). Orthogonal depth maps are easy to calculate and they normalize the data in the x and y directions. Another advantage of orthogonal depth maps is that many graphics processors are optimized to execute the z-buffer algorithm and this optimization could be used to speed up depth map calculations.

3.6.1 Virtual Scans

Since it is not generally practical to scan a subject at all possible rotations and angles, orthogonal depth maps are used to generate virtual scans of an object. The virtual rotation process rotates the points of a face scan in 3D space and then re-projects the points onto an orthogonal plane that is parallel to the xy plane. The orthogonal projection produces a virtual scan that includes a depth map, flag map and color image similar to the data produced by the Minolta scanner. Exact pose angles can be produced and control can be maintained over the sampling rate. The frontal image in Figure 3.10(c) is the true scan, while the other four images in Figure 3.10 are artificially rotated and re-sampled images.

Figure 3.10: Example of virtual yaw rotation.

3.6.2 Super Sampling

Another advantage of the orthogonal depth map is that it can be easily sampled at a higher rate than the original scan (see Figure 3.11). This super sampling is achieved by interpolating values between the points on the original scan. Super sampling allows for a much richer and more dense point space and can improve the performance of the ICP algorithm by avoiding local minimums. Super sampling the data allows it to achieve faster ICP performance with a point-to-point algorithm while achieving the quality of a point-to-plane ICP algorithm (see Section 3.5.3).

(a) Original (320×240) (b) (200×350) (c) (400×700)

Figure 3.11: Depth maps sampled at two different resolutions.

3.7 Summary

This chapter developed and presented various tools for processing 3D data. These approaches are fundamental to the methods described in the rest of the dissertation. Section 3.1 described methods for fitting a cubic surface to a sample of 3D points. Surface fitting is used in Section 6.2.3 to help establish a robust face based coordinate

system. This surface fitting method is also used to to calculate the surface curvatures, surface normals and shape index. The shape index described in Section 3.2 is a pose invariant feature space used in anchor point detection in Chapter 4. The step values described in Section 3.3 are used to establish the relative scale of a 3D scan and are used as a standard way to estimate point locations and distances throughout the dissertation. For example, in Section 4.3 step values are used to estimate the size of the anchor point search windows (bounding boxes), and in Section 5.2.1 the step size are used to help bound the location of the ICP control point region. Section 3.4 describes methods for cleaning 3D scans, which are necessary to eliminate spike noise in the data. Section 3.5 discusses simple methods for establishing surface alignment. These general 3D alignment algorithms form the foundation of the 3D face alignment methods used throughout this dissertation and are described in detail in Chapter 5. This Chapter concluded with a description of an Orthogonal depth map that is used in Chapter 6 as a way to standardize the resolution of the Canonical Face Depth Map.

Chapter 4

3D Anchor Point Detection

Chapters 2 and 3 introduced methods for collecting and working with 3D face data. Chapter 2 described 3D notation, modeling and scanners. Chapter 3 described some foundational algorithms for working in 3D, such as surface fitting, noise cleaning and orthogonal depth maps. This chapter discusses anchor points, which are used in many algorithms to ground the location of key facial components. In the context of this dissertation, anchor points are points on the face that are common to all faces and that can be detected reliably in different facial scans. Some examples of anchor points are shown in Figure 4.1.

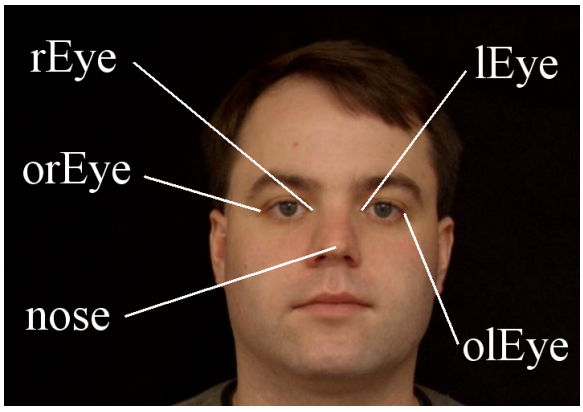


Figure 4.1: Example anchor point locations. rEye - Inside of the right eye. orEye - Outside of the right eye. lEye - Inside of the left eye. olEye - Outside of the left eye. nose - Nose tip.

This chapter outlines methods for detecting key anchor points in 3D face scanner data. These anchor points can be used to estimate pose and match the test image to a 3D face model. Two algorithms are presented for detecting face anchor points in the context of face verification; one for frontal images and one for arbitrary pose. A 99% success rate is achieved when finding anchor points in frontal images with neutral expressions, and 86% success is achieved for scans with variations in pose and expression. These results demonstrate the challenges in 3D face verification when dealing with arbitrary pose and expression.

Anchor points are used by many algorithms to align and manipulate 3D data from different scanners. For example, Chapter 5 presents a method for using common anchor points (such as the tip of the nose) to coarsely align faces. Anchor points such as the nose tip and eye corners are also used to localize regions of the face that do not vary much with expression. These regions are used to select control points for the ICP algorithm.

Section 4.1 provides a general description of the anchor point detection problem. Section 4.2 examines existing 2D anchor point detection systems, while the remainder of this chapter discusses the detection of anchor points in 3D for both frontal face scans (Section 4.3) and scans of faces with arbitrary pose (Section 4.4).

4.1 Problem Description

Viola and Jones [138] automatically learn anchor points from manually labeled data. However, this method tends to refer to regions on the face (rather than actual points) and does not meet the accuracy requirements for many algorithms, such as alignment, which require the location of specific anchor points rather than regions. Using saliency, robustness, and utility as the three main criteria, the core anchor labels shown in Figure 4.1 were selected for study in this dissertation.

Saliency means that an anchor point is prominent and easily identifiable in most face scans. For example, the eye corners and nose tip can be used as anchor points on almost every face, and are within the accuracy required for coarse alignment of the 3D face alignment system (see Section 5). In addition to being salient, it is also important for anchor points (or sets of anchor points) to be robustly identifiable across all types of images. For example, the center of the pupil is a commonly used anchor point because it is salient and can be quickly identified within an image [153]. However, depending on the pose, it is possible for a scan to have only one (profile view) or even zero pupils (eyes closed) visible for detection. Thus, it is desirable to find anchor points that are robust across variations in lighting, facial pose, and facial expression. Currently, the MSU database (described in Section 2.5.1) contains variations in neutral or smiling expressions and open or closed eyes.

A third consideration for selecting anchor points is their utility, which is based on the algorithm that will be using the anchor points. For the coarse registration problem (see Chapter 5), anchor points that are farther apart tend to work better than ones closer together. The ICP algorithm (see Section 3.5.3) works best with non-deformable points (i.e., points that do not change with respect to each other). Thus, points such as the mouth corners are not a good choice for use with ICP because they deform as expression changes.

The five core anchor points labeled in Figure 4.1 are easy to find within the image, are stationary with respect to the skull (and therefore do not move much with respect to each other), span the width of most images, and finding them does not require that the eyes be open. There is still the problem of occlusion of specific points due to pose, but the coarse alignment algorithm discussed in Chapter 5 only requires three of the five possible points in order to achieve rigid alignment. Due to this redundancy, the five anchor points in Figure 4.1 are sufficient for most images, including profiles. These core points are also sufficient to calculate the location of the grid of control

points used by ICP (see Figure 5.29) for surface alignment in Chapter 5.

If fewer than three of the five core anchor points are present, then the set will not be sufficient to calculate the transformation between two face scans. A common example of this problem is in a full profile scan where the eyeball occludes the inside corner point of the eye. In such cases, additional labeled anchor points are needed to increase the likelihood of finding three usable points. For this reason, four additional anchor point labels are used: the two corners of the mouth, and a pair of non-surface points that represents the central axis of the scan. Depending on the pose of the scan, these secondary points may or may not be used as the three transformation points.

The corners of the mouth are normally avoided because they can change with changes in expression. However, they are included in the secondary point list because they are salient and can help in cases where the other anchor points are unavailable. The other secondary points, the centroid and the top-centroid, represent the 3D vertical axes of both the scan and the model. The centroid is calculated as the average location of all the points in a scan or model, and the top-centroid is the same point moved 100mm in the z-direction. In most scans these points do not align well with the models. However, due to their similar structure and orientation, they have a tendency to be useful points because it is simple to assume that most faces are vertical and most of the transformations are just rotations about the vertical axis (with a small translation to account for slight variations in the vertical axis).

More candidate anchor points could be added in the future; for example, a point on the ear was considered. However, it is difficult to detect a reliable point on the ear due to self-occlusion and occlusion due to hairstyle. Although adding more anchor points increases the chances of finding at least three good anchor points, it also requires more computational resources to find and label such additional points.

4.2 Background

Identifying anchor points for 2D face recognition is a well studied problem, and Table 4.1 overviews several automatic anchor point detection systems that work with 2D faces. Most of these systems fall under two areas of research: face detection and pose estimation. Face detection is the problem of determining whether one or more faces exist within an image, which is a necessary precursor to any vision system that works with faces. Pose detection analyzes an image to determine in which direction a face is looking [10, 147].

Many types of algorithms have been developed to find faces in 2D images. Some of the faster methods learn statistical anchor points in the face region and exploit these anchor points for detection [138]. For example, the area around the eyes can be found by using a differentiation filter that looks for a dark-light-dark region, indicating the dark eyes and a light area around the bridge of the nose (see Figure 4.2). This approach does not identify a particular anchor point, but instead determines a pattern of relative intensities. This is reasonable for 2D face detection algorithms because they do not need to be constrained to find exact anchor points in real space. This method of face detection is used in [12] to identify the locations of faces in real time. Then, a set of Gabor filters is trained using Adaboost to determine up to seven expressions. Then these "mood" measures are fed into some robots to enhance their interaction and performance. For a good review of 2D face detection literature, see work by Yang *et al.* [146], or Hjelmas [71].

The pose detection literature is mostly related to the problem of anchor point detection in 2D face images. This problem naturally lends itself to identifying specific anchor points, such as the eyes and the nose [10, 147]. Pose information is commonly used with algorithms that attempt to fit a face image texture to a wire frame model of the face [21, 143]. In these algorithms, anchor points are more useful than regions.

Another approach to anchor point detection is to collect heuristics about common

Table 4.1: Automatic face anchor point detection literature overview.

Papers	2D	2.5D	Pose Variation	Expression Variation**	Lighting Variation	Notes
(Ballard, Stockman) [10]	√		Some* 20°	√		Tracked anchor points on the face as an alternative computer control.
(Bakic, Stockman) [9]	√		Some* 20°		√	Tracked anchor points in 2D face video as an alternative computer control.
(Yilmaz) [147]	√		Some* 20°	√	√	Tracked anchor points on the face as an alternative computer control.
(Viola, Jones)[137]	√		Some*	√	√	Face detection algorithm. Learns regions instead of points. The exact location of the face is unknown.
(Blanz, Vetter)[21] (Xiao, Baker, Matthews Kanade)[143]	√		Some*	√	√	Attaches 2D and 3D mesh models to 2D face video.
(Brunelli, Poggio)[29]	√		Some*		√	Surveys the trade-off between anchor point and template models in face recognition.
(Hsu, Jain)[75, 74, 73]		√				Uses Snakes to fit a deformable 3D model to the head.
(Wang Chua Ho)[139]	√	√	Some*	√		Limited database size but can handle some pose variations.
(Conde, Cipolla)[48]	√	√	Some*	√		Slow.
System developed in this dissertation		√	√	√	√	Designed to work under variations in pose, expression and lighting.

∗ Most of these systems can handle small variations in pose angle. These systems tend to require that the entire area of both eyes be visible in the image.

∗∗ The methods without checks were tested with a database that did not vary in expression. These systems will probably work with some degree of facial variation.

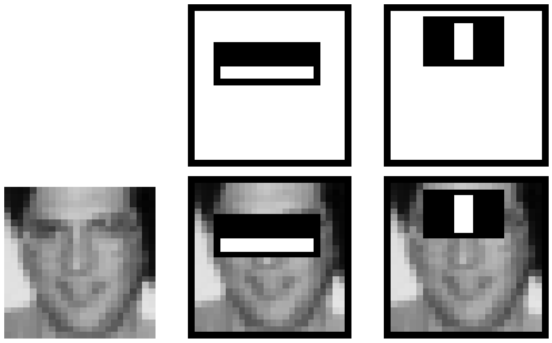


Figure 4.2: Simple and efficient 2D face detector using a cascade of dark-light-dark anchor point detectors [138].

facial anchor point locations, and then systematically exploit the heuristics to find the anchor points. For example, face color is a common attribute used to segment the face region and then identify the dark regions within the face that correspond to eyes, eyebrows and an open mouth. Simple models of the face are also used to guide heuristic searches for anchor points [29, 9]. For instance, knowledge about the shape of a face can be used with Snakes to find the outline of the head and the eyes. Hsu and Jain [75] applied Snakes to frontal 2.5D images to accurately determine the outline of the head. However, as with many face detection and pose estimation algorithms, some assumptions were made about the pose angle (frontal faces only) in order to reliably detect the face anchor points.

In addition to the work presented in this dissertation on 2.5D anchor point detection [45], there are other frontal anchor point detectors [74, 139, 23]. These frontal anchor point detection algorithms each have different requirements, such as the eyes being open or the face having a neutral expression. Although ideally no assumptions about the pose should be made, imposing some simple restrictions on pose greatly reduces the computational requirements for anchor point detection in 2D and 2.5D.

4.3 Frontal Anchor Point Detection in 3D

This section describes the frontal anchor point detection algorithm outlined in Table 4.2. The first step is to remove spike noise from the scan using a gradient filter, as described in Section 3.4. The second step is to find a base reference point, such as the top of the head. (Actually, any point near the top of the head should do because this point is only used to establish the vertical location of the head.) Once the top of the head is found, a bounding box for the location of the nose can be produced based on the inter-subject distance model shown in Table 4.3. The frontal anchor point detection algorithm then uses other models to generate bounding boxes (step 4) and uses local feature-based methods to find the remaining anchor points (step 5).

Table 4.2: Frontal pose anchor point detection algorithm.

1. Use the gradient filter to filter out depth spikes.
2. Find the top of the head and generate nose bounding box.
3. Find the tip of the nose as the closest point of the scan.
4. Calculate the bounding box for the inside corners of the eye, mouth, and chin.
5. Calculate the location of each individual anchor point using the shape index.

The first step in the 3D frontal anchor point detection algorithm relies on a gradient filter. When dealing with frontal pose faces, the z gradient is defined as the change in the z distance with respect to the x and y directions. Typically there is a large change in z when moving from the base of the nose to the tip of the nose, and from the boundary of the face and hair. However, in most other locations on the face the z gradient is small. It is common for spike noise to appear in the scans (especially in the FRGC dataset) due to errors in reading the reflected laser light. This spike noise is easily removed by filtering out points that have a gradient larger than the gradients of points around the nose (see Section 3.4). Filtering the scan gradients to eliminate spike noise before selecting the nose point increases the success rate of the nose detector from 95% to 97%. If the search is limited to areas that may reasonably contain the nose (see Table 4.3), the success rate can be increased to 99%.

This success rate is sufficient for a frontal-based identification system since, when the orientation of the head is known, one anchor point is sufficient to initially align the head with the model (see Chapter 5). However, in order to allow for minor variations in the pose angle (± 5 deg) other anchor points on the face are needed.

For the second step in the anchor point detection algorithm, the top of the head is detected by simply searching from the top down on the scan. This is a naive detector that works quite well, as long as there are not other objects in the range of the scan located above the head. The top of the head is used as a reference point to reduce the search area for the nose point detector by creating a bounding box to define the search area. The size of the bounding box is determined based on a statistical model of the face. The statistical model identifies where the anchor points may be located on the face relative to other anchor points. To create this statistical model, the distance between each pair of anchor points was measured in a database of over 2000 test scans. The minimum, maximum and average distances were included in the statistical model and used to bound the search region for each of the anchor points (see Table 4.3).

Table 4.3: Statistical model of the face. Each set of distances are calculated from point A to point B only in the specified direction.

Point A	Point B	direction	Minimum (mm)	Average (mm)	Maximum (mm)
Top of Head	Nose Tip	vertical	84.0	121.7	181.0
Nose Tip	Mouth	vertical	16.9	34.4	46.3
Nose Tip	Chin	vertical	49.7	70.6	95.4
Nose Tip	Nose Bridge	vertical	21.2	32.2	48.4
Nose Tip	Inside Eye	horizontal	13.2	17.5	23.6
Inside Eye	Outside Eye	horizontal	15.9	30.6	39.7

Creating bounding boxes can greatly reduce the time needed to search for a particular anchor point. For example, once the tip of the nose is found, bounding boxes can be generated to locate many of the surrounding anchor points, such as the eye corners or chin (see Figure 4.3). In addition to increasing the efficiency of the anchor

point detection algorithm, bounding boxes can also improve the chances of accurately detecting an anchor point because the search is now limited to a region where the anchor point is most likely to be found. Reducing the search area reduces the chances of finding a local minimum.

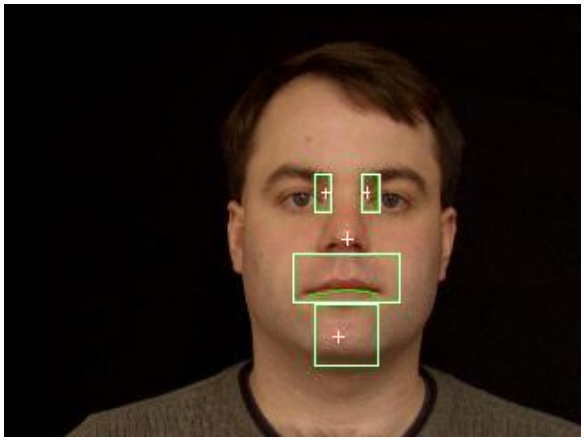


Figure 4.3: Example bounding boxes produced by the statistical face model and the location of the nose tip.

4.3.1 Experiments and Results

The frontal anchor point detection algorithm described in the previous section was tested on the following datasets:

- Dataset A - The primary dataset, composed of 111 test subjects with approximately three frontal scans each (two with neutral expression and one smiling) for a total of 330 test scans. Scans have little spike noise and most faces have a neutral expression. See examples in Figure 4.4.

- Dataset B - 953 scans of 275 subjects taken from a database provided by The University of Notre Dame. These scans are all frontal neutral expression with higher levels of noise than Set A due to different lighting conditions [118].

- Dataset C - Limited dataset containing 49 images with a handful of subjects. Explores significant variations in the data, such as background changes, rota-

tions, occlusions, expression, lighting and noise. Most of these images do not satisfy the specifications of the 3DID face verification system (see examples in Figure 4.5).

Figure 4.4: Detected anchor points for example images from dataset A.

Table 4.4: Experimental results for frontal anchor point detector.

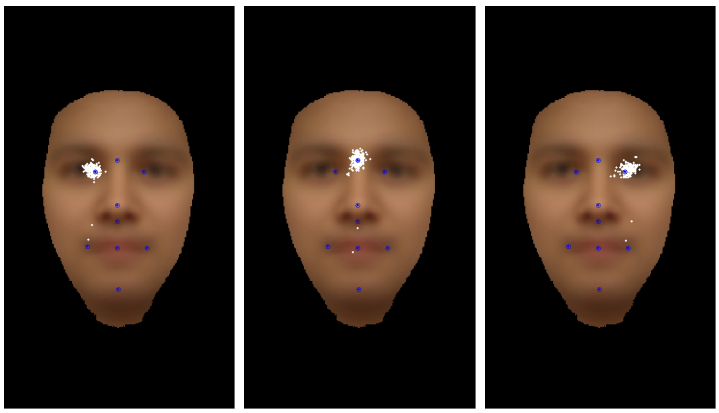
Point Name	Average (mm)	Standard Deviation
Nose	4.1	5.1
Nose Bridge	4.7	6.0
Right Eye	5.5	4.9
Left Eye	6.3	5.0
Chin	11.0	7.6
Nose Base	4.9	6.6
Mouth	4.0	6.7
Left Mouth	6.7	9.3
Right Mouth	6.9	8.6

Table 4.4 lists the mean and standard deviation between the automatically selected anchor points and the manually selected anchor points. Figure 4.6 plots a set of manually selected anchor points onto the average face to represent the type

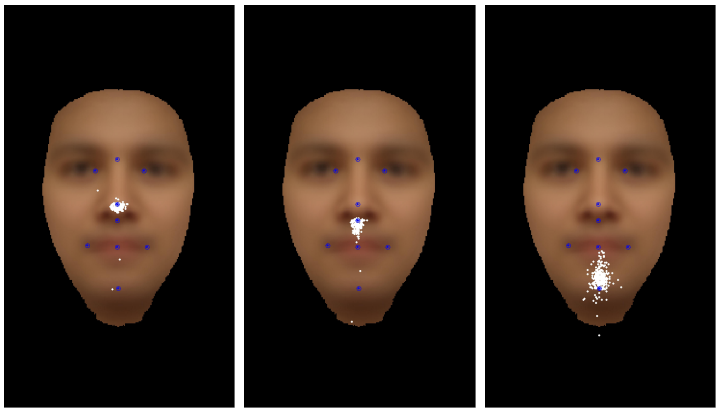
Figure 4.5: Detected anchor points for example images from dataset C (these images violate the specs of the verification system).

of errors found in the dataset. Each of the white dots correspond to the location of the automatically selected anchor points relative to the manually selected points. Figure 4.6d represents the nose point precision. There are three outliers in the entire dataset, two of them are due to limitations in the nose point detector (the head is pitched back and the subject has a large lip) and the third outlier is due to head motion during scanning.

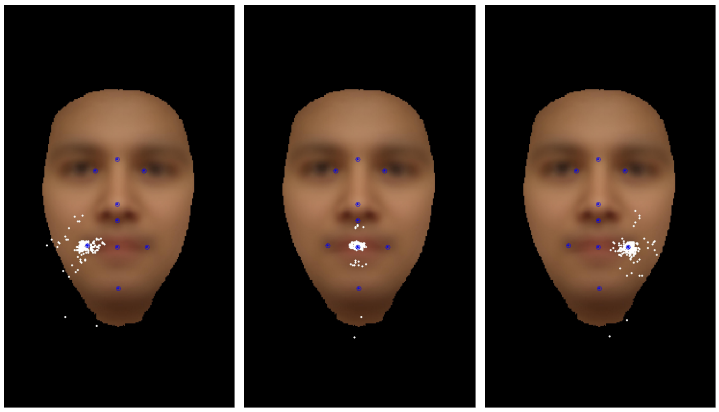
The anchor point detection algorithm described in Table 4.2 was tested on all three datasets to see how accurately the algorithm found anchor points on the face. The anchor point detector was considered successful if it correctly located three of the five anchor points shown in Figure 4.1. Success was a qualitative judgment made by the investigator and verified by checking the convergence of the ICP algorithm. These results are summarized in Table 4.5. Of the five anchor points detected, the worst performance was with the mouth corners due to variations in expression; therefore the success rate for the mouth anchor points is also reported as a reference. Figure 4.6 plots the relative difference between the manually selected anchor points and



(a) Right Eye (b) Nose Bridge (c) Left Eye



(d) Nose (e) Nose Base (f) Chin



(g) Right Mouth (h) Mouth (i) Left Mouth

Figure 4.6: Average face image with each of the manually selected anchor points (green) and the relative location of the automatically generated points (white) from the MSU 330 image dataset.

Table 4.5: Experimental results for successful frontal anchor point detector.

Data Set	Number of Scans	Approximate Number of Subjects	Detection Rate	Detection Rate Mouth Only	Average Algorithm Time (sec)
A	330	110	99.1%	97.0%	0.33
B	953	275	99.8%	97.4%	0.30
C	49	10	85.7%	83.7%	0.33

the automatically generated anchor points. The anchor point detector was run on a Pentium 4 CPU with two 3.20GHz processors and 1.00GB of RAM. All of the times reported are wall clock times and will vary depending on the load of the cpu, this load variation being most noticeable during file loading.

4.4 General Pose Anchor Point Detection in 3D

The previous section described a frontal anchor point detection algorithm sufficient for dealing with cooperative subjects in a constrained environment. This section examines the more difficult problem of detecting anchor points under conditions of arbitrary pose. Instead of developing specific heuristics to determine the anchor points, redundant methods are used to find candidate anchor points and then these anchor points are examined to identify the best ones.

The goal of the arbitrary pose anchor point detection algorithm is to find three labeled points on the test scan that correspond to three labeled points on the 3D face model. These three pairs of points are then used to calculate a transformation from the test scan to the model in the database. This transformation coarsely aligns the test scan to the 3D model, and then the ICP algorithm is used to calculate the best matching distance between the scan and the model using a set of control points defined by the anchor points (see Figure 5.29). Once the 3D model and the test scan are aligned, the remaining anchor points can be found on the test scan by back projecting them from the 3D model.

Sections 4.4.1 and 4.4.2 describe the three major steps in aligning a test scan to a model with significantly different pose. Section 4.4.3 presents data from experiments with the arbitrary pose anchor point detection algorithm.

4.4.1 Candidate Point Selection

When pose is unknown, it cannot be guaranteed that the user is facing the scanner. Since anchor points need to be detected robustly regardless of the pose of the face, the arbitrary pose anchor point detection system uses a set of model-based methods to locate each candidate point. Surface curvatures (shape index), as described in Section 3.2, are used as a pose invariant feature to help guide the search for anchor points. A similar approach was applied by Beumier and Acheroy [18] to find anchor points on the profile ridge of a 2D profile image. However, it can be difficult to identify specific anchor points based on the shape index alone; for example, finding a region of high curvature does not necessarily result in a single point. Thus, the general locations of the anchor points are determined by combining information about the shape index with information from a model of the face. Once the general location of the anchor point is found, the results are further refined by finding a more specific point. This is normally done using an edge map generated from the 2D color component of the scan. The edge map is used to find a specific edge point (such as a corner point) closest to the general anchor point location.

Because only the three best anchor points are chosen for use in the arbitrary pose alignment algorithm, spurious candidate points are acceptable. This simplifies the development of individual anchor point detectors because possible anchor points can be selected by simple criteria and eliminated later by the constraint relaxation algorithm [76]. This makes the anchor point detection system flexible in special cases, such as when the face is occluded, because anchor point detectors can be added to account for special situations.

Inner Eye Detection

The easiest anchor point to identify in scans with arbitrary pose is the inside edge of an eye next to the bridge of the nose. This inner eye point has a shape index value that is close to zero, and the area around this point has a consistent shape index value across all face images and poses. An averaging mask (size 20 high $\times$ 10 wide) can be used to identify this area in the shape space (see Figure 4.7). More complex masks were considered, but the averaging mask proved to be robust across face scans.

The averaging mask locates a point with the largest neighborhood average *cup* shaped curvature. The inside edge of the eye is then used as an anchor in order to identify other anchor points within the face image. Because there are two eyes, the two largest regions of high *cup*-shaped curvature are selected. Often, this results in a point within the ear or hair being classified as a possible eye (see Fig 4.7b); this error is tolerable as long as the constraints can eliminate these points later.

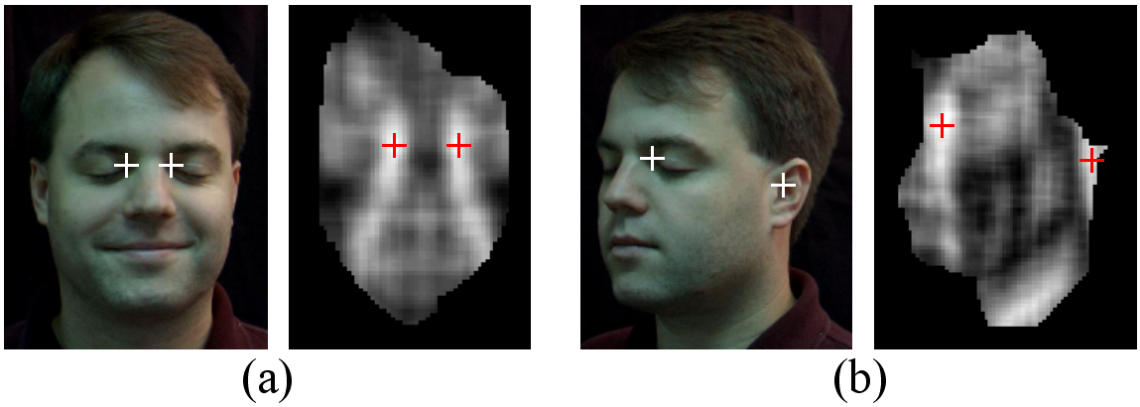


Figure 4.7: Correlation of the shape space with a simple vertical mask to find the inside points of the eyes. (a) frontal view (b) a semi-profile view. The bright spots in the scan represent the locations of the inside of the eye next to the bridge of the nose.

Outer Eye Detection

Once the inner eye candidate points are found, the outside of the eye is detected by following the *rut* (defined by the shape space) that consistently runs along the bottom

of the eye (see Figure 4.8). This calculation is made from all possible eye candidates. This particular detection method is less reliable than the other methods because the outside of the eye is flatter (in the shape space) and is not as salient as the inside eye.

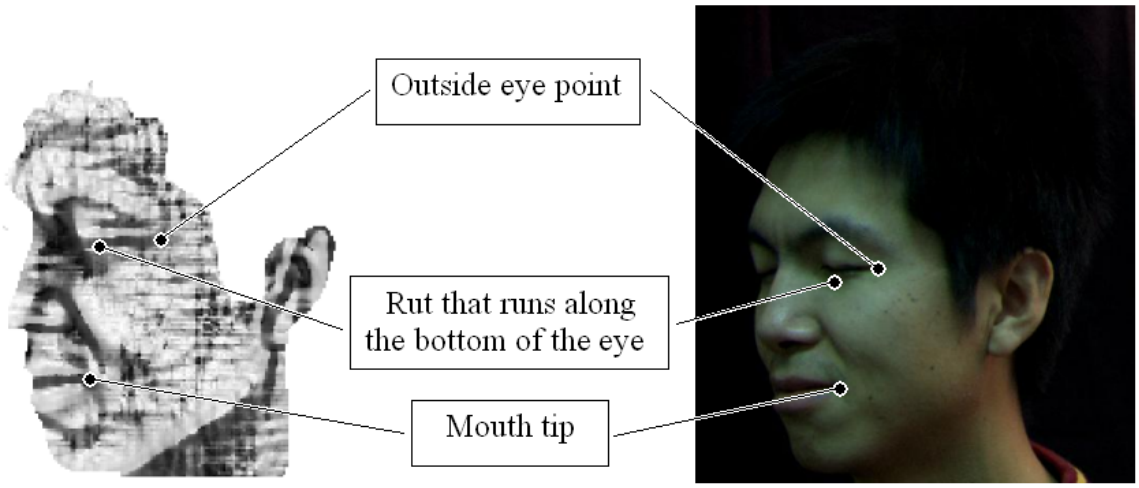


Figure 4.8: An example shape-space scan with key anchor points identified.

Nose Tip Detection

Like the inner eye point, the nose tip is generally easy to detect. For most people, the nose tip sticks out beyond other points of the face and is salient in most poses. For example, Figure 4.9 shows a face scan where up to six different simple detection methods (outlined in Table 4.6) were used to find the nose tip. The expectation is that one of the points produced by these six methods is in fact the tip of the nose. In order to evaluate the quality of these detection methods, over 1000 face scans were analyzed and the minimum distance to the actual nose tip was recorded. The average minimum distance was 5.1mm, with a standard deviation of 8.3mm. Experiments have shown that any point on the nose within a 15mm radius from the tip is an adequate starting point for the ICP algorithm. 15mm may seem like a large tolerance, but the goal of the arbitrary pose anchor point detection algorithm is to come up with a coarse alignment that can then be passed to ICP for fine alignment. Most of the nose points

detected by the algorithm had close to zero error; however, there are still a few cases where the algorithm missed the nose tip completely. In these cases, there is still a possibility that the relaxation algorithm (discussed in Section 4.4.2) will find a better set of three anchor points that does not include the nose tip.

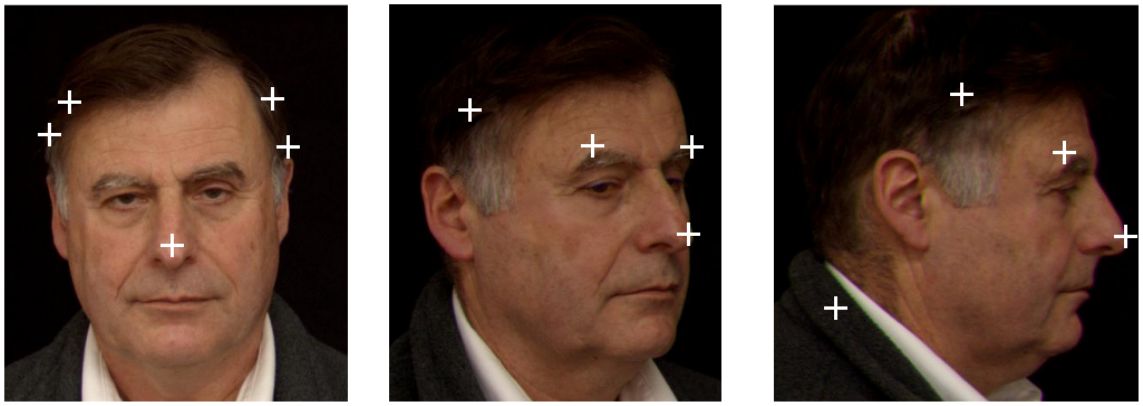


Figure 4.9: Example nose candidate points for three different poses. Notice that many of the points are errors, but in all of the scans at least one of the candidate points is actually the nose tip.

Table 4.6: Methods for finding candidate nose points.

1. Point closest to the scanner (minimum z direction point).
2. Point farthest to the left (minimum x direction point).
3. Point farthest to the right (maximum x direction point).
4. Point farthest from the vertical plane formed by points 1 and 2.
5. Point farthest from the vertical plane formed by points 1 and 3.
6. Point with the largest shape index.

4.4.2 Relaxation / Search Algorithm

Section 4.4.1 described how model-based detection methods are used to identify candidate points that may prove to be anchor points for the scan. This section describes the next step for identifying anchor points, which is to examine the set of candidate points to to find possible sets of three anchor points and their associated labels. For example, even if only three candidate points are found, each of these three points could be assigned up to nine labels (Nose Tip, Inside Right Eye, Inside Left Eye, Outside

Right Eye, Outside Left Eye, Left Mouth Corner, Right Mouth Corner, Centroid, and Top Centroid), which results in 504 possible labellings. When more candidate points are available, the total number of labellings goes up exponentially. Therefore, an exhaustive search of all candidate points with all possible labellings is not feasible. Instead, some simple constraints are developed to help filter the labeling possibilities.

There are many methods for solving constraint satisfaction problems [11]. Relaxation filtering was chosen because it is easy to implement and is efficient given that there is a limited number of candidate points as well as a limited number of possible labellings. The algorithm presented here is called discrete relaxation [76]. There are two components to discrete relaxation: the relaxation matrix and the relaxation constraints. Using a well-formulated combination of the relaxation constraints, the relaxation algorithm is able to iteratively eliminate impossible labellings using the following rule.

> **Relaxation Rule** - If a particular label assignment for any point fails ALL the constraints, then this labeling is not possible and the corresponding row and column in the relaxation matrix should be changed to zero.

This relaxation rule follows the philosophy of least commitment where only the obviously wrong labels or points are eliminated.

Relaxation Matrix

The relationship between the candidate anchor points and the possible anchor point labels is represented as a Relaxation Matrix (see Table 4.7). A value of one in the relaxation matrix represents the possibility of a point having a particular label. A value of zero means the point cannot have that particular label. The null label is used for points that do not match up with any label. The matrix is initialized with

all ones because, without any knowledge from the constraints, every point could have every possible label (note that this is before the unary constraints are applied).

Table 4.7: Initial relaxation matrix.

	Possible Labels									
	nose	rEye	lEye	orEye	olEye	rMouth	lMouth	topCentroid	centroid	null
point1	1	1	1	1	1	1	1	1	1	1
point2	1	1	1	1	1	1	1	1	1	1
point3	1	1	1	1	1	1	1	1	1	1
point4	1	1	1	1	1	1	1	1	1	1
point5	1	1	1	1	1	1	1	1	1	1
point6	1	1	1	1	1	1	1	1	1	1
point7	1	1	1	1	1	1	1	1	1	1
point8	1	1	1	1	1	1	1	1	1	1
point9	1	1	1	1	1	1	1	1	1	1

Relaxation Constraints

For the 2.5D face arbitrary pose anchor point detection problem, the following four constraints are used: shape constraint, absolute distance constraint, relative distance constraint and relative cross-product constraint.

Shape Constraint - The shape range of a candidate point with a specific label must fall between a minimum and maximum value. Note that the shape index is a unary constraint and only needs to be applied once when the registration matrix is being initialized.

$$\begin{gathered} S_{min} < S_L < S_{max} \\ \text{Example: } 0.7 < S_{nose} < 1.0 \end{gathered} \tag{4.1}$$

Absolute Distance Constraint - The distance between any two labeled points

must be between a minimum and maximum value.

$$\begin{gathered} D_{min} < |P_{L1} - P_{L2}| < D_{max} \\ \text{Example:} \\ 30mm < \sqrt{(P^x_{nose} - P^x_{rEye})^2 + (P^y_{nose} - P^y_{rEye})^2 + (P^z_{nose} - P^z_{rEye})^2} < 140mm \end{gathered} \tag{4.2}$$

Relative Distance Constraint - The distance between a single component ($x, y,$ or z) of any two labeled points must be between a minimum and maximum value.

$$\begin{gathered} D_{min} < (P^x_{L1} - P^x_{L2}) < D_{max} \text{ or} \\ D_{min} < (P^y_{L1} - P^y_{L2}) < D_{max} \text{ or} \\ D_{min} < (P^z_{L1} - P^z_{L2}) < D_{max} \\ \text{Example: } 10mm < (P^x_{rEye} - P^x_{lEye}) < 100mm \end{gathered} \tag{4.3}$$

Relative Cross-Product Constraint - The cross product between the vectors formed by (P_{L1}, P_{L2}) and (P_{L1}, P_{L3}) has a particular direction within a single ($x, y,$ or z) component.

$$\begin{gathered} \Big[(P_{L1} - P_{L2}) \times (P_{L1} - P_{L3})\Big]^x \cdot Dir > 0 \text{ or} \\ \Big[(P_{L1} - P_{L2}) \times (P_{L1} - P_{L3})\Big]^y \cdot Dir > 0 \text{ or} \\ \Big[(P_{L1} - P_{L2}) \times (P_{L1} - P_{L3})\Big]^z \cdot Dir > 0 \\ \text{where } Dir = \pm 1 \\ \text{Example: } \Big[(P_{L1} - P_{L2}) \times (P_{L1} - P_{L3})\Big]^z \cdot Dir > 0 \end{gathered} \tag{4.4}$$

These four constraints were derived by qualitatively looking at example scans and models and determining the relative distances between anchor points. Similar constraints can be added to speed up the algorithm; however, the four constraints described in this section proved to be sufficient for identifying possible labellings. One challenge in identifying these constraints was to not make too many assumptions

about the relative locations of the anchor points. For example, the left eye points are not always located to the left of the nose; in some profiles, the left eye is actually to the right of the nose.

Relaxation Example

The relaxation algorithm iteratively applies all of the constraints to the Relaxation Matrix until the Relaxation Matrix achieves a steady state. The resulting Relaxation Matrix should contain fewer ones than the original initial state (see Table 4.8).

Table 4.8: Relaxation matrix after first relaxation.

	Possible Labels									
	nose	rEye	lEye	orEye	olEye	rMouth	lMouth	topCentroid	centroid	null
point1	1	0	0	1	0	0	0	0	0	1
point2	0	0	0	0	0	0	0	0	0	1
point3	0	1	1	1	0	0	0	0	0	1
point4	1	0	0	1	0	0	0	0	0	1
point5	0	1	1	0	0	0	0	0	0	1
point6	1	0	0	0	0	0	0	0	0	1
point7	0	0	0	0	0	0	1	0	0	1
point8	0	0	0	0	0	0	0	1	0	1
point9	0	0	0	0	0	0	0	0	0	1

Note that in this example, candidate Points 2 and 9 have all zeros across their rows (except for null). This means that they cannot be any of the points and are eliminated from further consideration. Also note that none of the points can be labeled as an outside left eye point, right mouth or left mouth. The algorithm has significantly reduced the number of possible labels. However, there are still several possible labellings represented in this Relaxation Matrix. The next step is to search though the possible assignment of points to find the best labeling. We now search the points in a depth first method and apply more than pair-wise constraints. Three

labels need to be assigned, so the first point is picked from the matrix. In the example from Table 4.8, Point 1 is assigned a nose label. If Point 1 is a nose then none of the other points can be the nose and Point 1 cannot take another label; so, the Relaxation Matrix changes to the one shown in Table 4.9. This change in the Relaxation Matrix allows the constraints to be reapplied to relax the matrix further (see Table 4.10). This relaxation results in finding that Point 5 cannot be labeled as rEye.

Table 4.9: Relaxation matrix after selecting the first point.

	Possible Labels									
	nose	rEye	lEye	orEye	olEye	rMouth	lMouth	topCentroid	centroid	null
point1	1	0	0	**0**	0	0	0	0	0	**0**
point2	0	0	0	0	0	0	0	0	0	1
point3	0	1	1	1	0	0	0	0	0	1
point4	**0**	0	0	1	0	0	0	0	0	1
point5	0	1	1	0	0	0	0	0	0	1
point6	**0**	0	0	0	0	0	0	0	0	1
point7	0	0	0	0	0	0	1	0	0	1
point8	0	0	0	0	0	0	0	1	0	1
point9	0	0	0	0	0	0	0	0	0	1

Now, the second point needs to be selected from the resulting list in Table 4.10. In this example Point 3 is selected to be the rEye and the relaxation algorithm is applied again. However, this time relaxing the matrix does not change its values (see Table 4.11).

The final step of the search algorithm is to select the last labeling. In this example there are four points left, so each point is selected individually and the fitness function is evaluated for all four sets of points.

- Point 1 = nose, Point 3 = rEye, Point 4 = orEye
- Point 1 = nose, Point 3 = rEye, Point 5 = lEye
- Point 1 = nose, Point 3 = rEye, Point 7 = topCentroid

Table 4.10: Relaxation matrix after the second relaxation step.

	Possible Labels									
	nose	rEye	lEye	orEye	olEye	rMouth	lMouth	topCentroid	centroid	null
point1	1	0	0	0	0	0	0	0	0	0
point2	0	0	0	0	0	0	0	0	0	1
point3	0	1	1	1	0	0	0	0	0	1
point4	0	0	0	1	0	0	0	0	0	1
point5	0	**0**	1	0	0	0	0	0	0	1
point6	0	0	0	0	0	0	0	0	0	1
point7	0	0	0	0	0	0	1	0	0	1
point8	0	0	0	0	0	0	0	1	0	1
point9	0	0	0	0	0	0	0	0	0	1

Table 4.11: Relaxation matrix after the second point selection.

	Possible Labels									
	nose	rEye	lEye	orEye	olEye	rMouth	lMouth	topCentroid	centroid	null
point1	1	0	0	0	0	0	0	0	0	0
point2	0	0	0	0	0	0	0	0	0	1
point3	0	1	**0**	**0**	0	0	0	0	0	**0**
point4	0	0	0	1	0	0	0	0	0	1
point5	0	0	1	0	0	0	0	0	0	1
point6	0	0	0	0	0	0	0	0	0	1
point7	0	0	0	0	0	0	1	0	0	1
point8	0	0	0	0	0	0	0	1	0	1
point9	0	0	0	0	0	0	0	0	0	1

- Point 1 = nose, Point 3 = rEye, Point 8 = centroid

After evaluating the fitness function for these labellings, the algorithm backtracks and selects a different second point (in this example Point 3 would be a lEye). When all the possible second points have been searched, the algorithm backtracks again and searches through all of the first points. The output of this entire algorithm is a list of possible labellings with their fitness values. The labeling with the best fitness is output to the next stage of the system. The feature point system could also output the top k labellings and have the next stage determine the best fit.

Fitness Functions

In order to evaluate the quality of the set of anchor points, the matching score generated by the Iterative Closest Point (ICP) algorithm is used as a fitness function (see Section 3.5.3). The key to using ICP is to have a set of consistent control points from within the scan. Because the pose of the scan is unknown, the control points related to the pose cannot be used. Instead, all noisy data (in the z direction) are filtered out of the scan and then an evenly spaced 20×20 grid of points is placed on the remaining scan. Because a symmetrical grid is used, not all of these points are valid due to holes in the scan. The invalid points are discarded. The same final set of control points is used in all of the fitness calculations so that the distance errors generated by the ICP algorithm are comparable. The downside of using ICP as a fitness function is that it requires the use of a 3D face model (see Section 2.4.7) to calculate the distance value. This model is available in the case of verification, but perhaps not for recognition.

4.4.3 Experiments and Results

In order to evaluate the quality of the anchor points selected using the 3-step algorithm outlined in Table 4.12, each detected anchor point was compared to a manually

selected point. However, many times the manually selected points were not as accurate as the automatically selected points. This can occur when the experimenter selects a point that looks correct in the texture image but may actually be incorrect in the registered range image due to the movement of the subject's head. (This was a common problem when points were near the edges of the scan where the range data can be invalid.)

Table 4.12: Arbitrary pose anchor point detection algorithm.

1. Candidate point selection (Section 4.4.1).
2. Relaxation/search algorithm (Section 4.4.2).
3. Fitness function evaluation (Section 4.4.2).

The arbitrary pose anchor point detection system was tested on a set of 600 scans from 100 subjects (6 scans per subject). The scans include frontal pose, full profiles and smiling expressions. Each test scan in the dataset was compared to 3D models of the same subjects generated from an additional 5 scans stitched together to produce a 3D surface model (see Section 2.4.7).

The histograms of the error for all five of the core surface anchor points are shown in Figure 4.10. The median error of the five points is around $10mm$, and approximately 90% of the scans are below 20mm error for each anchor point. Extensive testing shows that the ICP algorithm can tolerate these errors for matching. Some examples of the detected points are shown in Figure 4.11.

To quantify the success rate of the entire transformation, all of the coarse transformations were evaluated by hand. Each test scan produced a transformation of the test scan onto the 3D model. If this transformation was judged to be good enough for ICP to converge, then the anchor point detection was labeled as successful for this scan. Of the approximately 600 test scans, there was an 85.6% success rate when matching a subject's test scan to the same subject's 3D model.

To fully understand where the errors occurred, the test scans were also separated into groups based on subject attributes (see Table 4.13). From this table it can be

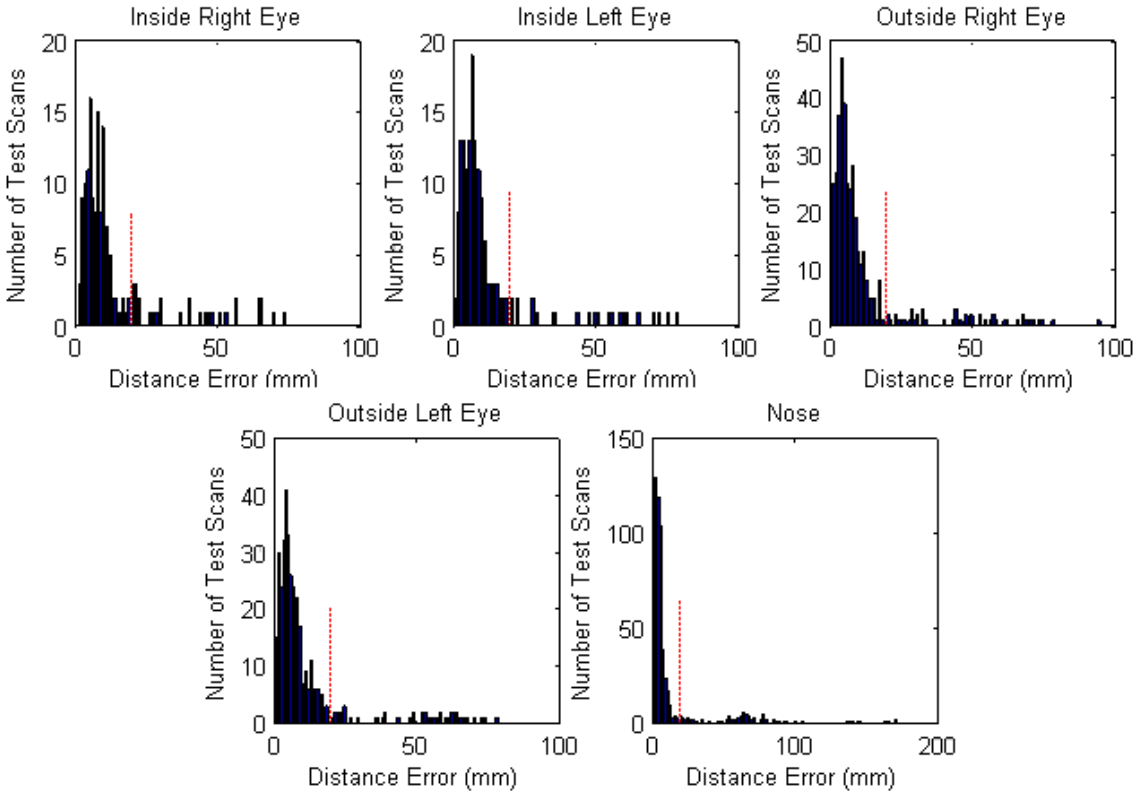


Figure 4.10: Error histograms for each of the surface candidate points. The error represents the distance from the final extracted anchor point to the manually selected point. The dotted vertical line represents the 20 mm cutoff, which 87% of the scans fall under.

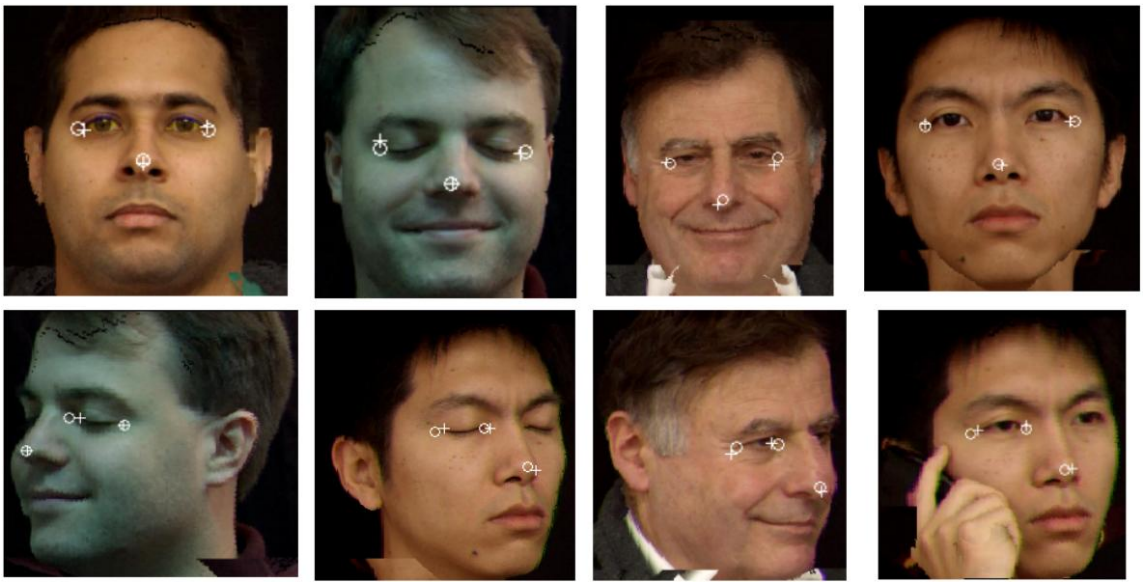


Figure 4.11: Example anchor point locations. The plus (+) represents the automatically located anchor points and the circle (o) represents the manually selected anchor points.

seen that facial hair and dark skin made it more difficult to identify key facial anchor points. This is an understandable result because both of these attributes increase the noise produced by the scanner (due to problems reading the reflected laser light). It is also interesting to note that it is easier to identify anchor points in scans with eyes closed than with the eyes open. This is probably also due to the increase in surface noise that occurs with the eyes open, since the laser dose not reflect off the pupil and causes a "hole" in the scan where the eyes should be.

Figure 4.12 shows how the success rate of the arbitrary pose anchor point detection system varies with pose and facial expression. As Figure 4.12 demonstrates, the anchor point detection system works 99.0% of the time on neutral expression, frontal pose scans. The success rate goes down a little with a smiling expression and it degrades significantly for profile views.

To improve the system performance, a reject option could be added to reject incoming scans that contain too much noise to work well with the anchor point detection system. In other words, if the system is unable to process a scan properly, then that

Table 4.13: Attribute population size success rate.

Attribute	**Population Size**	**Success Rate**
Female	25.2%	85.4%
Male	74.8%	85.7%
Facial Hair	11.2%	80.6%
Dark Skin	10.0%	81.7%
Eyes Closed	12.0%	98.6%
Asian Features	26.5%	84.3%
Profile	67.3%	79.6%
Frontal	32.7%	97.7%
Smile	47.6%	82.7 %
No Smile	52.4%	88.5%

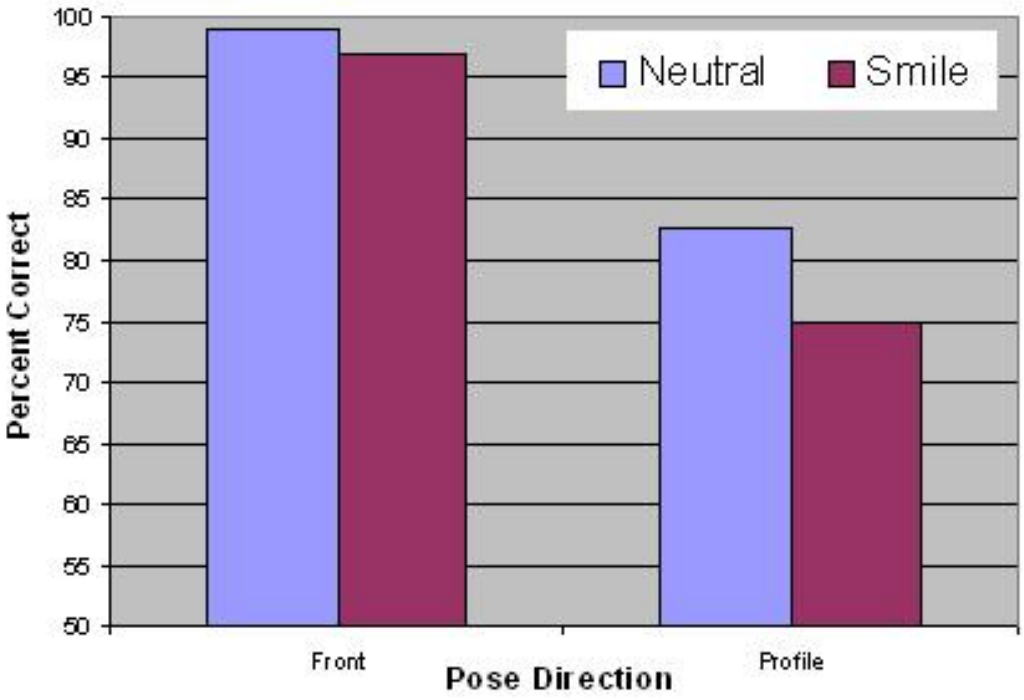


Figure 4.12: Percent of correctly selected anchor points versus the pose direction and facial expression. The success rate for frontal neutral faces is 99.0%.

scan is rejected. The goal is to find a rejection criterion that will consistently reject the bad scans and minimize the number of good scans that are rejected. Experimental results showed that extremely unsuccessful scans were often abnormally rotated. So, a rejection criterion was developed that calculates the angle of head pitch by calculating the change in angle of a line (drawn though the eye points) with the horizontal. In the MSU database there are no head pitches of greater than 35 degrees. If the rejection criteria is set to over 35 degrees, the system success rate increases to 87.0% without any false rejects. As Figure 4.13 shows, adjusting the rejection criteria allows for a design trade off between the error rate of the system and the false accept rate.

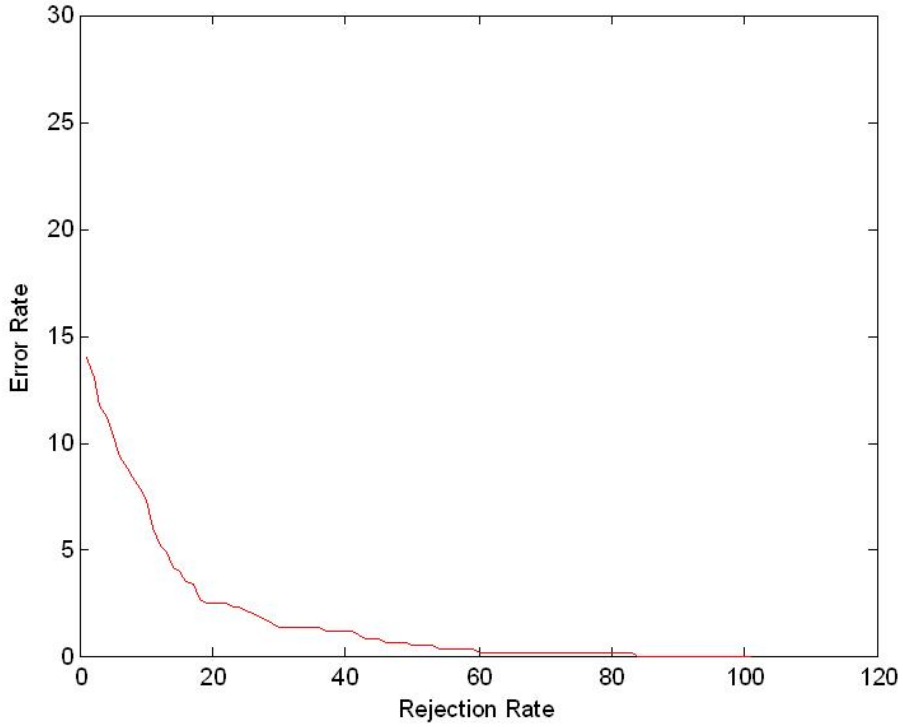


Figure 4.13: Trade off between the error rate and the false rejection rate as a function of modifying the rejection angle.

Chapter 5

3D Face Alignment

This chapter describes, in detail, a two step face alignment algorithm that accurately aligns the surfaces of two face scans (a probe scan that is to be aligned to a target scan). Figure 5.1 shows a system diagram of this two step face alignment algorithm.

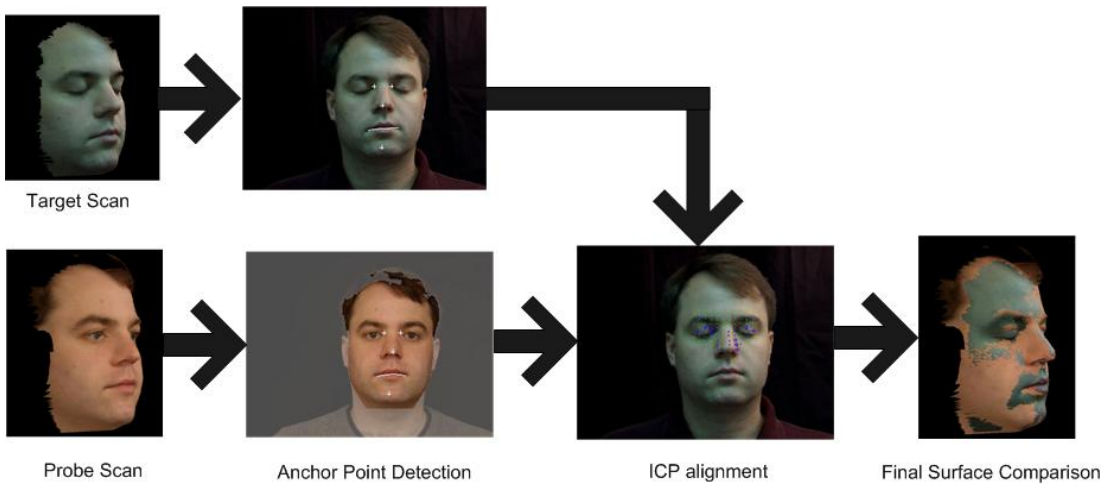


Figure 5.1: Face alignment algorithm.

The first step in this alignment process is coarse alignment using anchor points (which were described in Chapter 4). These anchor points include the nose tip, eye corners, the mouth, chin, etc., and are detected by using a model of the structure of the face, statistics on the population of human faces, and the curvature (second derivative) at various points. Anchor points detected in both the probe and target

scans are used to estimate a coarse transformation between the two scans.

After coarsely aligning the scans, the second step in the alignment algorithm is to use the Iterative Closest Point (ICP) algorithm (see Section 3.5.3) to finely align the scans. ICP samples a set of control points on the probe scan and calculates the closest distances to the surface on the target scan, then calculates a transformation that minimizes these error distances. The ICP algorithm terminates when the change in error is below a threshold, or when the iteration limit is reached. The ICP algorithm is not guaranteed to converge to the correct surface match and may fall into local minima. ICP is also intolerant to errors in the data such as spike noise and holes. To minimize this risk, the points with the largest distance errors are trimmed from the list of control points [39].

The Surface Alignment Measure (SAM) is defined as the root mean squared error over all the control points, after trimming. The SAM is used as a matching measure that allows the surface alignment algorithm to be evaluated by calculating the Receiver Operating Characteristics (ROC) and Equal Error Rate (EER). The ROC and EER allow for simple comparison and evaluation of the surface alignment algorithm without the need for manual evaluation. The ROC and EER are relative measures used to evaluate how different input parameters affect the quality of the alignment.

The remainder of this chapter describes 3D face alignment in more detail. Section 5.1 describes existing work in 3D face alignment, and Section 5.2 describes the specific face registration algorithm used in this dissertation and the experiments used to evaluate the quality and speed of the algorithm. Section 5.3 presents experiments for testing the reasonable operating range of the SAM score by testing the surface alignment algorithm on a mannequin head. Finally, section 5.4 discusses using the SAM value as a matching score, and applies this technique to the Face Recognition Grand Challenge (FRGC) dataset. Section 5.5 presents methods for extending the face alignment algorithm to account for arbitrary face poses.

5.1 Background

Image registration is "the process of determining the point-to-point correspondence between two images of a scene" [68]. Registration is one of the most difficult obstacles for biometric recognition algorithms, which must first process (or normalize) the feature space in such a way as to make the biometric data from two sources comparable. In most 2D face recognition systems, this registration/normalization process involves identifying key anchor points or areas and then transforming the data sources to have these areas align.

The first step in a traditional 2D face identification system is to detect the face in the image. Once the face is detected, the image is usually cropped and the size is standardized to match the faces in the database; this standardization may include rotation and translation components. For 3D face scans, face detection is a much simpler problem than in the 2D world. 3D faces are normally distinct blobs in the 3D space, so detecting a face is not a difficult problem. In 2D face recognition systems pose is a factor that is usually ignored (by assuming a frontal pose) because the number of different projections of a 3D face object in 2D is arbitrarily large. For 3D, however, pose is less of an issue.

The alignment algorithm presented in this chapter uses a two step procedure. The first step is coarse alignment, and the second step is to finely align the scans. Similar two step procedures can be found throughout the face recognition literature. Chua *et al.* [41] found anchor points using point signatures based on the local neighborhood of surface normals, while Conde and Cipolla [48] found anchor points by using spin images. In both of these approaches [41, 48], three anchor points are chosen to make a coarse transformation and then ICP is used for fine alignment. An advantage of these two methods of anchor point detection is that both anchor point detectors have been shown to work on general 3D objects [42, 84]. However, both spin image and point signature algorithms are prohibitively slow and have not been tested throughly

on faces. In addition, the results presented by Chua *et al.* [41] offer no evidence to show that this method would work on images with arbitrary pose.

5.2 Face Alignment Algorithm

The face alignment algorithm is outlined in Figure 5.2. The first step is to detect common anchor points on each scan; these anchor points are then used to coarsely align the scans to each other. The second step is to use the ICP algorithm (described in Section 3.5.3) to finely align the scans.

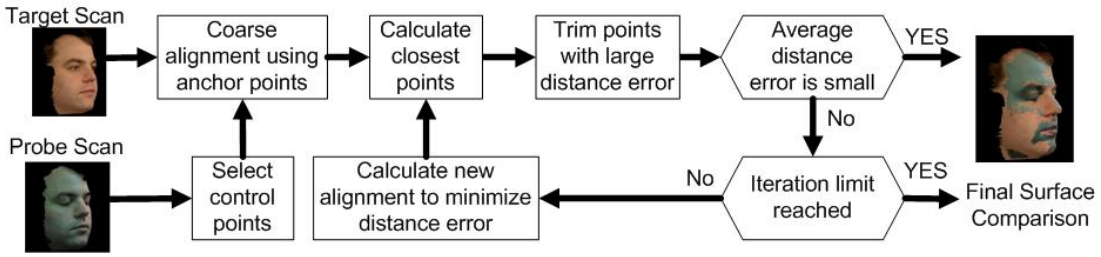


Figure 5.2: Alignment algorithm outline.

The first step of the alignment algorithm uses automatically identified anchor points (see Chapter 4) to coarsely align the probe scan to the target scan. Coarse alignment is an estimation of a three dimensional transformation from the probe scan to the target scan. The alignment does not need to be exact, but it must be close enough to then allow fine alignment algorithms to converge to an optimal solution. If it can be assumed that the faces have the same pose (i.e., frontal), then it is reasonable to only use one point for coarse alignment (see Section 3.5.1).

After coarse alignment, the second step is to determine a better estimate of the alignment. The fine alignment process follows the Iterative Closest Point (ICP) algorithm (described in Section 3.5.3). Starting with an initial estimate of the rigid transformation, ICP iteratively refines the transformation by alternately choosing corresponding (control) points in the target and probe scans and calculating the best

translation and rotation to minimize an error function based on the distance between the target and probe scans.

Table 5.1: List of design parameters used by the two step face alignment algorithm described in this chapter. The default values were initially determined by program evaluation and then verified using the experiments discussed in the listed sections.

Parameter	**Default Value**	**Section**
Control point selection region	region I	5.2.1
Number of control points selected	100	5.2.1
Maximum Number of Iterations	10	5.2.2
Error Trim Value	10%	5.2.3
Distance Threshold	10mm	5.2.4
Swap compare	true	5.2.5
Reject Option	7.5mm	5.2.6

Table 5.1 lists all of the adjustable design parameters used to develop the two step alignment algorithm. The remainder of this section discusses a series of experiments to study these design parameters. All of the experiments were run on the MSU dataset (described in Section 2.5.1) unless stated otherwise, and the time results are estimated wall times for a single processor on the MSU High Performance System.

5.2.1 Control Point Region Selection

The ICP algorithm selects a set of control points from the probe scan and uses them to calculate the best alignment to the target surface. The quality of the alignment depends on the control points chosen. There are two parameters to consider when selecting the control points: the region where points are selected and the number of points selected within this region. This section presents two experiments for each of these parameters.

In the first experiment, three control point selection regions are compared. Figure 5.3 shows the three regions chosen for the study, which used face scans that varied in expression. The quality of the surface alignment was determined by comparing the equal error rate generated by the SAM scores; Figure 5.4 shows the EER and time

changes for the different control point regions. The results in Figure 5.4 demonstrate that the control region around the eyes performs the best, which is not surprising since this area of the face does not change much with expression. There was no significant difference in the execution times for these control ranges.

(a) Region I (b) Region II (c) Region III

Figure 5.3: Control point selection regions.

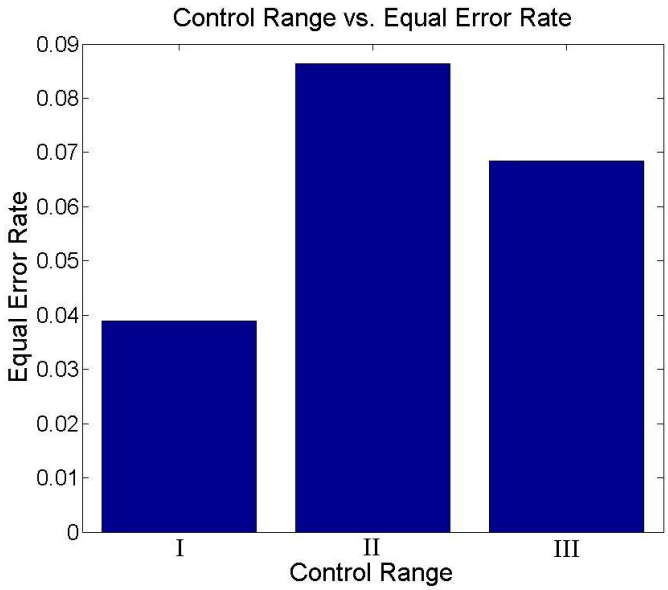
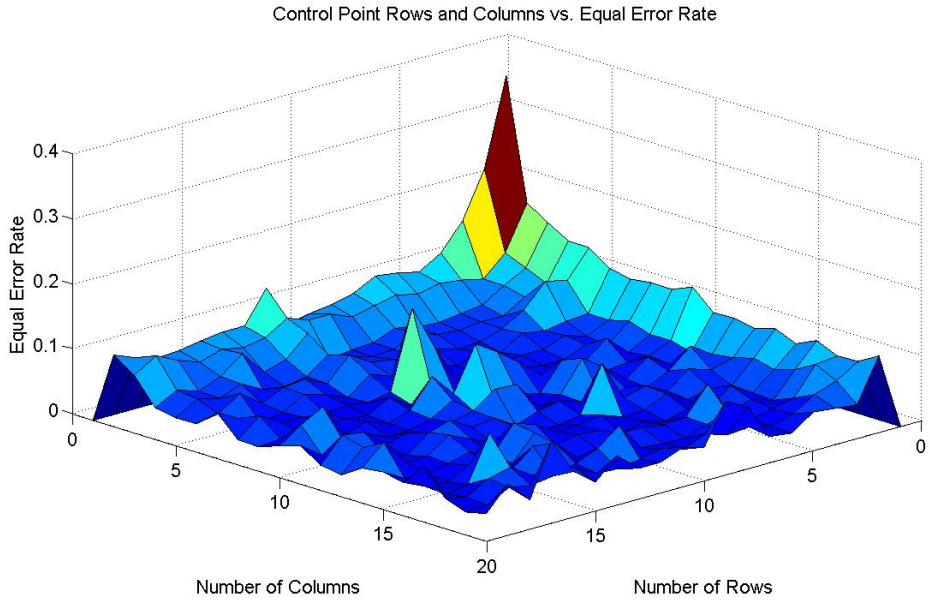
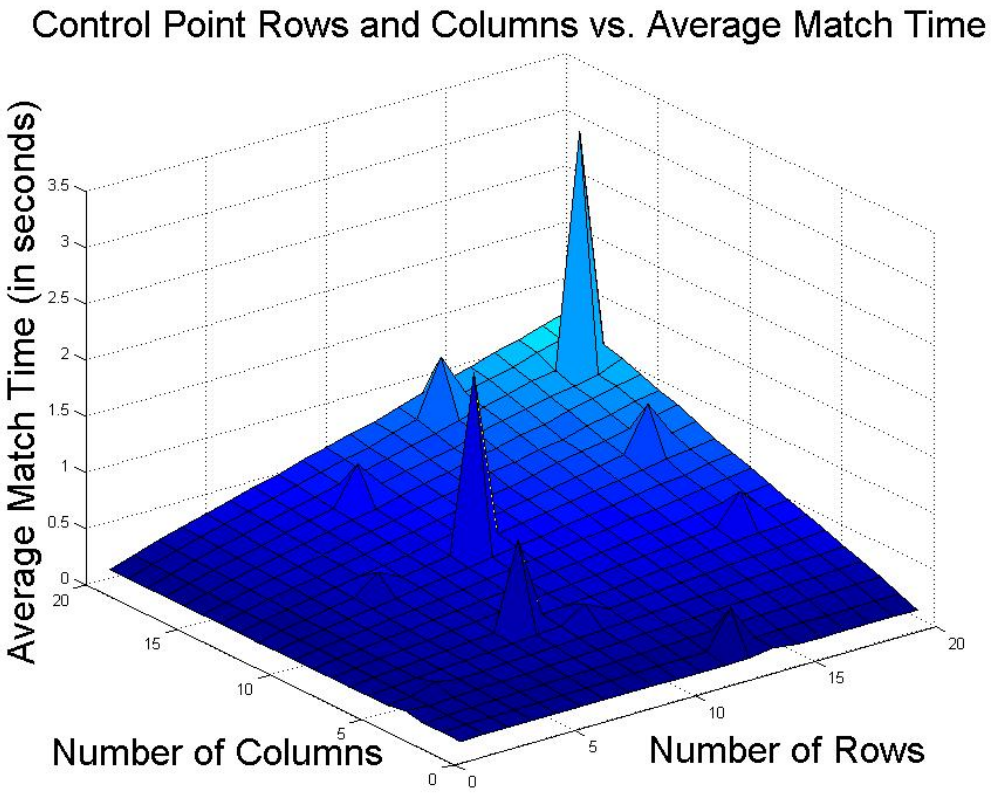


Figure 5.4: Results when varying control point ranges; the numbers correspond to regions I, II and II from Figure 5.3.

The second factor considered when selecting control points was the number of control points used. Figure 5.5 shows experimental results when varying the number of control points in a grid within the selection region. The results in Figure 5.5a



(a) EER



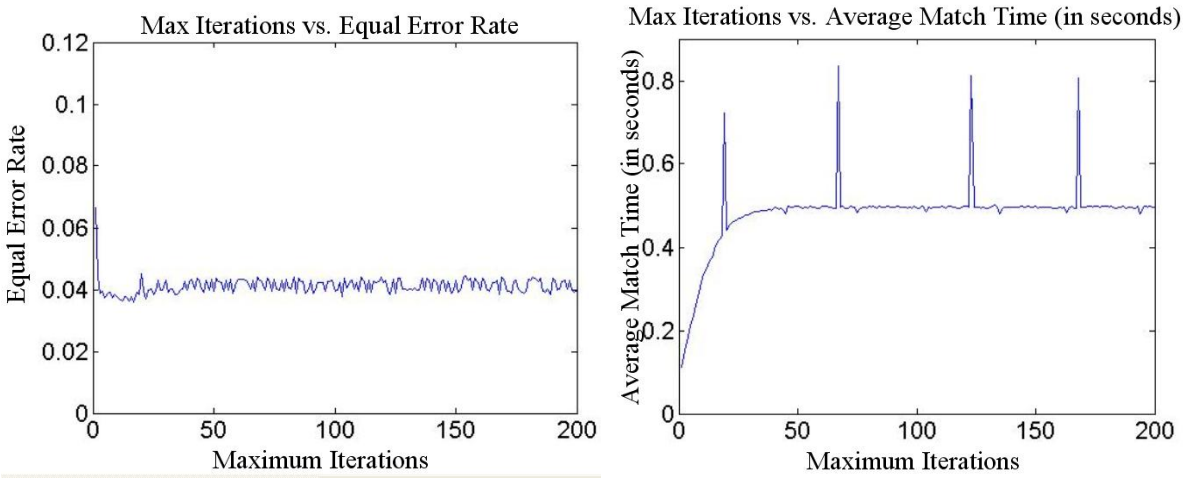
(b) Time

Figure 5.5: Results when varying the number of control points.

indicate that the number of control points within the region is not a factor, as long as there are enough control points (a minimum of about 25) to ensure that the EER stays fairly constant. Figure 5.5b confirms that increasing the number of control points within a region increases the processing time linearly.

5.2.2 Number of Iterations

It is possible that the ICP algorithm will oscillate between a number of values instead of converging during the fine alignment stage. In such cases, an iteration limit is needed to ensure that the ICP algorithm will terminate. If the iteration limit is too high, ICP will take longer than necessary to run; however, if the iteration limit is too low, ICP may not have enough time to converge.



(a) EER

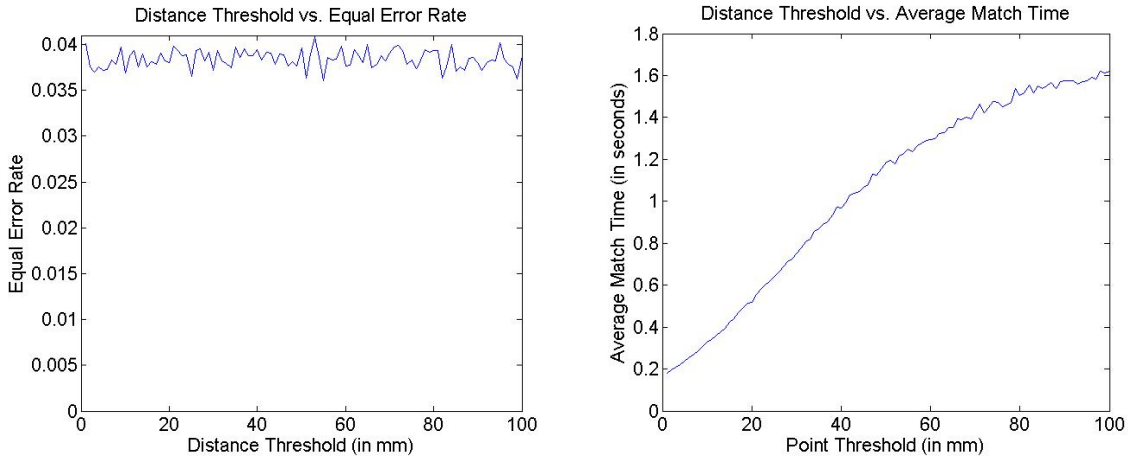
(b) Time

Figure 5.6: Results when varying the maximum number of iterations.

Experiments were conducted to determine the optimal number of iterations. Figure 5.6a shows that increasing the iteration limit above 5 does not improve the EER, while Figure 5.6b demonstrates that an iteration limit higher than 40 allows all the scans to converge (error spikes are due to inconsistent system load).

5.2.3 Distance Threshold

The distance threshold is the maximum distance at which the algorithm will look for the closest point. This is a required parameter for the fast 2.5D matrix ICP algorithm introduced in Section 3.5.3. A high distance threshold allows ICP to make large jumps in its alignment; however, small distance thresholds reduce the point search area, help remove outliers, and speed up the algorithm.



(a) EER (b) Time

Figure 5.7: Results when varying the distance threshold.

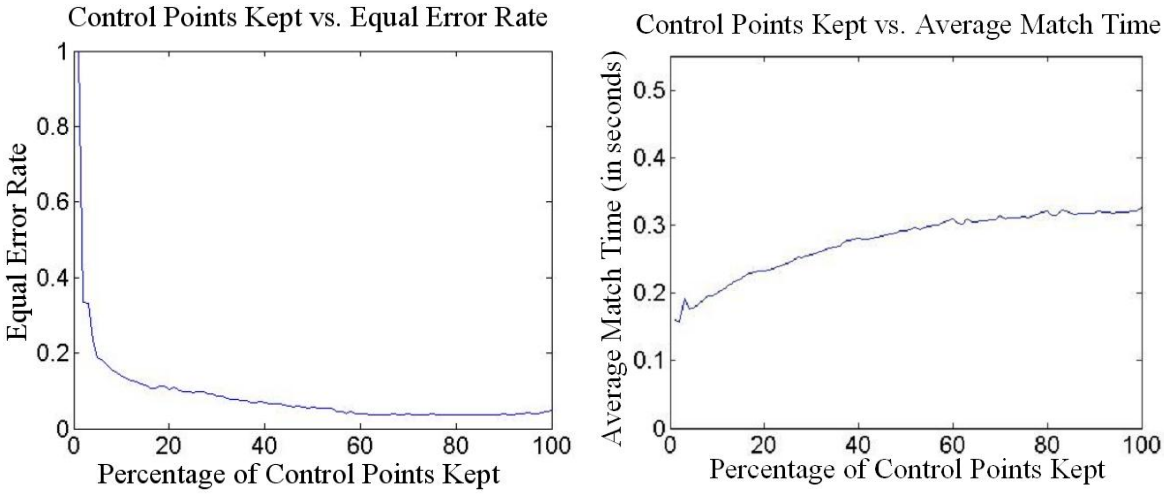
Figure 5.7 gives the results of varying the distance threshold. The minimum for the threshold values shown in Figure 5.7 is the minimum of the scan resolution. Figure 5.7 indicates that increasing the distance threshold above this minimum does not affect the quality of the matching error. Figure 5.7b demonstrates that the smaller the distance threshold, the faster the algorithm.

5.2.4 Trimming Study

The two step registration algorithm (coarse and then fine) described in this chapter works well in many cases. However, errors may exist in the final transformation matrix due to noise in the data or changes in the surface shape of the object (such

as a change in expression). Trimming is used as a robust method for improving the calculation of the alignment to better tolerate noise and variations in expression.

An example of a more robust algorithm is Trim ICP [39]. This is a simple variation on classical ICP that incorporates a trim threshold, which is a fraction from zero to one that represents the percentage of control points that will be used to calculate the transformation matrix. For example, if one hundred control points are used and the trim algorithm is set to 0.8, only 80 of the best points (those with the smallest distance errors) will be used to calculate the matrix. This will improve the algorithm performance for data where up to 20% of the information is spike noise.



(a) EER (b) Time

Figure 5.8: Results when varying the number of points trimmed.

Figure 5.8 shows that, in general, more control points results in better performance. However, there is a small drop off around 95% due to errors in the scans (such as spike noise and holes). The results in Figure 5.8a show that the trim parameter is robust to minor variations in its value, and a trim as low as 60 still produces good results while improving execution time (see Figure 5.8b).

5.2.5 Scan Swap

The alignment algorithm presented in Section 5.2 is asymmetrical, meaning that it is possible to get different answers depending on which scan is used as the target and which one is used as the probe. These discrepancies occur because different sets of control points are selected on each scan. Also, it is possible that control points will fall into holes or data spikes that can cause large matching errors if the scan is used as a target.

By swapping the target and the probe, two different SAM values are calculated and the lowest matching score (SAM) is chosen as the surface matching score. Figures 5.9, 5.10, and 5.11 compare asymmetric and symmetric (swap) algorithms on various datasets.

As Figures 5.9, 5.10 and 5.11 demonstrate, there is no significant difference between the asymmetric and symmetric matching results. This suggests that it may be better to run the algorithm in an asymmetric mode because the alignment algorithm will only have to run once. However, during exploratory test runs of the system it was found that swapping the scans and taking the minimum SAM improved results, especially if there were large pose variations in one of the scans. These types of large pose variations are not well represented in the dataset.

Instead of using the swap as a way to avoid bad target data, another option is to set some standards on the quality of the target scan. For example, during enrollment target scans should be evaluated for the size and number of holes and spikes. Surface cleaning techniques could also be used to improve the quality of the target model (see Section 3.4). With these standards in place there is less need to run a symmetric (swap) alignment algorithm.

A third option is to speed up the swap algorithm by terminating early (before the swap) if the first match is below the matching threshold. For example, in both FRGC 2.0 and FRGC 1.0 at least 92% of the pairs of genuine matching scores are

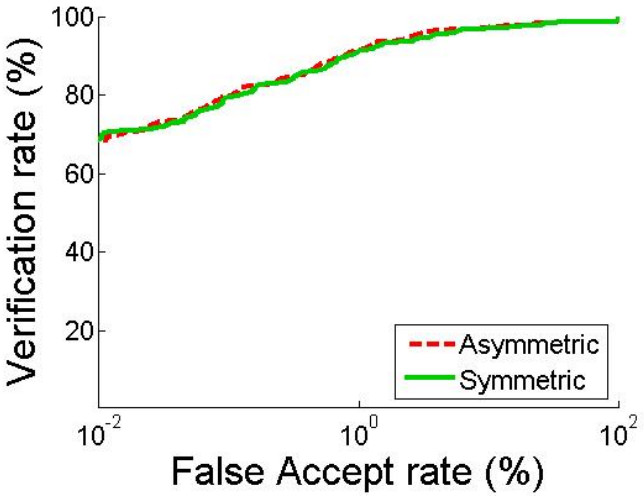


Figure 5.9: Symmetric matching performance on MSU dataset.

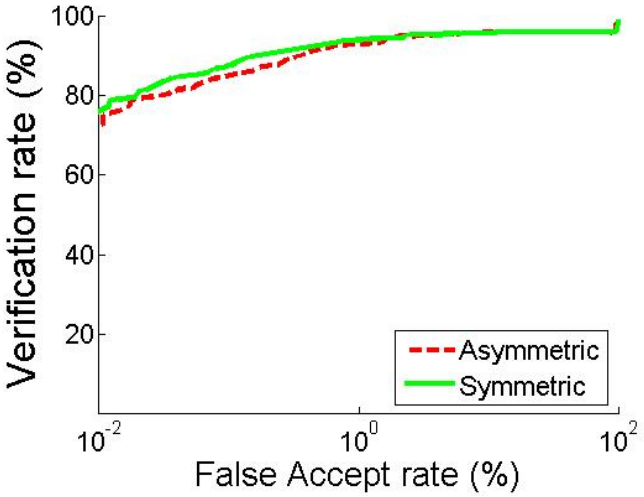


Figure 5.10: Symmetric matching performance on FRGC version 1.0 dataset.

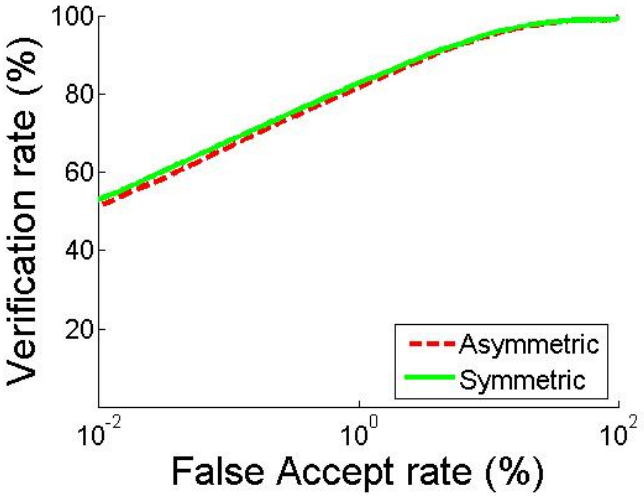


Figure 5.11: Symmetric matching performance on FRGC version 2.0 dataset.

both under $1mm$ SAM. If a subject is automatically accepted when the first SAM value is below this threshold, then the second ICP match (i.e., the swap) could easily be skipped.

5.2.6 Automatic Reject Option

There are times when the alignment algorithm fails due to poor quality scans. This can occur if there is noise in the scan, if the subject moves during scanning, or if the pose varies too far from frontal. In these cases it may be better to have the algorithm detect that there is a problem with the scan and notify the subject.

A rejection algorithm was developed based on the symmetry of the face, using ICP to align the current probe scan with a mirror image of itself. If the anchor points were not found correctly, then the SAM score between a probe scan and its mirror image will likely be high. However, if the anchor points are found correctly the SAM score is normally quite low when matching a scan to its mirror image. Using this assumption, 14 (1.5%) of the 948 scans in the FRGC 1.0 database were rejected due to poor symmetry in the scan.

Figure 5.12 show some examples of the rejected scans. Errors such as spike noise in the hair (Figure 5.12a), spike noise around the eyes (Figure 5.12b) and scans with no valid depth points (Figure 5.12c) are typical types of errors that the anchor point algorithm cannot handle. Only 1 good scan was incorrectly rejected and 2 bad scans were not rejected using the symmetry based reject criteria. After implementing the reject option, the ROC curves were recalculated and a new EER of 1.2% was achieved (see Section 5.4).

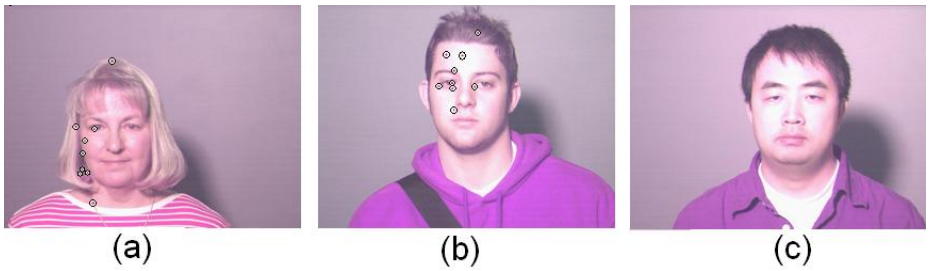


Figure 5.12: Rejection examples.

5.3 System Error Analysis

When using human subjects, it is difficult to determine if variations in the SAM are associated with actual changes in the head (e.g., changes in expression, weight, hairstyle, etc.) or are just an artifact of sensor noise. Figure 5.13 shows a mannequin head (named Marie) bought from a local beauty salon. Data from this mannequin was used as ground truth for evaluating the errors associated with the scanner and the matching system developed for this dissertation.

Figure 5.13: Example mannequin data with anchor points detected.

Multiple scans of Marie were collected while varying the roll, pitch, and yaw of the head, as well as the distance to the camera, in order to evaluate the best operational distance for face matching. Marie's nose tip was initially set to 152cm from the camera, then moved forward (toward the scanner) in increments of 10cm. It was discovered that any measurement made closer than 77cm to the scanner could make

nose detection difficult because the scanner's focus range did not include the nose (see Figure 5.14). This was an intermittent problem that occurred when the scanner missed the tip of the nose during the focusing procedure.

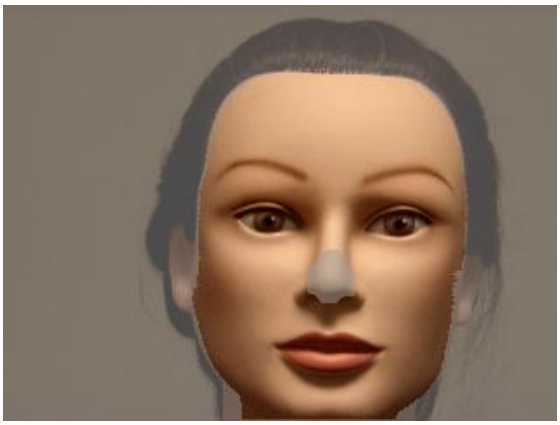


Figure 5.14: Nose was not captured due to the focusing range of the Minolta scanner. Scans that are closer than 0.77m can cause problems.

Five experiments were conducted to evaluate the performance of the surface matching system due to external conditions. The first examined the effects of facial color and lighting. The second experiment was a control experiment to determine the baseline SAM values for standard operation of the system when subjects are cooperative, maintain a neutral expression and are 1 meter from the scanner. The third experiment evaluated the performance of the system under changes in head pose (roll, pitch, yaw). The fourth experiment explored the changes in SAM when varying the distance between the camera and the subject. In the final experiment, the face matching system was used to calculate the SAM scores on the FRGC version 1.0 dataset (948 scans of 275 people with non-rigid features). The remainder of this section discusses these experiments in detail.

5.3.1 Color and Lighting

Changes in color (skin tone, makeup, lighting, etc.) can affect the Minolta scanner's ability to detect the laser, which leads to failure of the triangulation scheme and of

computing the depth value. Washout from severe lighting inhibits the camera's ability to detect the laser beam, as do some dark colors. These types of color and lighting changes, however, usually do not prohibit the matching system from generating an acceptable SAM value. Figure 5.15 gives examples of extreme lighting and color changes that do not prevent the alignment system from functioning properly.

(a) Low Lighting (0.23 mm SAM) (b) Camouflage (0.59 mm SAM)

Figure 5.15: Example scans with extreme lighting and color variations.

5.3.2 Optimal Conditions Baseline SAM

20 scans (totalling 190 matching scores) were made of Marie in a stationary frontal pose with constant florescent lighting. The resulting SAM values had a mean of $0.37mm$, $\pm 0.15mm$. These results indicate the range of expected variation due to sensor noise, and suggest that two scans that fall within this range indicate a proper alignment of the same face surface.

A histogram of these SAM values for Marie is shown in Figure 5.16. The dotted line represents the Gaussian approximation to the histogram with a mean at $0.37mm$ and a standard deviation of $0.15mm$. As a comparison, two more Gaussian distributions are shown, one for intra-class (Genuine) SAM values and the other for inter-class (Imposters) SAM values. The "Genuine" curve represents scans taken from

111 people; in half of these scans the people were smiling. Notice that the mean of the genuine curve is slightly higher than that for Marie due to changes in expression ($0.7mm \pm 0.15$). The "Imposters" curve was taken from the same 111 subjects. Notice that the SAM values for Imposters are much higher ($1.5mm \pm 0.3$). The overlap between the Genuine and Imposters curves represents the error region for the face matching system.

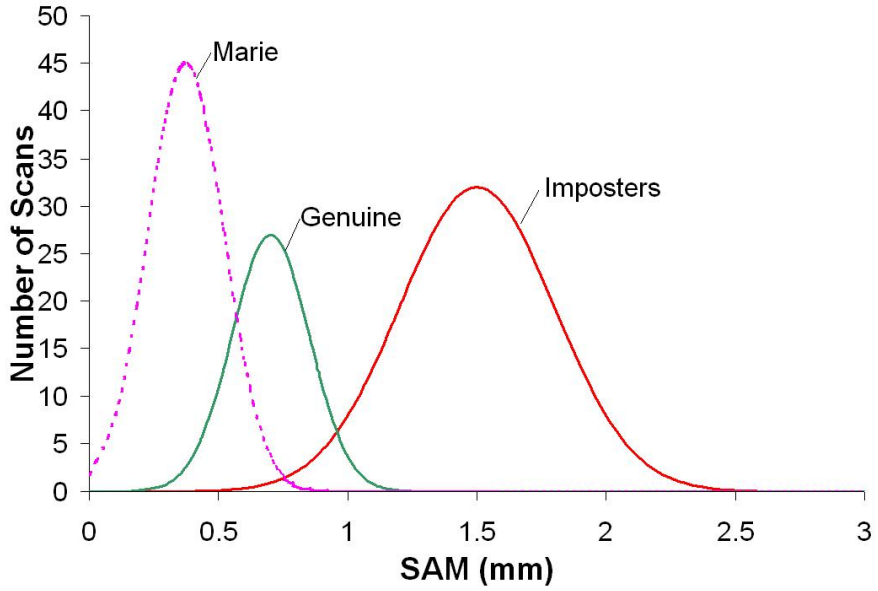


Figure 5.16: Gaussian approximation of SAM distributions.

This baseline experiment is similar to one reported by Boehnen and Flynn [22], which compared characteristics of different scanner technologies. In their study, scans were made of rigid masks produced from a milling machine and then compared to the original shape of the masks. Their study showed that the Minolta Vivid 910 scanner provided the most accurate surface measurements. However, surface matching errors reported in [22] are much lower than many of the averages seen in the experiments reported in this dissertation. Four experimental differences account for this discrepancy. First, the study in [22] used the fine resolution mode on the Minolta scanner (640×480 pixels) instead of the fast resolution mode (320×240 pixels) used

in this dissertation. This change in resolution will affect the baseline distance errors for the closest point calculation. Second, Boehnen and Flynn preprocessed the scans to remove small holes and spike noise on the surface of the scan. Third, the faces were rigid and the entire face area was used including the area around the mouth and chin, enabling more accurate pitch values. Finally, the masks used in [22] were manufactured to have an optimal white surface to reflect the light from the Minolta scanner. These experimental differences combined account for the differences in error scores between the study reported here and the study described in [22].

5.3.3 Pose Variation

The third set of Marie experiments tested the effects of changing the relative pose angle between the face and the camera. There are three types of pose changes that a face can undergo in 3D space: yaw, pitch, and roll (see Figure 5.17).

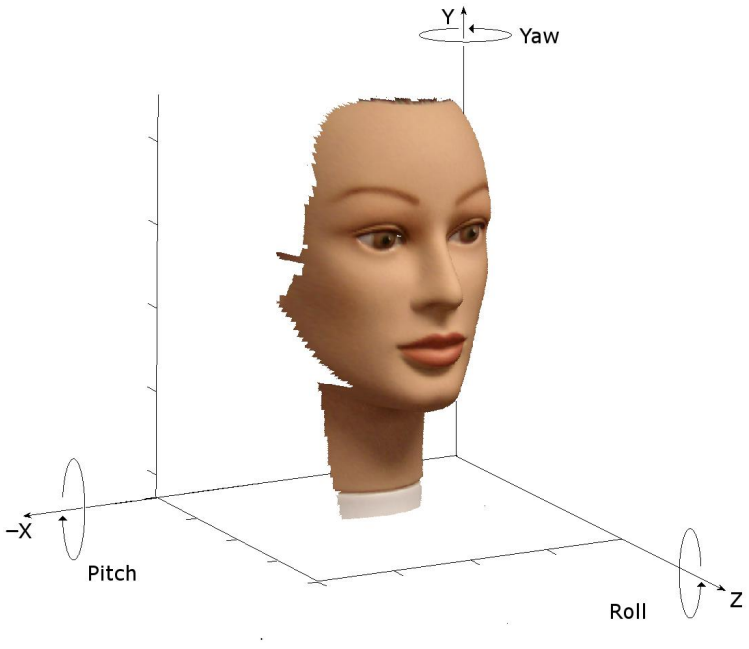


Figure 5.17: Pitch, yaw and roll.

The yaw, pitch and roll were calculated on all of the datasets and the average values are shown in Table 5.2. Also included in the table is the maximum roll, pitch

and yaw between the two scans with the most variation. These values are used as an estimate of how much variation may be expected in a real world application because the subjects in these datasets were asked to maintain a frontal pose.

Table 5.2: Differences between valid scans in the database.

	Roll	Pitch	Yaw	Distance
MSU	0.36°	1.62°	1.52°	$63.69mm$
	± 0.49	± 2.25	± 1.99	± 66.08
max	3.11°	16.03°	17.30°	$431.97mm$
FRGC1.0	0.73°	2.95°	2.54°	$164.44mm$
	$0.81\pm$	$2.50\pm$	$2.13\pm$	$78.17\pm$
max	6.12°	13.03°	17.58°	$646.75mm$

The results of the pose experiments can be found in Figures 5.18, 5.19 and 5.20. The Marie results represent an average of 5 scans, while the virtual rotation database results are averaged over 300 scans for each rotation angle. Any matching score above $5mm$ is considered to be a false match.

In the yaw and pitch experiments, the increase in the SAM values at high rotations ($>$ 10 degrees) is due to inaccurate anchor point detection. This is an expected result because the anchor point detection system was designed to find anchor points on strictly frontal pose scans. In experiments with roll, the nose is always detected because the nose tip always remains closest to the camera. The anchor point algorithm uses the nose location to identify other anchor points by calculating bounding boxes on statistics generated from a number of faces, under the assumption that the face is relatively vertical. Thus, if the head is significantly rolled the calculated bounding boxes will not contain the desired anchor points.

ICP does not need all of the anchor points to correctly align two surfaces; therefore, the gradual increase in SAM shown in Figure 5.18 (10 ICP Iterations) is not due simply to the missing anchor points. Instead, the increase is due to the number of iterations used by ICP. The roll experiment were re-run using fifty ICP iterations instead of ten; the results are also shown in Figure 5.18. The graph confirms the

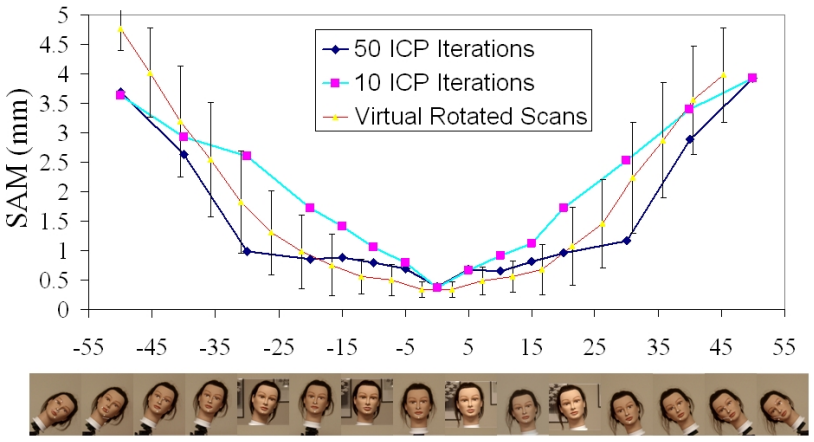


Figure 5.18: Change in roll vs. SAM.

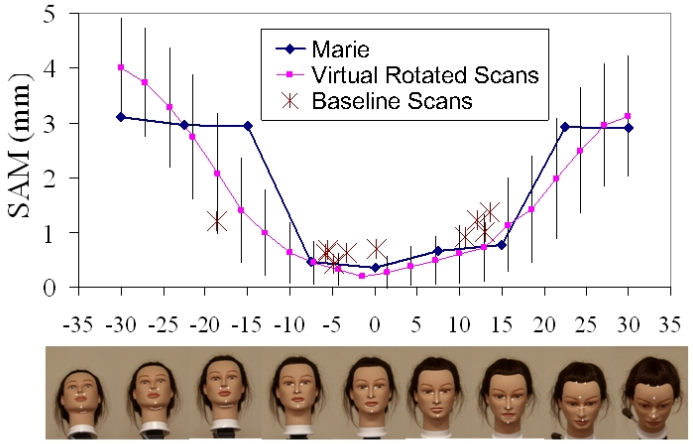


Figure 5.19: Change in pitch vs. SAM.

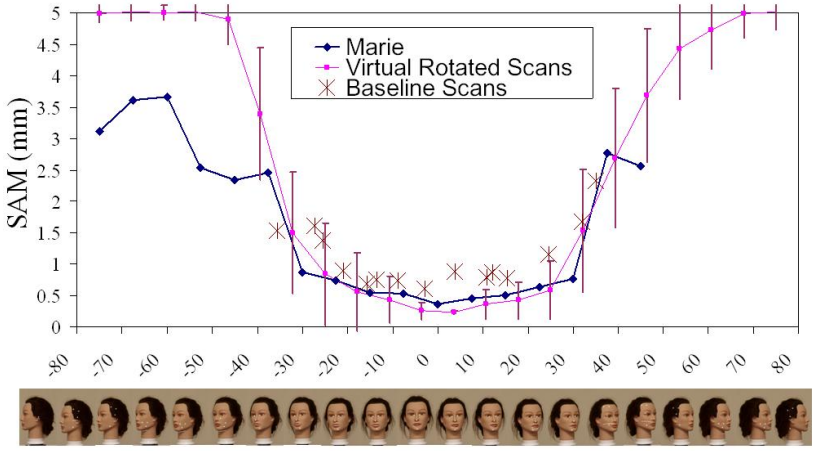


Figure 5.20: Change in yaw vs. SAM.

hypothesis since a flattened out, low SAM area, is now present between -30 and 30 degrees for both the Marie and the virtual rotation data. With 50 iterations, the range of acceptable SAM greatly expands and almost all of the tests converged (the average number of iterations was 19, with a standard deviation of 13 iterations).

5.3.4 Change in Stand Off Distance

The Minolta Vivid 910's operating stand off for the medium distance lens is reported to be between 0.6 to 2.5 meters [108]. In the Marie experiment, the system was tested with the distance between the camera and the subject varied from 0.35 and 3.05 meters. The system found no valid points on the face when the distance was below 0.5 meters and above 2.9 meters. The results of the valid scans can be seen in Figure 5.21. The closest viable distance for Marie to the scanner was found to be 0.65 meters. At this distance, the matching error level was measured to be $0.41mm$. If Marie was closer to the camera, the computed SAM value was not reliable. As the distance increased, the SAM increased at an almost linear rate.

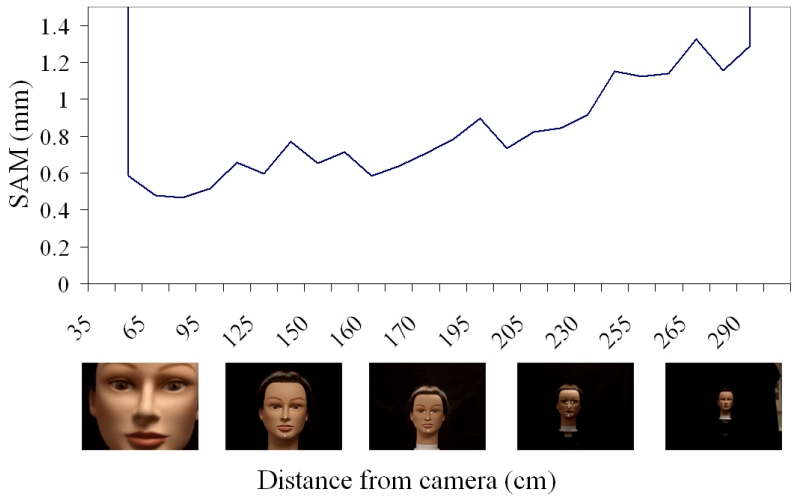


Figure 5.21: Change in stand off vs. SAM.

5.4 Large Dataset Experiments

In the final experiment, the alignment algorithm was evaluated on the Face Recognition Grand Challenge (FRGC 1.0) database, containing 275 subjects with 948 subject scans. All the scans are frontal pose, neutral expression; however, there is some deviation from this due to natural human pose. The data were gathered using a different Minolta 910 scanner from the one used in the Marie study, and the 640×480 high resolution setting was used instead of the 320×240 fast resolution mode. The higher resolution can cause lower sampling errors for ICP; however, the higher resolution scan also takes longer to capture, so there are more data errors due to motion of the subjects. A baseline 3D matching performance based on PCA (similar to [133]) is also available for this dataset from the University of Notre Dame. In this baseline algorithm, both the color and depth components of the face scans are normalized to a standard width and height using manually selected points at the center of the eyes. Once the images are normalized, the PCA algorithm is applied to both data channels independently and the matching results are reported. Figure 5.22 shows the ROC curves produced by the baseline algorithm and the SAM matching score on the MSU dataset. Figure 5.23 shows the same graphs on the larger FRGC1.0 dataset and Figures 5.24, 5.25 and 5.26 show the graphs for the FRGC2.0 dataset.

The Notre Dame baseline algorithm has an EER of 4.6% and the SAM (as a matching score) has a baseline EER of 3.8% on the same database. Figure 5.27 shows the three face pairs with the lowest false positive SAM scores. The SAM score performs much better for low values of the false positive rate. Many of the errors on the top end of the ROC curve are due to incorrect detection of the anchor points. In fact, the 3.8% EER includes only 23 False Reject errors. One of these scans is missing its depth data, and the other 22 errors are due to bad anchor points in the same 15 scans. These error rates demonstrate some of the difficulty with this testing method: a single error in anchor point detection on a single scan can propagate and

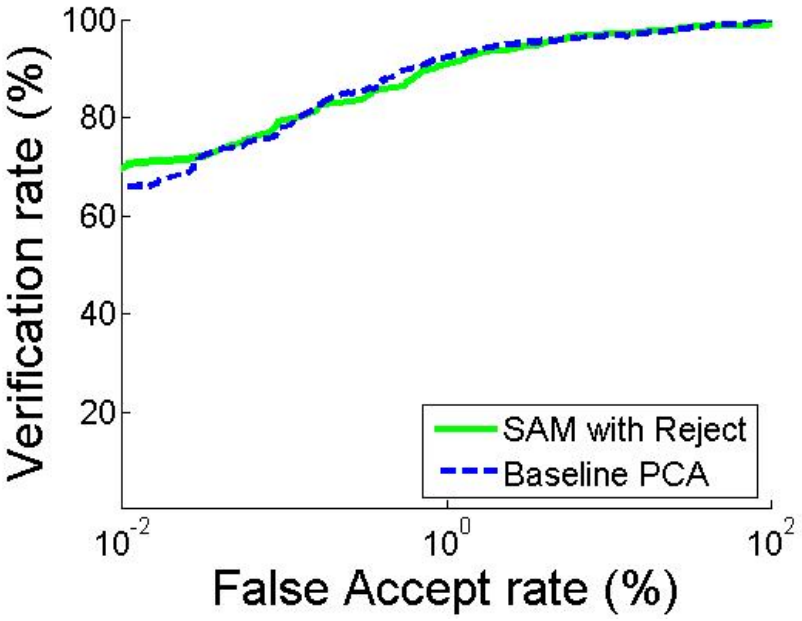


Figure 5.22: ROC curves for the MSU dataset.

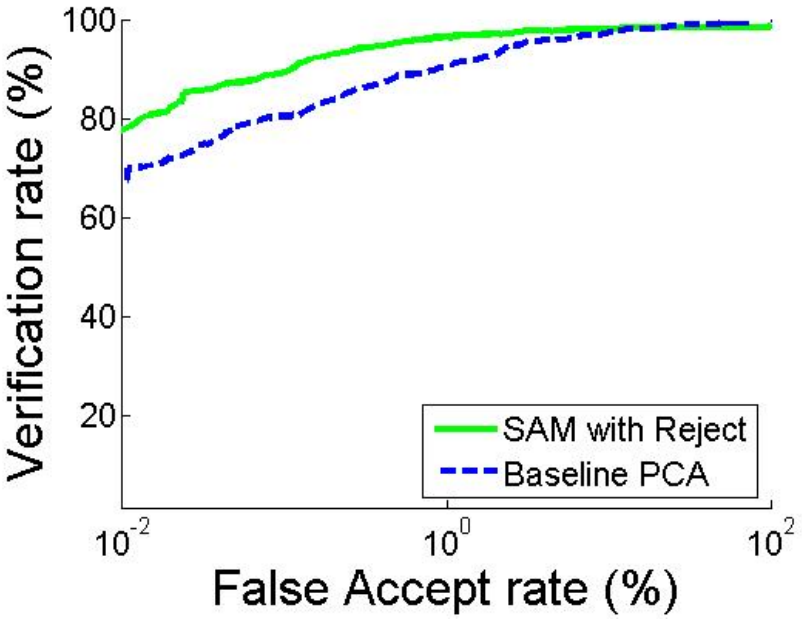


Figure 5.23: Resulting ROC curves for the FRGC 1.0 dataset.

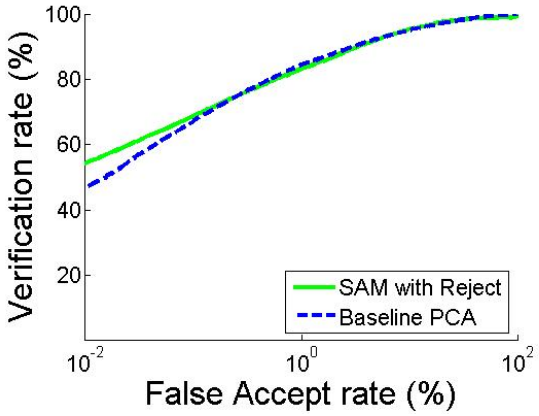


Figure 5.24: Performance on FRGC 2.0 dataset when the training and testing data are both from the same semester.

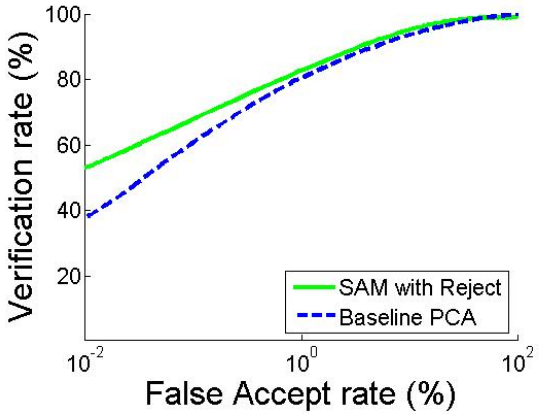


Figure 5.25: Performance on FRGC 2.0 dataset when the training and testing data are taken from two different semesters.

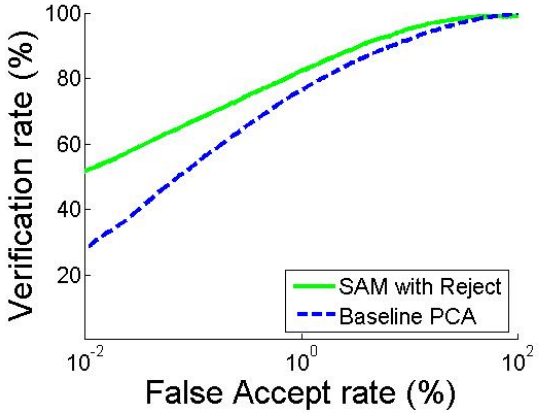


Figure 5.26: Performance on FRGC 2.0 dataset when the training data is from one semester and the testing data is from a later semester.

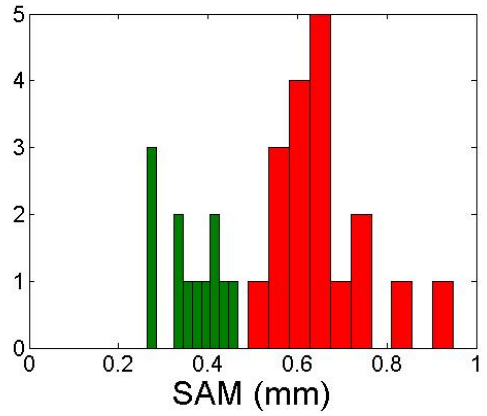
cause misleading results. In a real application, it would be better to automatically identify and reject some of the bad scans and have the subject rescanned.

Figure 5.27: Example false accept SAM scores.

Similar experiments were conducted on the FRGC2.0 dataset. As described in Section 2.5, this dataset is much larger and more difficult than the version 1.0 dataset. Results are reported for three matching conditions. Figure 5.24 shows the results with the target and query scans taken from the same semester. A more difficult situation is shown in Figure 5.25, where the target and query data span two different semesters. The most difficult situation is shown in Figure 5.26, where the target data is gathered in one semester and the query data is taken from a different semester. In all cases, the SAM score performs slightly better than the Baseline PCA algorithm. However, quite a few research groups anonymously reported 100% performance on the FRGC1.0 dataset. These algorithms tend to use manually selected points and train extensivly on the dataset. It is unlikely that a valid cross section of the human population will yield 100% accuracy using face recognition. Human faces vary too much and there is at least a subsection of the population that looks alike. Figure 5.28 shows the SAM scores for matching two pairs of twins. Notice that even though the genuine scores are smaller than the imposter scores, most of the scores are well below the standard

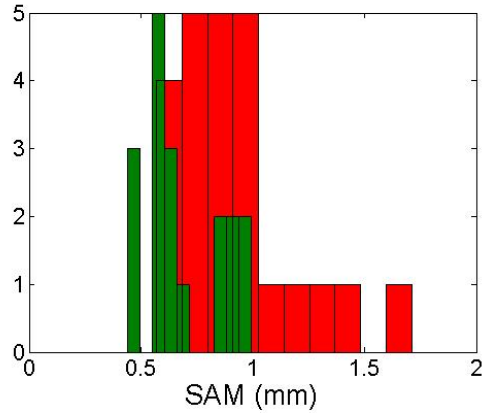
threshold of around 1mm. This means that the algorithm cannot know if the values are from the same subject or different subjects. This does suggest that a recognition system may be able to tell twins apart, but a verification system that normally relies on a globally trained threshold would be unsuccessful.

(a) Twin Set A



(b) Histogram of SAM for Twin Set A

(c) Twin Set B



(d) Histogram of SAM for Twin Set B

Figure 5.28: Results for matching two sets of twins. The genuine scores are in green (thin bars) and the impostors are in red (wide bars).

5.5 Arbitrary Pose

Section 5.4 demonstrated that the frontal alignment algorithm performs quite well even with small variations in pose and expression. However, the experiments in

Section 5.3.3 also demonstrated that large variations in pose can cause the frontal face alignment algorithm to fail. This section describes the necessary modifications to allow the frontal alignment algorithm to be used as an arbitrary pose alignment algorithm.

When designing an arbitrary pose alignment algorithm, some of the assumptions made in the frontal alignment system do not apply. Specific problems that need to be addressed when working with arbitrary poses include:

- A single anchor point is no longer sufficient for coarse alignment.
- There may not be enough points in the control point selection region for proper alignment.
- Two scans with different poses may not overlap at all.

To address these problems, the following modifications were made to the alignment algorithm:

- A full 3D face model (see Section 2.4.7) was used as a target scan to ensure enough overlap with the probe scan.
- The three point rigid alignment algorithm (described in section 3.5.2) was used for coarse alignment.
- The control point selection regions changed depending on the pose of the scan (see Figure 5.29).

The Besl ICP algorithm [17] uses point-to-point distance and a closed-form solution when calculating the transformation matrix during each iteration. The point-to-plane distance used by Chen [37] makes the ICP algorithm less susceptible to local minima than the point-to-point metric [17, 64]. Figure 5.30 shows the results of the ICP matching of a full 3D model with a 2.5D test scan.

The two classical ICP algorithms [17, 37] are also integrated in a zigzag running style called the hybrid ICP algorithm [100, 99]. Each iteration consists of two steps:

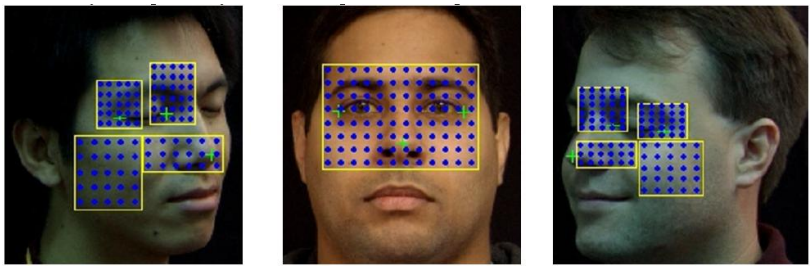


Figure 5.29: Grid of control points used by the fine alignment ICP algorithm. The grid is adapted to the pose of the test scan. The location of the grid points is determined by the location of the extracted anchor points.

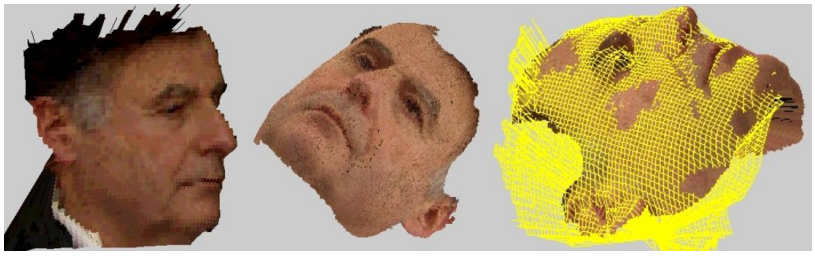


Figure 5.30: Matching results after fine alignment. From left to right: the 2.5D test scan, the 3D model, the 3D textured model overlaid by the wire-frame of the test scan after fine alignment.

using Besl's scheme [17] to compute an estimation of the alignment, followed by Chen's scheme [37] for a refinement. The two different distance metrics are utilized together, which has the potential for a better registration result.

In order to properly evaluate the performance of the arbitrary pose alignment system developed in this dissertation, it was tested in two experimental modes: automatic and manual. In automatic mode, the algorithm selects the anchor points used in coarse registration, while in manual mode the anchor points are labeled by hand. The average error between the manually selected anchor points and the automatically selected anchor points was 18.12mm, with a standard deviation of 37.58mm. This error seems large because of the outliers in the automatic point selection. The error was calculated by taking the average distance between the manually labeled points and the automatically extracted points. Many of the points automatically selected by the system are repeatable, even though they do not match the manual points. In

addition, there are some outliers where the automatic anchor point selection system failed completely (see Chapter 4).

The results reported here are from two experiments dealing with arbitrary pose matching. In the first experiment [100], the system was tested on a database of 63 scans of the same 10 people, which varied in pose (up to ±60 degrees from the frontal view), facial expression and lighting. In the second experiment [99], the database was extended to 18 subjects for a total of 113 scans. These studies give an idea of the success rate when matching 2.5D to 3D faces.

In the first experiment, the fine alignment and face recognition systems were tested on both manually selected anchor points and automatically selected anchor points. The recognition accuracy is given in Table 5.3. The hybrid ICP achieved the best performance among three ICP schemes, and Table 5.3 presents two classical ICP algorithms for comparison. In this test, a success rate of 1 error out of 15 (93.3% accuracy) for frontal test scans was achieved when comparing frontal 2.5D face images. This is noteworthy considering that in many of the test images people had different expressions than in the training images (i.e., some were smiling).

Table 5.3: Number of failed tests out of 63 scans due to recognition in the first test. The number of iterations (10,10) represents a total of 10 iterations for Besl's ICP scheme and 10 iterations for Chen's scheme.

Algorithm	Number of Iterations	Anchor Point Detection	Range image only	Range Texture Shape Index
ICP_hybrid	(10, 10)	Manual	6	5
ICP_hybrid	(10, 10)	Auto	13	10
ICP_Chen [37]	20	Manual	10	6
ICP_Chen [37]	20	Auto	16	14
ICP_Besl [17]	20	Manual	19	5
ICP_Besl [17]	20	Auto	33	15

When the testing database was extended to include semi-profile images, the error went up (92% accuracy). These results assume that the coarse alignment procedure

is correctly done. The combined total error of the entire automatic system (which consists of all changes in pose and expression) was 10 out of 63 (84% accuracy). The hybrid ICP scheme is still robust to the localization errors of anchor points, which outperforms the other two classical ICP algorithms on the surface matching (range image only) [100]. The combination of surface, texture and shape index matching achieves better performance than surface matching based on the range image alone.

In the second experiment, the face recognition system was run with two different matching scores. The first matching score only used the distance measurement produced by the ICP algorithm. The second matching score used the distance measurement as well as the local shape index for each of the control points. A summary of the experimental results is shown in Table 5.4. The best matching occurred when both surface matching and shape index were used for the matching score. All the errors were due to changes in facial expressions, which actually changed the 3D shape of the model. Table 5.5 summarizes the errors with respect to two variations: expression and pose.

Table 5.4: Second test matching error rates.

	Best Match Classification
ICP Only	4.4%
ICP + Shape Index	3.5%

Table 5.5: Distribution of matching error from a total of 113 test scans in the second experiment.

	Frontal	Profile	Total
w/o smile	0	0	0
w/ smile	1	3	4
Total	1	3	4

Figure 5.31 shows two examples of difficult 2.5D scans that were correctly matched by the arbitrary pose algorithm to the 3D model. Figure 5.32 shows two examples of difficult 2.5D scans that were incorrectly matched.

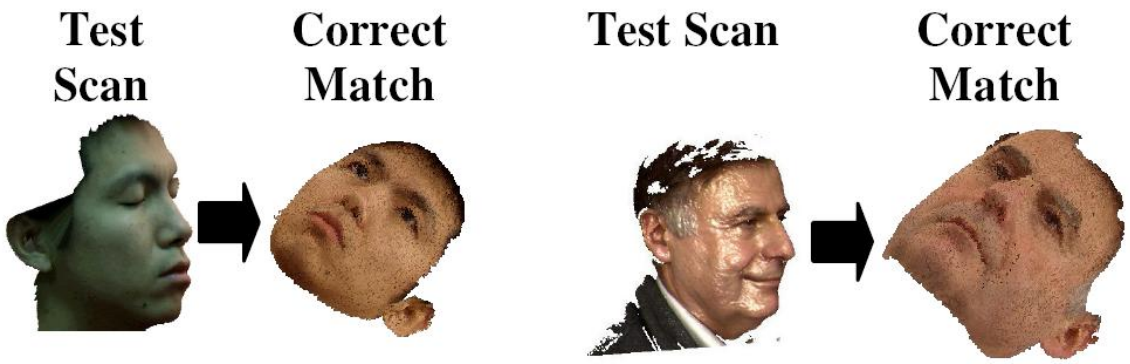


Figure 5.31: Examples of correctly matched test scans. Notice the changes in lighting, pose and facial expression.

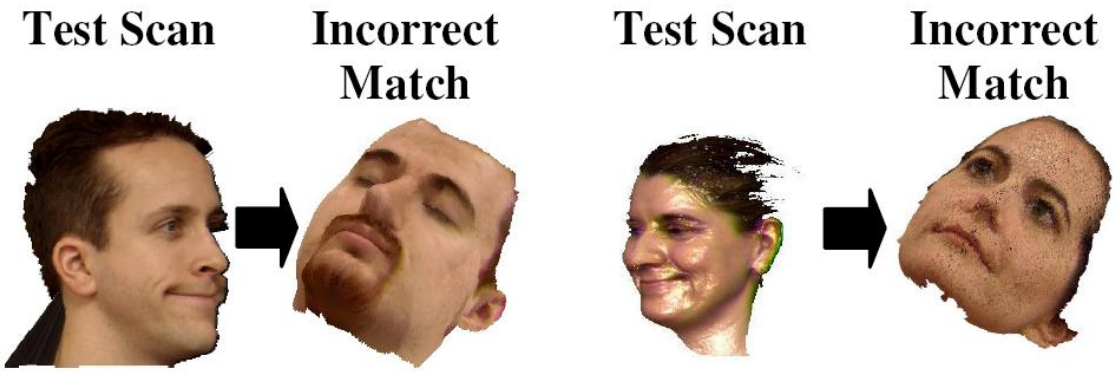


Figure 5.32: Examples of incorrectly matched test scans.

Chapter 6

Toward a Canonical Face Format

This chapter describes the development of a Canonical Face Depth Map (CFDM), which provides a normalized format for 3D face data. The CFDM is created by establishing a face-based coordinate system and then reprojecting the 3D data onto an orthogonal data structure that is fixed with regard to scale and resolution. The orthogonal depth map can be represented as a function of z in terms of x and y ($z = f(x, y)$) as was described in Section 3.6. The normalized CFDM format is demonstrated to improve the performance of a SAM-based matching system, to reduce the memory required to store a scan, and to help clean up holes in the face scan data. The CFDM can also be used as a preprocessing step to provide normalized data for matching algorithms such as PCA and correlation.

This chapter demonstrates that the CFDM can serve as an automated first step to the combined CFDM/PCA algorithm provided with the FRGC. Results indicate that this PCA algorithm produces very good matching results while eliminating the need for manual normalization (in the PCA algorithm). In addition, the CFDM performs well as a preprocessor for a correlation algorithm that requires the probe data to have the same scale and resolution as the model. Experimental results show that

combining the CFDM with the correlation algorithm provides a reasonable matching system as well as an anchor point localization algorithm.

In Chapter 5, two faces are compared to each other by using a two step surface alignment algorithm. This alignment procedure searches for a rigid transformation $R \in \mathcal{R}$ that minimizes the distance between a set of control points on the query scan to the surface of the model. If the two scans have large variations in size then the alignment algorithm may fall into different local minima, causing different values for the rigid transformation.

In other words, using the ICP algorithm to align two arbitrary surfaces may result in more than one possible solution transformation. This is not a problem when the alignment algorithm is only used for matching because the corresponding SAM scores are also large and the query scan is properly rejected. However, it may be useful to have an alignment algorithm that aligns faces in a standard (canonical) way.

In developing a canonical face-based coordinate system, the goal is to find a function $f^c|S \rightarrow \mathcal{R}$ such that $\forall s_1, s_2 \in S$ where $s_1 \sim s_2$ with rigid transform $R \in \mathcal{R}$ gives:

$$R = f^c(s_1) f^c(s_2)^{-1} \tag{6.1}$$

A surface $s \in S$ is in its canonical format if $f^c(s) = I$

The CFDM format is a face-based coordinate system that is tolerant to variations in pose and noise. A coordinate system transformation is defined by six degrees of freedom (x_0, y_0, z_0, roll, pitch, yaw). The canonical function (f^c) developed in this chapter is a two steps process. The first step is to establish the mid-line face plane by applying the 3D face alignment algorithm (described in Chapter 5) to a 3D face scan and aligning the scan to its mirror image. The 3D face alignment algorithm is robust

to changes in pose and to noise, and will reliably establish the mid-line face plane. The mid-line plane is used to limit two (roll and yaw) of the six degrees of freedom required to establish a face-based coordinate system. The second step in defining the face-based coordinate system is to calculate the third rotational degree of freedom (pitch) by fitting a parabolic cylinder to the surface of the face along the mid-line plane. The axis of this cylinder is used to define the pitch of the head.

Once the face-based coordinate system is properly defined, the next step in creating the CFDM is to establish an orthogonal depth map from the nose tip into the face scan. This depth map is perpendicular to the xy plane and allows for consistent re-sampling of the surface of the 3D face. Linear interpolation between the points in the scan is used to fill in the gaps. This re-sampling allows for the normalization of the face in terms of scale and resolution, providing the standardization for the CFDM.

Section 6.1 provides a background in canonical faces. Section 6.2 describes how the face-based coordinate system required for the orthogonal projection normalization is developed. Section 6.5 discusses properties of the CFDM, while Section 6.6 demonstrates how the CFDM can be used as a preprocessing step for other algorithms.

6.1 Background

The canonical format (CFDM) described in this chapter has two main components: a face-based coordinate system and a normalized feature vector based on an orthogonal depth map. The face-based coordinate system developed in this dissertation is similar to the one presented by BenAbdelkader and Griffin [14]. However, their approach [14] relies on just seven manually selected anchor points to properly determine the location of the coordinate system. In contrast, this dissertation uses ICP to generate a robust bisection of the face and uses high sampling rates to develop a good representation

of the mid-line plane. The CFDM developed here does not assume that the nose is the closest point to the camera, which is a weakness of other face-based coordinate systems, such as the approach offered by Malassiotis and Strintzis [101]. Assuming the nose tip is the closest point to the camera is particularly problematic when there is a possibility of spike noise in the data or large hair styles.

The CFDM provides a consistent feature vector that can be used to apply space transformation algorithms such as PCA. The CFDM is similar to the mesh model presented by Xu *et al.* [144], which uses the nose tip as the origin and then conducts a rigid, triangular re-sampling of the data that can be done at different resolutions. PCA is also used in the base algorithm of the Face Recognition Grand Challenge (FRGC) [118]. In this case, 3D faces are normalized using techniques similar to those for resizing 2D faces, which creates images with the same between-eye-center distance.

The face-based coordinate system defined for the CFDM requires the calculation of the mid-line plane, which is assumed to be similar to the plane of symmetry. A simple method for determining the axis of symmetry using low frequency filters is suggested by Marola [103]. The centroid of the object in question is determined and all possible axes of symmetry that pass though this centroid are examined. A simple coefficient of symmetry is used to determine the best axis. One problem with using these filters to find symmetry is that low frequency filters tend to blur salient regions that may be used to determine symmetry. Scognamillo *et al.* [128] first enhanced the low frequency salient regions using edge detection, and then a vertically-oriented Gaussian filter was convolved to find the location of the axis of symmetry. The orientation of this Gaussian filter could be adjusted to find different axes of symmetry. This work was done primarily on gray scale faces, but the authors also demonstrated that this approach worked even on gray scale images of symmetric dots. A major assumption in [128] is that the axis of symmetry must pass through the centroid; however, this

may not be true for the face mid-line depending on pose and occlusion. The general concept of object symmetry is also explored by Zabrodsky *et al.* [149], who examine various methods for determining the symmetry of an image. The algorithm proposed in [149] examines all possible axes of symmetry and determines the best axis based on a Symmetry Distance score. This approach is interesting because it looks for a 3D axis of symmetry on a 2D image, where the third dimension is the gray scale (depth) value. However, the algorithm in [149] is prohibitively slow even when the authors implement a multiple resolution scheme to increase the speed and decrease the search space.

There is also a body of research examining the symmetry of the brain in MRI images. This plane of symmetry is called the Mig-Sagittal plane and is important for registering different slice images of the MRI. Liu *et al.* [96] determined the symmetry of a human brain scan by convolving the mirror image of a (virtual) slice of the head and rotating it about the center of mass of the image. The best correlation would be the best match. Another brain symmetry algorithm presented by Tuzikov, Colliot *et al.* [134] uses the ellipsoid of inertia to calculate the initial location of the symmetry plane and then optimizes the estimate with a search.

For this dissertation, the plane of symmetry is established by aligning a face scan with its mirror image using the ICP-based face alignment algorithm described in Chapter 5. Pan and Wu [116] and Zhang *et al.* [150] also found the axis of symmetry using the ICP algorithm, with similar results. However, these papers assume that the face is vertical and no trimming is used to account for noise.

6.2 Face-Based Coordinate System

This section explores methods for identifying a face-based coordinate system, which requires accounting for six degrees of freedom: roll, pitch, yaw and the origin

(x_0, y_0, z_0). Having a well defined face-based coordinate system makes it simple to transform data into whatever format or pose an algorithm requires. The goal of creating a face-based coordinate system is to establish a format that is easily calculated and repeatable for the same face. In addition, face-based coordinate systems should be robust to noise in the data, and it should be pssible to estimate the face-based coordinates even when some of the face is occluded or missing. Ideally, it will also be possible to visually estimate the face-based coordinate system by using 3D data, which would allow databases of different modalities to be registered with each other.

Section 6.2.1 introduces a basic approach for establishing a face-based coordinate system by using identified anchor points. Section 6.2.2 extends this method by using the surface alignment algorithm described in Chapter 5 to finely align a scan with a baseline model. Section 6.2.3 describes the canonical method for using more robust face-based features, such as the mid-line of the face, which do not require a generic 3D model.

6.2.1 Anchor Point Alignment

The majority of 3D face-based coordinate systems are based on the selection of anchor points. The simplest of these approachs is a single point transformation, where the 3D scan is assumed to already be in a known pose (normally frontal); this pose accounts for three degrees of freedom: roll, pitch and yaw. The remaining three degrees of freedom are accounted for by translating the scan using a single point, such as the tip of the nose or the centroid.

Single point alignment is by far the easiest method for normalizing a scan, but it is also the most susceptible to minor variations in the data, such as changes in pose, specularities, etc. Even a seemingly frontal database can have inter-pose variations as large as 15 degrees (as described in Section 5.3.3). In addition, many single-point

methods use heuristics (such as the closest point to the scanner) to find the tip of the nose. These types of heuristics are not robust to spike noise and fail completely if the nose is partially occluded or if there is a hole in the scanning data.

A multiple anchor point alignment method, which is a little more robust, requires a minimum of three known, non-co-linear, anchor points to establish all six degrees of freedom in a face-based coordinate system (see Section 3.5.2). In a three point system, the frontal pose assumption does not have to be made. However, the anchor points must be chosen such that any errors in selecting the points are minimized. For example, if the selected points are close together the transformation will be biased toward a small section of the scan, which can cause errors in more distant areas of the scan. Additional points can be added to the calculation to minimize the error caused by a single point; BenAbdelkader adn Griffin [14] used seven points to estimate a plane on the face.

6.2.2 Face Model Normalization

Although anchor points can be used to establish a face-based coordinate system (as described in the previous section), minor errors in selecting these points can translate into major variations in the coordinate axes. A more robust method for establishing a face-based coordinate system is to use the face alignment algorithm described in Chapter 5 to align every face in the database with a generic face or generalized model. The generic face could be randomly selected from the database, or even selected based on some similarity metric. A generalized face model such as in [5] could also be used. The face-based coordinate system for the generic face is then used to define the face-based coordinate system for the entire database.

The generic-face method for establishing a face-based coordinate system makes some fairly broad assumptions. First, it is assumed that the alignment algorithm can align any face with the generic face model. Second, it is assumed that the generic

face model will always align with a model from the same subject in the same way. However, with an arbitrary generic face model these two assumptions are not always true. For example, if the generic face model is of a large face then a smaller face scan may not fit properly (see Figure 6.1). In cases where the generic face model and the test face scan are of different sized faces, it is common for the nose of the smaller face to align properly but roll may be induced to make the smaller face fit with the wider one. It is also likely for more than one local minima to exist in this situation, and if the SAM is high then there are likely several equivalent minima. An ideal generic face model would minimize these types of errors (roll and local minima).

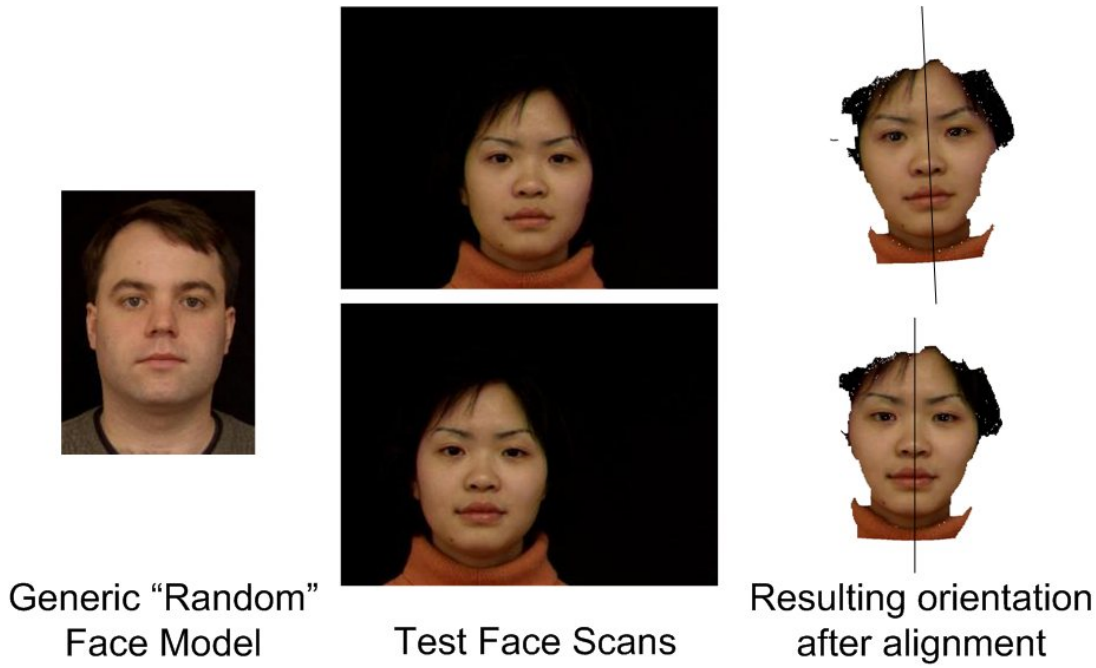


Figure 6.1: Example error where the generic model and the test scan subject do not have the same sized head, resulting in multiple local minima.

Two different experiments were used to explore normalized faces. The first used a randomly selected face model from the database and aligned all the other scans in the database to this model. The second used an average face model (see Figure 6.2), which was developed by transforming a set of training faces into a canonical format based on face symmetry (see section 6.2.3) and using the average of these faces as the

generic model. Overall, the average face did not fall into as many local minima but there were still scans that did not fit well to the model.

(a) Frontal View (b) Side View

Figure 6.2: 3D model of an average face generated using a Canonical Face Depth Map. This model can be used as a generalized target model for new scans.

One advantage of using a generalized model is that the ICP alignment algorithm is simple and fast. However, misalignment can occur when there are differences in size between the general model and the test scans. Another approach for creating a generalized face is to vary the model to match the size of the face. This variable model could account for variations in size, but increases the search space.

6.2.3 Defining a Canonical Coordinate System with Global Features

In addition to the use of anchor points and generalized face models, a third option for generating a face-based coordinate system is to identify robust and salient global features on the face that are consistent across all faces. One such feature is the mid-line plane of the face.

Defining the Mid-line Plane

There are two salient properties of the mid-line plane on a human face: the first is a high curvature profile ridge and the second is face symmetry. Since the mid-line has two salient properties, it is possible to estimate the mid-line location using two completely different methods.

In the case of the mid-line plane, the symmetry property is very accurate but breaks down if there is not enough surface to determine the symmetry (i.e., the face is turned too much) or if the face is not very symmetrical. Perceived asymmetries can be introduced by sensor noise or lighting conditions, or the face may not be particularly symmetrical. Simple external variations, such as changing expression (wink) or hairstyle, can cause asymmetries. Asymmetries can also occur due to injuries such as a broken nose or a scar. Research has also shown that there are natural asymmetries that occur in humans during their development [88]. All of these factors make finding the plane of symmetry on the face more difficult. However, in a profile image the mid-line plane can be detected using the prominent profile points. It is important to note that the profile ridge, plane of symmetry and mid-line plane may all represent slightly different planes of the face. However, the planes should be similar enough that together these properties make the mid-line plane a useful feature as a baseline for establishing a face-based coordinate system.

One method for detecting the mid-line of the face relies on the high-curvature profile ridge that defines much of the mid-line. The profile ridge is defined as the points on the surface of the profile projection of the face (see Figure 6.3). These points are fairly salient, since much of the profile plane protrudes from the surface of the face (e.g., the nose and chin). Because of the high relative curvature of the nose in the y direction, changes in estimating the profile direction (yaw angle) will result in similar profile locations (see Figure 6.4). In fact, if the head is known to be vertical, meaning at minimum roll and pitch values, then these high curvatures can

be used to generate an estimate of the location of the mid-line of the face. However, as Figure 6.4c shows, errors can occur if there are variations in the pose of the scan.

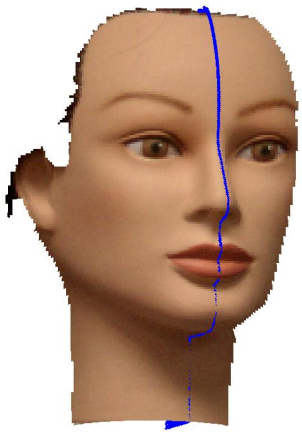
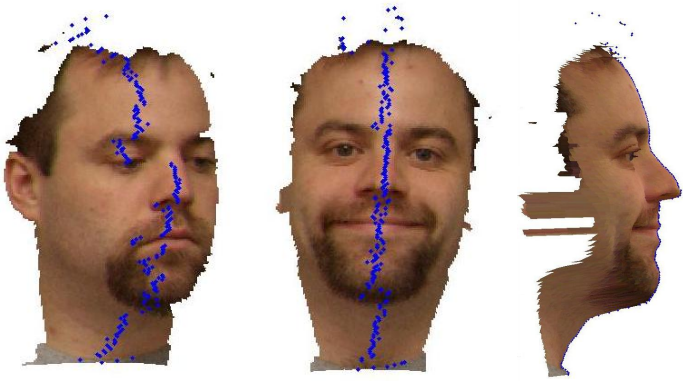


Figure 6.3: Profile ridge. Each point represents the largest z value for every row in the scan.



(a) Non-Frontal Scan (b) Frontal View (c) Profile Projection

Figure 6.4: Projection profile examples.

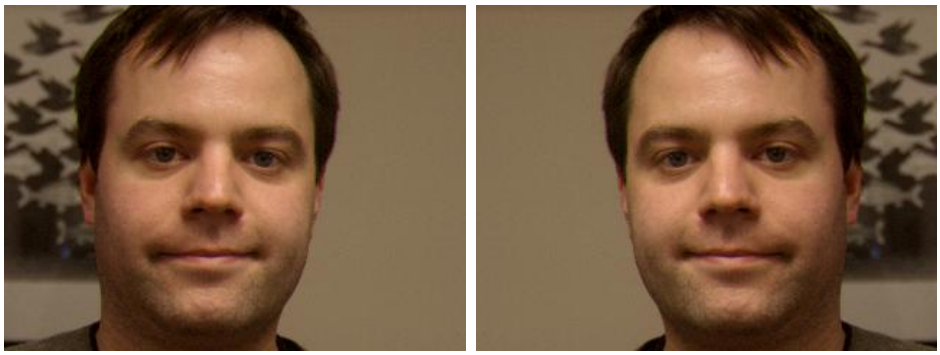
The second salient feature of the mid-line plane is face symmetry. For 3D faces, the mid-line plane can be calculated robustly by matching a face scan with a mirror image of itself (see Figure 6.5) using the surface alignment algorithm described in

Chapter 5. For this dissertation, a mirror image of an object is formally defined as follows:

$$P_M = M \cdot P \tag{6.2}$$

where P represents the original 3D points in the scan, P_M indicates the mirror points and M is the mirror transform:

$$M = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



(a) Original Scan (P) (b) Mirror Transformed Scan (P_M)

Figure 6.5: Example mirror transformation.

The mirror image is a left-hand projection of the original scan. The process for aligning the original and the mirror scans is the same as for aligning any two faces and uses the alignment algorithm described in Chapter 5. The resulting transformation between the two scans is defined as the mirror transform and it is denoted by T_m.

With the mirror transform, the mid-line plane can easily be calculated. First, an arbitrary point p is chosen. This point does not have to lie on either scan but must

not lie on the mid-line plane. Using this point, the surface normal $\vec{N}$ of the mid-line plane is calculated as:

$$p_m = T_m M \cdot p \tag{6.3}$$

$$\vec{N} = \frac{pm - p}{|pm - p|} \tag{6.4}$$

If a, b, and c are the x, y, and z components of the surface normal and d is the distance along the surface normal from the origin to the plane, then the following equation defines the plane:

$$ax + by + cz + d = 0 \tag{6.5}$$

Figure 6.6 shows an example the face mid-line plane. The plane defined by Equation 6.5 accounts for two of the six degrees of freedom (yaw and roll).

Figure 6.6: Example location of the mid-line plane relative to a face.

By assuming that the pitch angle is negligible, the face scan can be normalized using only the mid-line plane and maintaining the existing pitch; this may produce a reliable face depth map. To test this hypothesis, the MSU database was transformed

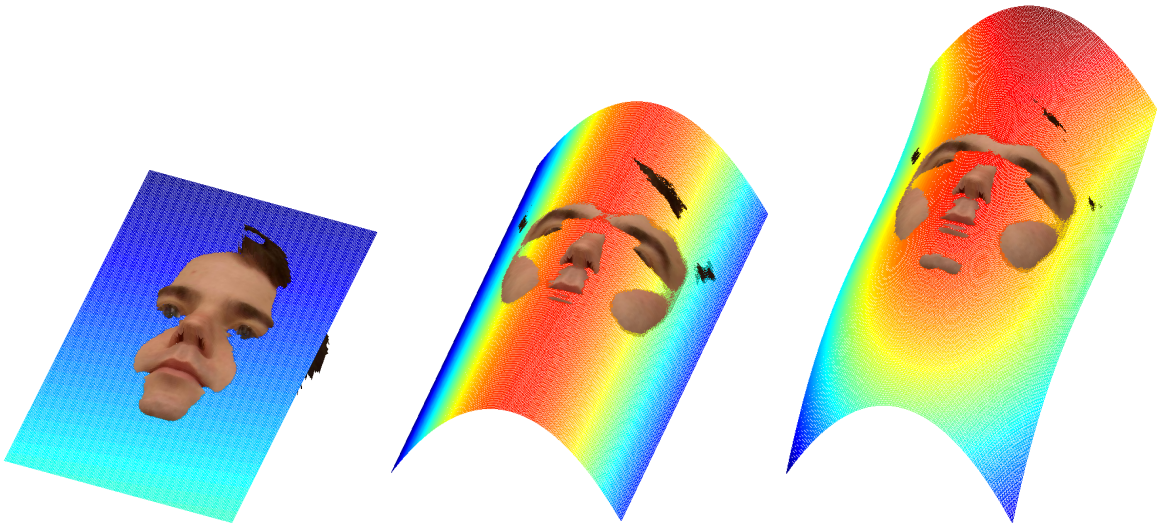
into the mid-line corrected depth map (at a resolution of 400×700) and then each scan was aligned to every other scan from the same subject. After alignment, the roll, pitch and yaw differences were calculated, as shown in Table 6.1. The results in Table 6.1 demonstrate that the mid-line normalization removes the effects of the roll and yaw rotation, but does not improve the pitch rotation.

Detecting Surface Pitch

The mid-line plane can be accurately estimated using the plane of symmetry of the face and the surface alignment algorithm. However, the mid-line plane only accounts for two of the six degrees of freedom. The origin can also be estimated by selecting a robust point, such as the tip of the nose. Yet, this still only accounts for five of the six degrees of freedom. The pitch (tilt) of the head is still problematic. This section explores methods for robustly calculating head pitch. These methods assume that the mid-line plane is already calculated and uses this plane to help restrict the direction of the pitch rotation axis to ensure that it is orthogonal to the xy plane.

The experiments described here fit a simple surface to the face and use the surface parameters to calculate the pitch. Figure 6.7 shows three parametric surfaces, a plane, a parabolic cylinder, and a quadratic surface, fit to a single face scan. Each of these surfaces was applied to the database of scans and the results are shown in Table 6.1. The goal of this surface fitting is to calculate a robust and repeatable measurement of the pitch of the surface of the face. All the rotations shown in Table 6.1 have values close to zero. Figure 6.8 shows the variance for each of the pitch correction methods described in this section.

From the results in Table 6.1 and Figure 6.8, it seems that the plane model is too simple for accurate pitch detection, while the quad model is over-fitting the data. The parabola seems to be the best surface fitting approach. A similar model has also been used in 2D pose correction for head tracking [32].



(a) Plane Surface $ax + by + c$ (b) Parabolic Cylinder Surface $ax^2 + bx + cy + d$ (c) Quadratic Surface $z = ax^3 + bx^2y + cxy^2 + dy^3 + ex^2 + fxy + gy^2 + hx + iy + j$.

Figure 6.7: Parametric surface fitting using forced symmetrical surface fitting.

Table 6.1: Improvement of mid-line normalization over database roll, pitch, and yaw differences. In the first row, the original roll, pitch, yaw and distance statistics represent the variations between different scans of the same subjects within the original database (this first row is taken from Table 5.2). Each subsequent row represents the variation after the different normalization techniques. The Plane, Parabolic Cylinder, Quadratic, and Average are all applied after the baseline symmetry is established.

	Roll	Pitch	Yaw	Distance
Original	$0.36 \pm 0.49^\circ$	$1.62 \pm 2.25^\circ$	$1.52 \pm 1.99^\circ$	$63.69 \pm 66.08mm$
max	3.11°	16.03°	17.30°	$431.97mm$
Symmetry	$0.01 \pm 0.53^\circ$	$0.00 \pm 3.44^\circ$	$0.00 \pm 0.56^\circ$	$\pm 3.102.14mm$
max	2.49°	16.68°	3.20°	$34.7mm$
Plane	$0.01 \pm 0.52^\circ$	$0.10 \pm 3.15^\circ$	$0.02 \pm 0.54^\circ$	$\pm 5.122.48mm$
max	2.80°	33.97°	1.81°	$56.8mm$
Parabola	$0.02 \pm 0.53^\circ$	$0.10 \pm 2.59^\circ$	$0.00 \pm 0.50^\circ$	$2.06 \pm 3.33mm$
max	2.46°	32.50°	1.75°	$42.20mm$
Quad	$0.00 \pm 0.53^\circ$	$0.01 \pm 3.42^\circ$	$0.01 \pm 0.51^\circ$	$\pm 12.675.31mm$
max	2.56°	16.07°	2.51°	$75.85mm$
Average	$0.01 \pm 0.58^\circ$	$0.01 \pm 2.01^\circ$	$0.00 \pm 0.79^\circ$	$2.90 \pm 7.81mm$
max	2.38°	12.50°	2.85°	$73.95mm$

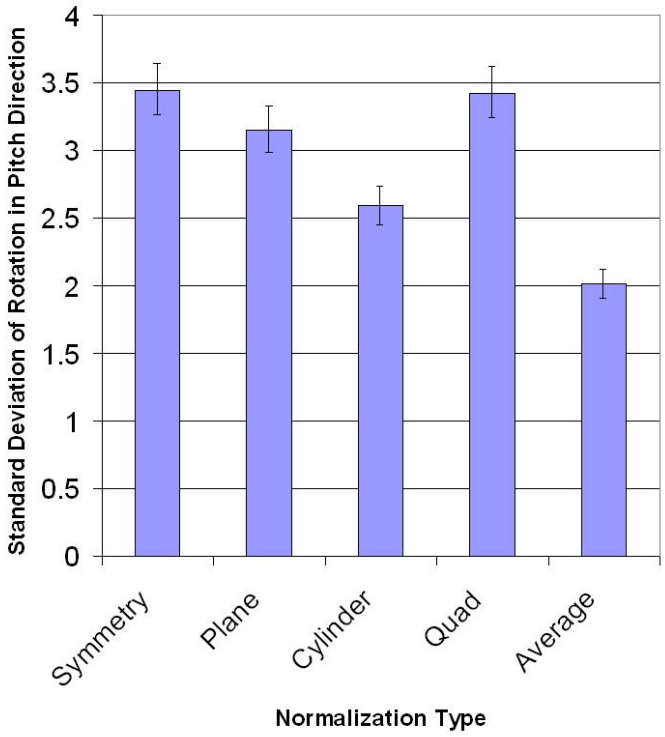


Figure 6.8: Comparison of pitch standard deviation over different fitting techniques.

As a comparison, the results for surface fitting and pitch normalization are also given for the average face model described in Section 6.2.2. Fitting a surface to an average face seems to be the most reliable pitch correction method, but this method is sure to break down if it encounters a face that is significantly larger or smaller than the average. Also, it should be noted that the average face is generated from the same dataset, so it is probably biased toward this data.

Overall, the approaches shown in Figure 6.8 are accurate for changes of up to $\pm 2^{\circ}$ in the pitch axis, which is much larger than the $\pm 0.5^{\circ}$ for the other two axes.

6.3 Canonical Face Depth Map

This section describes the use of face-based coordinate systems to establish a Normalized Face Depth Map (NFDM). Any face-based coordinate system can be used to

develop a normalized format. However, if the face-based coordinate system is highly repeatable over faces and scans that differ in pose, lighting, scanning noise, holes, artifacts, and expression, then this face-based coordinate system can be defined as a canonical format. In other words, a Normalized Face Depth Map that uses a canonical, face-based coordinate system is called the Canonical Face Depth Map (CFDM).

Regardless of which face-based coordinate system is used, the establishment of a CFDM requires that the face data be re-sampled using an orthogonal projection, as described in Section 3.6. The orthogonal projection produces a depth map of constant size and scale and the CFDM allows a database of face scans to be standardized and aligned. This standardization can act as a preprocessor to other 2D and 3D face processing algorithms and has the the following advantages:

- Single matrix representation to reduce memory
- Registration of the faces
- Standard feature vector size
- Normalized pose

Taking advantage of these attributes provides the following benefits:

- Fast correlation between 3D face data
- Trivial feature space transformations (ICP, wavelets, Fourier transforms, etc.)
- Better data comparison for matching

The Canonical Face Depth Map consists primarily of a single matrix depth map, which is standardized so that each increment of the index moves a static distance in the x and y direction (i.e., constant step value). This allows a depth map to store the x, y and z information in a single matrix (see Section 3.6). There is also a very large (or very small) value that is used to signify a bad point. In addition to the depth map structure, the Canonical Face Depth Map has a standardized face-based coordinate system.

Table 6.2 and Figure 6.9 outline the algorithm for converting raw scanner data into the Canonical Face Depth Map. A face-based coordinate system (from Section

6.2) is established first. Once the face-based coordinate system is properly detected, an orthogonal projection of the data is made onto the xy plane. Some difficulties may arise when this projection is made, and proper sub-sampling of the data may be necessary in order to fill the entire depth map.

Table 6.2: Steps for constructing a Canonical Face Depth Map.

1. Find key anchor points on the face.
2. Calculate the mid-line plane using face symmetry.
3. Calculate the head-pitch by fitting a parabolic cylinder to the face.
4. Transform the data into the face-based coordinate system.
5. Orthogonally project the depth values onto the Depth Matrix.

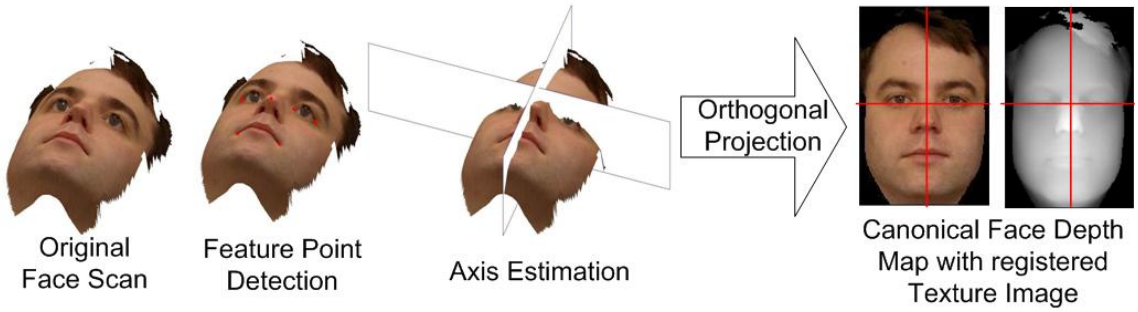


Figure 6.9: Process for generating a Canonical Face Depth Map.

One potential drawback of a Canonical Face Depth Map is that this format can only store information from one direction, so some information may be occluded or sub-sampled due to the angle. For example, points on the side of the nose will be less dense than points in the same area on the cheek because the normal direction for the cheek is in the direction of the projection plane. This loss of information in areas of high z gradient is a problem with any depth map and is problematic for 2.5D storage formats.

6.4 Robustness Analysis

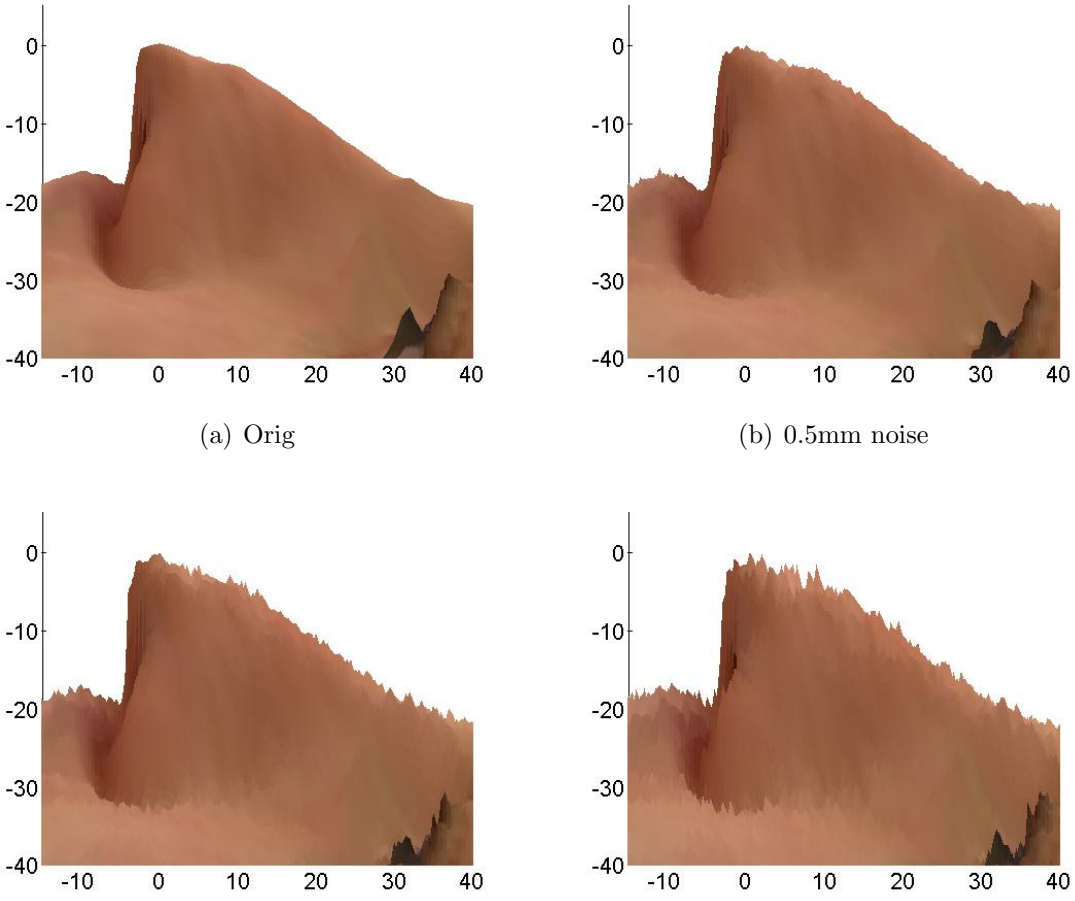
The CFDM described in the previous section was designed to be robust. This section tests the design to demonstrate its robustness to noise and holes in the data. In the

first experiment, Gaussian noise is added to the z component of the MSU dataset. The noise is generated in Matlab using a normal distribution and then multiplied by one of three noise amplitudes (0.5mm 1.0mm and 1.5mm). This range was chosen because the Minolta scanner accuracy is around 0.5mm and the the SAM verification threshold is around 1.0mm. Any lower errors should would not be significant, and any higher level of noise would result in inaccurate matching. Figure 6.10 shows some examples of these noise levels on a typical scan. The results for the noise experiments are shown in Table 6.3. These results show that the mean roll, pitch and yaw stay around zero degrees, but as the amount of noise increases the standard deviation of these values also increase. Fortunately this increase is very small, indicating that the CFDM is robust to Gaussian noise.

Table 6.3: Error with respect to noise

	Roll	Pitch	Yaw	Distance
Parabola	$0.02 \pm 0.53^{\circ}$	$0.10 \pm 2.59^{\circ}$	$0.00 \pm 0.50^{\circ}$	$2.06 \pm 3.33mm$
max	2.46°	32.50°	1.75°	$42.20mm$
$0.5mm$	$0.01 \pm 0.56^{\circ}$	$0.06 \pm 2.37^{\circ}$	$0.01 \pm 0.89^{\circ}$	$2.57 \pm 4.84mm$
max	2.46°	23.98°	5.30°	$63.95mm$
$1.0mm$	$0.00 \pm 0.62^{\circ}$	$0.07 \pm 2.49^{\circ}$	$0.01 \pm 1.00^{\circ}$	$3.80 \pm 5.99mm$
max	2.33°	31.47°	5.90°	$65.73mm$
$1.5mm$	$0.00 \pm 0.67^{\circ}$	$0.06 \pm 2.32^{\circ}$	$0.01 \pm 1.03^{\circ}$	$5.32 \pm 6.24mm$
max	5.23°	27.69°	7.82°	$62.65mm$

The second experiment is designed to test the CFDM against holes in the data. This experiment was tested on 36 subjects and 103 scans with ten holes per scan. The trial removes a Gaussian hole from the scan; this hole covers circular area of approximately 5mm radius taken from the region of the scan used by the surface alignment algorithm. Each hole removes a different number of pixels depending on the scale of the scan and whether the holes overlap with previous holes in the data. This process is repeated for a total of ten iterations, with each iteration adding another hole to the scan. Figure 6.11 shows an example of holes added to the scan. The graph displays the percentage of valid pixels removed from the entire scan by the



(a) Orig

(b) 0.5mm noise

(c) 1.0mm noise

(d) 1.5mm noise

Figure 6.10: Example noise on the surface of the face.

holes. The y axis shows the difference in the roll, pitch and yaw between the CFDM face processed without the holes and the CFDM face processed with the holes.

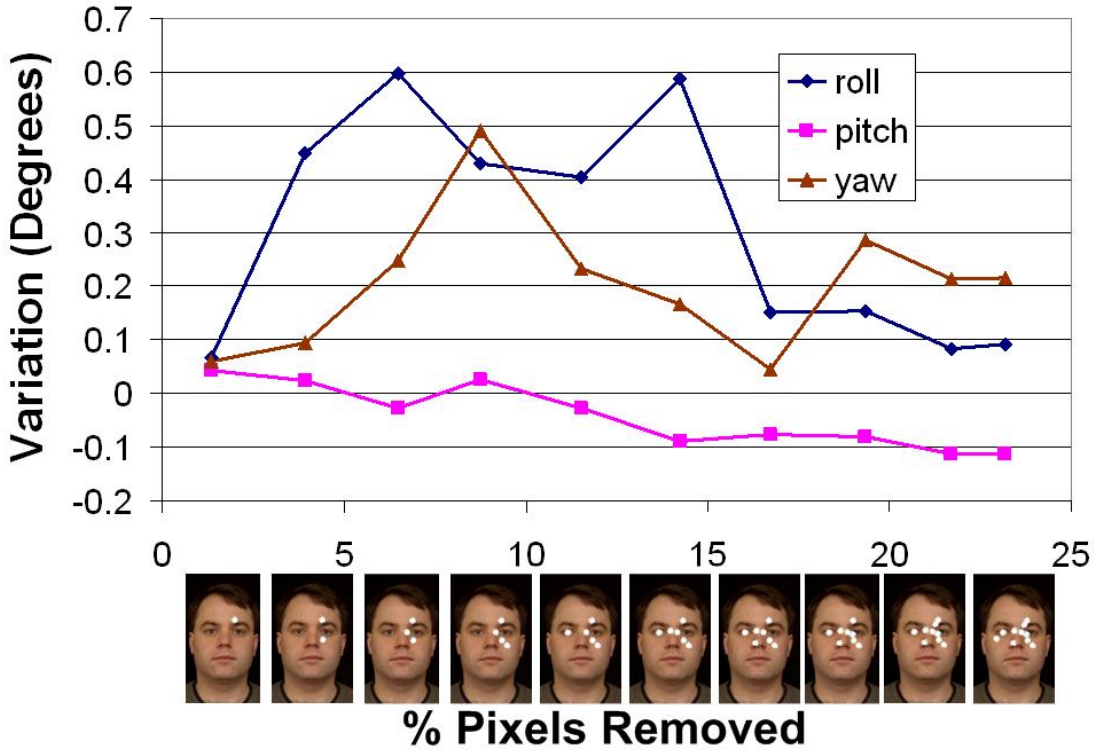


Figure 6.11: Example of holes on the surface of the face and how the angles in the canonical format change.

Figure 6.12 shows the results for the entire 1030 trials (103 scans with 10 iterations each). Notice that when less than 10% of the data is removed, the angle variation is no more than two degrees. As the number of holes increases, the system generates larger errors. In most cases, the angle difference is still well under 2 degrees of variation. However, there are cases where the errors spike, normally due to holes placed in key locations that interfere with anchor point detection or cause large asymmetries in the data.

Figure 6.13 compares the percentage of pixels removed against the distance to the origin. In this case, the origin is not as robust as the angles of rotation. The majority of points are well within $10mm$, but the origin can move around. This becomes

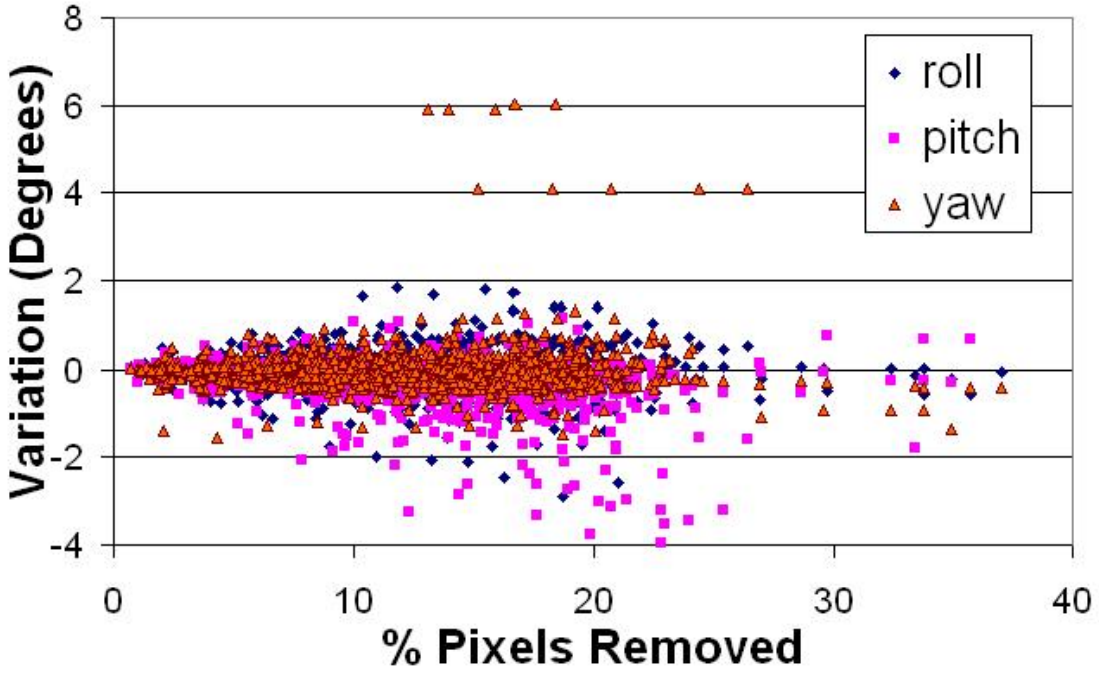


Figure 6.12: Comparison of how the holes in the face affect the CFDM angles.

clearer when looking at a breakdown of the x, y, and z components of the distance measure as shown in Figure 6.14. In this case, the depth value, or z component, of the origin transform is fairly stable (under 10 mm), and the y and z components have the most variation. Future work may need to focus on a method to improve the robustness of the origin to noise. However, for the current system this level of noise is not problematic because having 10% to 10% of holes in the data is unlikely. In any case, it is easy to calculate the number of holes in a scan and reject it if there are too many pixels missing.

This section has demonstrated that the CFDM is robust to Gaussian noise and holes in the data. In addition to these studies, extensive testing of the system (as described throughout this dissertation) has shown that the algorithm performs quite well in everyday uses.

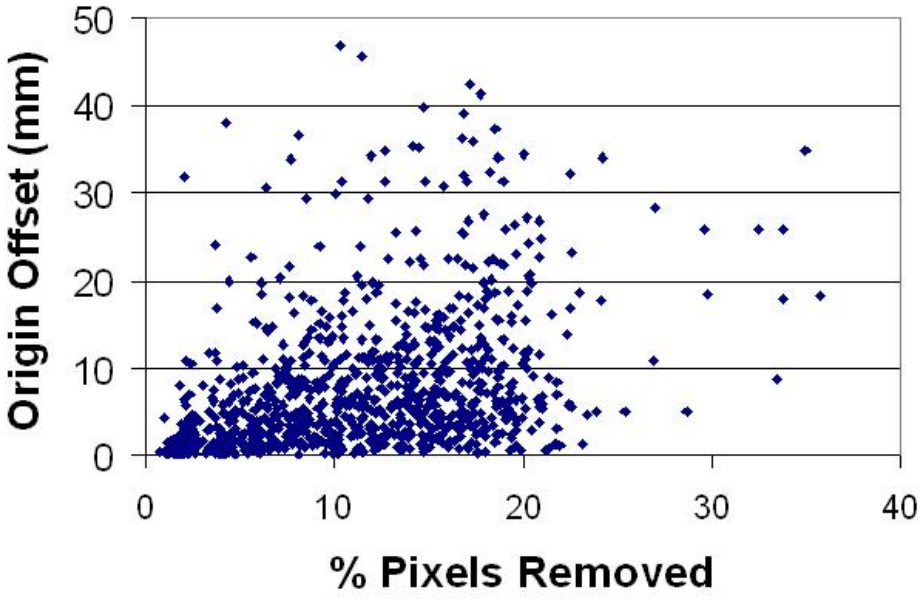


Figure 6.13: Comparison of how the holes in the face affect the CFDM origin.

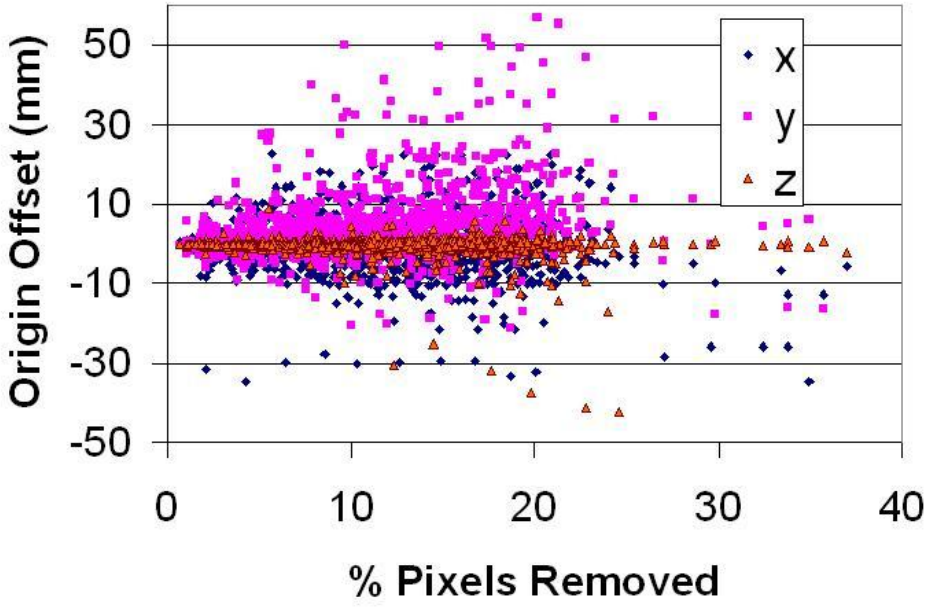


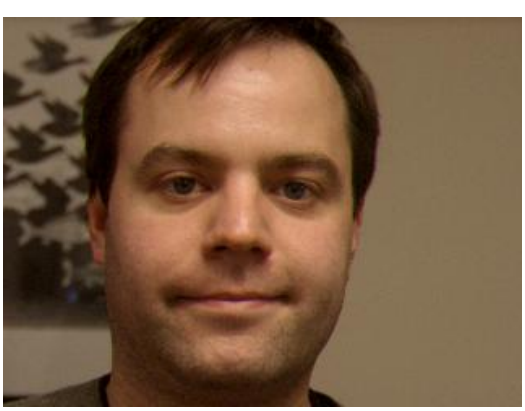
Figure 6.14: Comparison of how the holes in the face affect the CFDM origin relative to the x, y and z distances.

6.5 Properties of Canonical Face Depth Maps

Canonical Face Depth Maps have many advantages. The Canonical Face Depth Maps described in this dissertation are all sub-sampled using the orthogonal depth map, and this sub-sampling is at the same scale and dimension regardless of the input scan resolution. For example, the images in Figure 6.15 are at different camera offsets. The scans are normalized into the CFDM format (Figures 6.15b and 6.15d) and can then be directly compared. In contrast, 2D images have an unknown scale and must be normalized based on some common point distance, such as the distance between the eyes. However, different people have different between-eye distances, so the effect is a comparison of faces that must be scaled up or down to match each other.

Because all of the scans use a canonical point of view, the anchor point algorithm can be reapplied the CFDM faces. Figure 6.16 shows the offset difference between the manually selected points and the automatically generated points. Each image is paired with the same image from Figure 4.6 in Chapter 4. The right image represents the anchor point locations using the raw data and the left image represents the anchor point locations after applying the CFDM. Table 6.4 summarizes the results and shows that there is not much improvement over the original anchor point detection, and in some cases does not perform as well. However, overall there is more improvement using the CFDM and it does not significantly deviate from the original point location. Therefore the CFDM can help in point localization.

The facial mid-line plane of symmetry can be used to fill in holes in 3D data, since any point on one side of the mid-line plane should have a corresponding point on the other side of the plane. If a close point or region is not found on same side of the face, then a copy can be made from the other side to fill in the missing information. This method can be simplified by generating a depth map that is perpendicular to the plane of symmetry. For example, notice the hole around the eyebrow in the scan in Figure 6.17a. This hole can be filled using the mirror image (see Figure 6.17b).



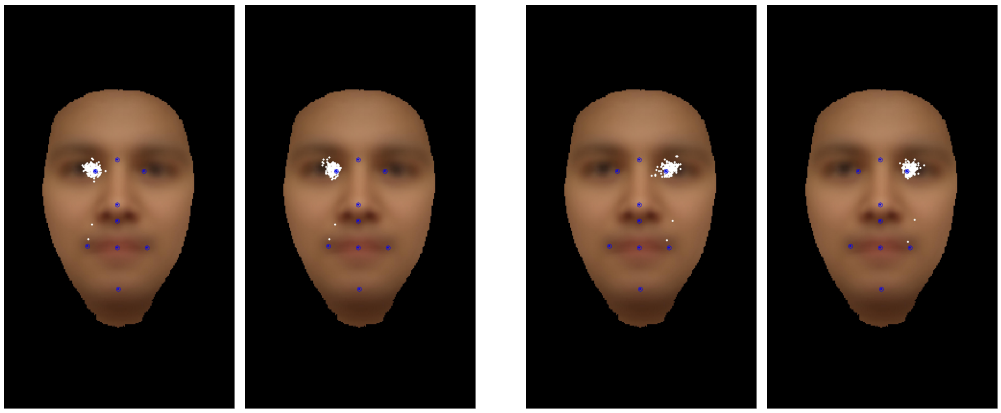
(a) Original Scan

(b) After Depth Mapping

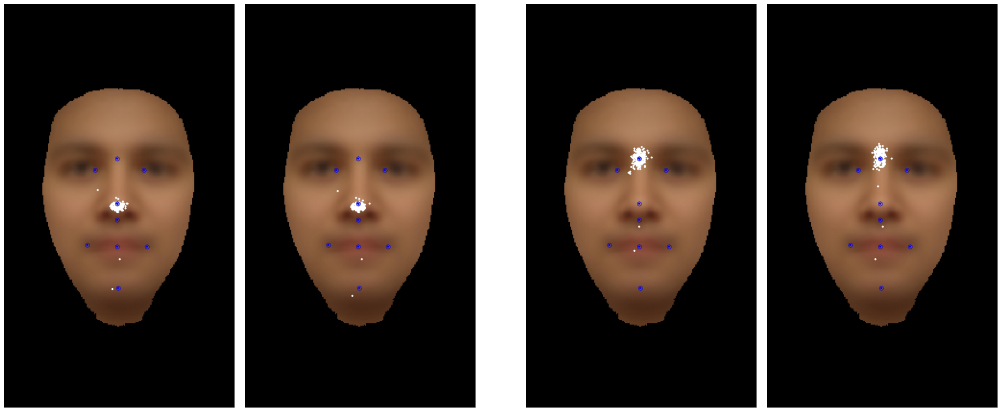
(c) Original Scan

(d) After Depth Mapping

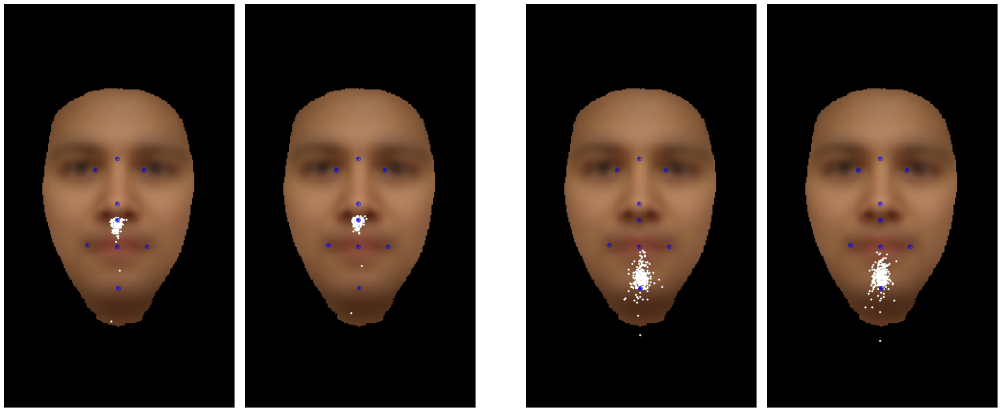
Figure 6.15: (a) and (c) are scans with different camera offset and resolution, (b) and (d) show how using a depth map normalizes the resolution.



(a) Raw (b) CFDM (c) Raw (d) CFDM



(e) Raw (f) CFDM (g) Raw (h) CFDM



(i) Raw (j) Nose Base (k) Raw (l) CFDM

Figure 6.16: Average face image with each of the manually selected anchor points (green) and the relative location of the automatically generated points (white) from the MSU 330 image dataset. The left images represents relative the anchor point locations using the raw data and the right images represents the anchor point locations after applying the CFDM.

Table 6.4: Experimental Results for Frontal Anchor Point Detector before and after appling the CFDM.

	Before CFDM		After CFDM	
Point Name	Average (mm)	Standard Deviation	Average (mm)	Standard Deviation
Nose	4.1	5.1	4.0	5.4
Nose Bridge	4.7	6.0	4.8	6.4
Right Eye	5.5	4.9	5.6	4.8
Left Eye	6.3	5.0	6.0	4.7
Chin	11.0	7.6	11.7	7.3
Nose Base	4.9	6.6	4.1	5.9
Mouth	4.0	6.7	4.0	6.0
Left Mouth	6.7	9.3	5.4	6.7
Right Mouth	6.9	8.6	5.4	6.8

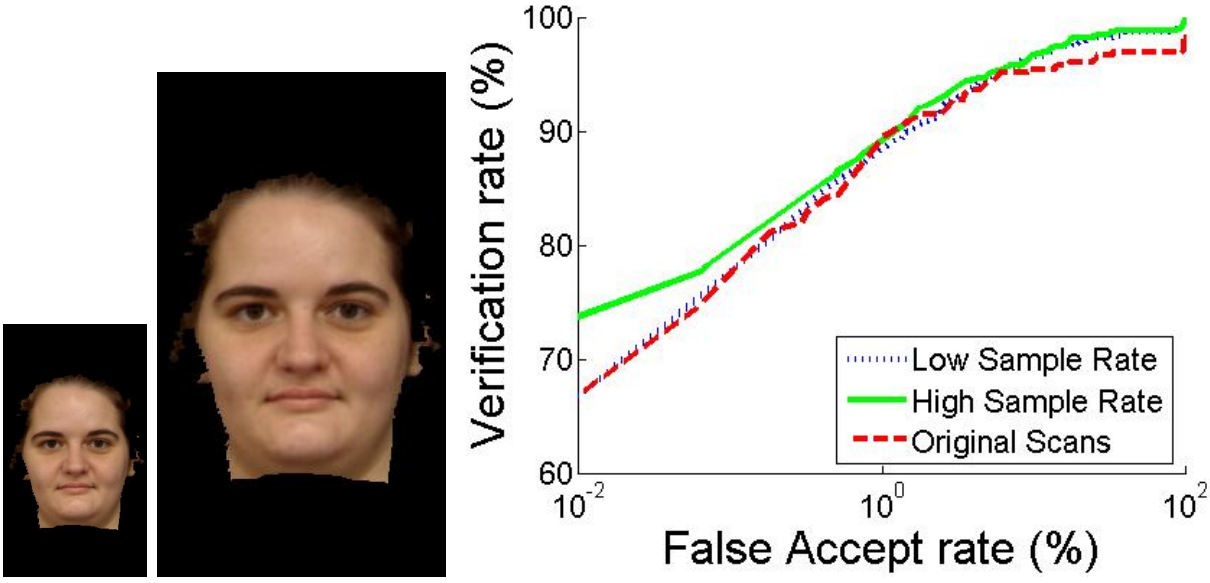
Notice that the projected color does not match; this hole-filling algorithm can also re-sample the color image from the original scan if these data points are available.

(a) Scan with holes (b) Scan after hole filling

Figure 6.17: Example of symmetric hole filling. Note that the color is maintained with the copied hole points.

Another advantage of using a Canonical Face Depth Map is that the depth map can be sampled at any resolution (see Figure 6.18a and b). Lower resolutions can be used to speed up some algorithms, while higher resolutions can be used to increase accuracy. For example, the hybrid Besl and Chen ICP algorithm described in Section 5.5 is used as a way to combine the speed of the Besl algorithm with the performance of the Chen algorithm. Similar algorithm accuracy can be achieved by super sam-

pling the CFDM and creating a highly dense target scan in which ICP can operate. Figure 6.18c shows two ROCs for face depth maps sampled at different scales. The higher sampling rate will slow down the matching algorithm, but it also increases the accuracy of the algorithm.



(a) Low Sample Rate (200×350)

(b) High Sample Rate (400×700)

(c) ROC Performance

Figure 6.18: Depth maps sampled at two different resolutions using the MSU database. The higher resolution data improves ICP performace by allowing for small incremental changes during each iteration of the algorithm. The increased surface matching performs slightly better, as seen in (c).

Because the CFDM uses an orthogonal project, it is possible to reduce the memory requirements for storage of the file. One goal for the CFDM is to be able to store it on a smart card. Table 6.5 shows two sets of bar graphs. In each graph a line representing the traditional smart card size is given as a goal (512 KB [66]). Note that even in its most compressed formats it is unlikely that a smart card will be able to store a face scan in its raw format. However, the CFDM does not need to store the x, y or even flag component of the face scan. This allows the face to be stored in a

Table 6.5: Results of file format memory comparison. A standard smart card has 512 kilobytes of memory [66]. This is not enough memory to store the raw image of a face scan. Because the CFDM requires only one matrix to represent the 3D surface there are many formats of the CFDM that can fit onto a smart card. It is even possible to fit more than one template on a card. All file formats under the 512 kilobyte memory goal are shown in bold.

	Raw 320 x 240 Scans	400 x 700 CFDM
Minolta (cdm)	1.58 MB	n/a
FRGC (abs / ppm)	20.84 MB	4.39 MB
Compressed FRGC	0.88 MB	**0.37 MB**
Matlab (mat)	8.64 MB	3.08 MB
Compressed Matlab	1.70 MB	1.02 MB
3BX	4.53 MB	n/a
VRML (wrl / jpg)	1.70 MB	n/a
Binary ppm	n/a	1.68 MB
Compressed Binary ppm	n/a	**0.24 MB**
Jpeg	n/a	**0.16 MB**

format that is much smaller than the goal of 65 Megabytes even at the high 400×700 resolution. It is even possible that more than one scan could be stored on a single memory card allowing for non-rigid variations, such as smiles.

6.6 Preprocessing using Canonical Face Depth Maps

The output of the Canonical Face Depth Map algorithm is trivially transformed into a data structure with the same format as the original face scan. Maintaining the data format allows the Canonical Face Depth Map to be used as a preprocessor to other algorithms. This section presents two common image processing algorithms, correlation and PCA, and applies the CFDM developed in this dissertation as a preprocessor for these algorithms. Correlation and PCA were chosen because they had potential for effective identification and they perform best if the input data is normalized and of fixed size resolution. Section 6.6.1 describes the correlation

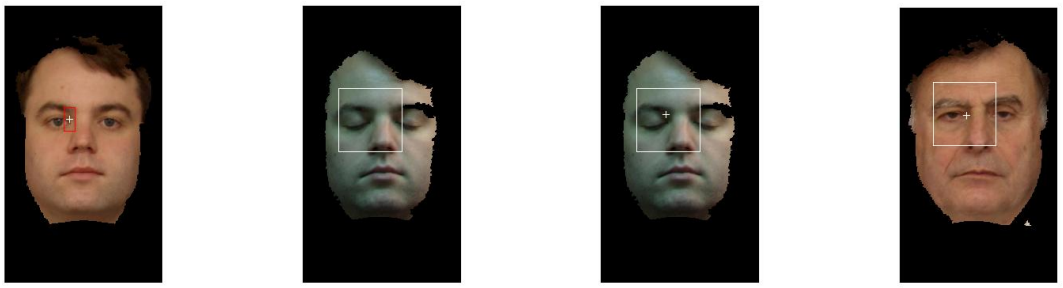
algorithm in detail and shows how it can be used for face verification as well as anchor point localization. Section 6.6.2 describes how the CFDM preprocessor was used with the baseline PCA algorithm [20] implemented for the Face Recognition Grand Challenge.

6.6.1 Correlation

A normalized 3D face correlation algorithm has been developed for this dissertation. For each window region, the points are normalized to their average depth. The correlation algorithm is similar to traditional correlation and minimizes the L2-norm, but takes into account missing data (due to flags) and trims errors in the z data to make the algorithm more robust to spike noise. Several experiments were conducted to compare feature regions on a probe scan to those on a target scan; these experiments used the following procedure:

1. Identify anchor points on the target scan.
2. Choose a region around each point as an independent mask. (See Figure 6.19a)
3. Find the anchor points on the probe scan.
4. Choose a larger search region about each point on the probe scan. (See Figure 6.19b)
5. Convolve the masked region in the search region for each corresponding point. (See Figure 6.19c and d)
6. Record the minimum correlation score and the row and column of the points.

The correlation algorithm was tested on the depth channel of the MSU dataset. All scans were normalized using the vertical symmetry Canonical Face Depth Map algorithm (see Section 6.3). The L2-norm (average root mean squared error over the entire correlation window) for each anchor point (shown in Figure 6.20) was used as



(a) Mask Region (b) Search Region (c) Point found on subject (d) Point found on different subject

Figure 6.19: Example of the correlation point algorithm.

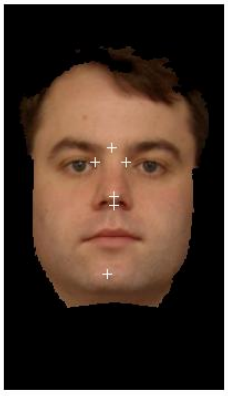


Figure 6.20: Points used in the correlation experiment: nose tip, nose bridge, nose base, inside of right eye, inside of left eye, chin.

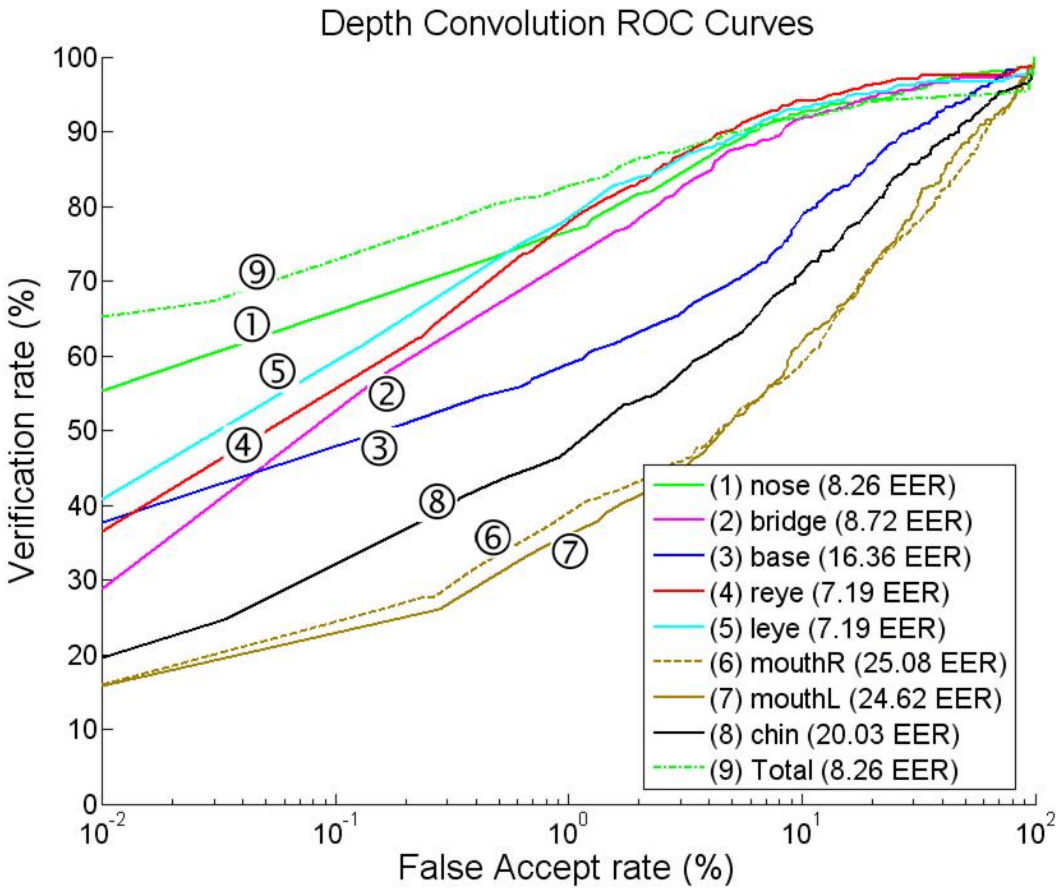


Figure 6.21: Correlation applied to the depth channel of a Canonical Face Depth Map using a large mask window.

a similarity score. Figure 6.21 shows the ROC results for the correlation over every anchor point.

The results were not impressive compared to the SAM results in Chapter 5, suggesting that the mask regions alone were not sufficient to uniquely identify a single person within a database of 330 subjects. Also included in Figure 6.21 is a curve represent the sum total of all of the L2-norm scores. This total performs better than any of the individual points by itself. The correlation algorithm was also applied to the shape index, as shown in Figure 6.22. The bridge of the nose achieved slightly better performance than the depth channel by itself, yet the summation did not improve any of the scores.

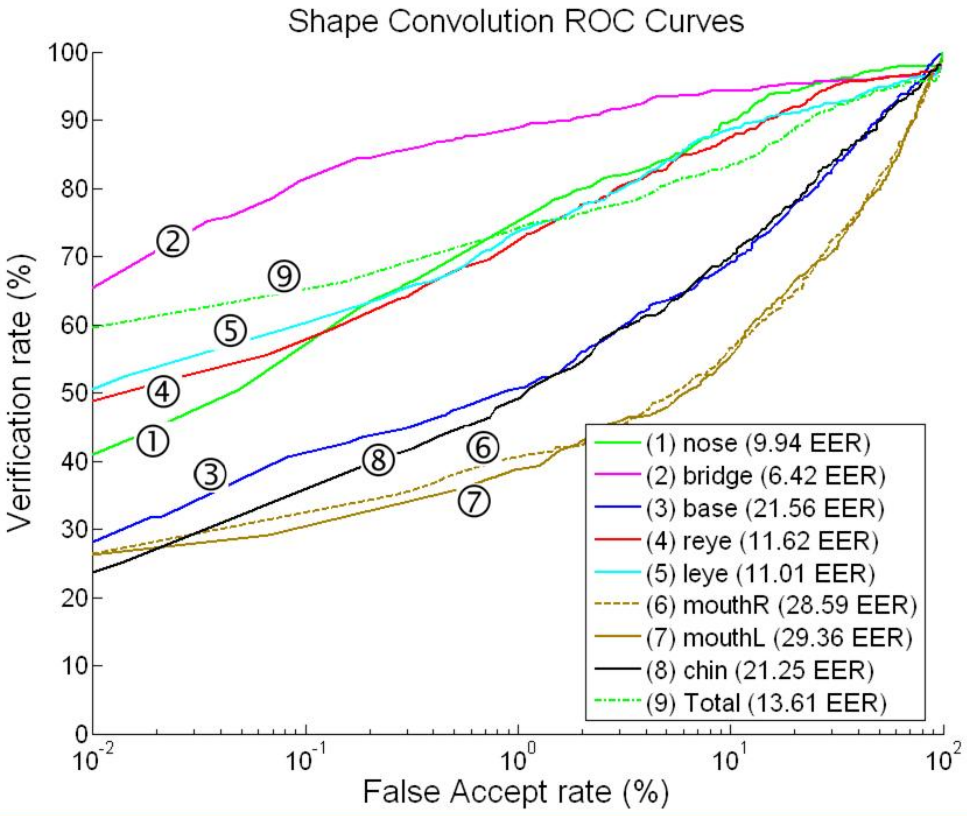


Figure 6.22: Correlation applied to the shape channel of a canonical depth map using a small mask window.

The correlation experiment results suggest that any single anchor point is not sufficient to properly identify a subject. However, combining these feature regions

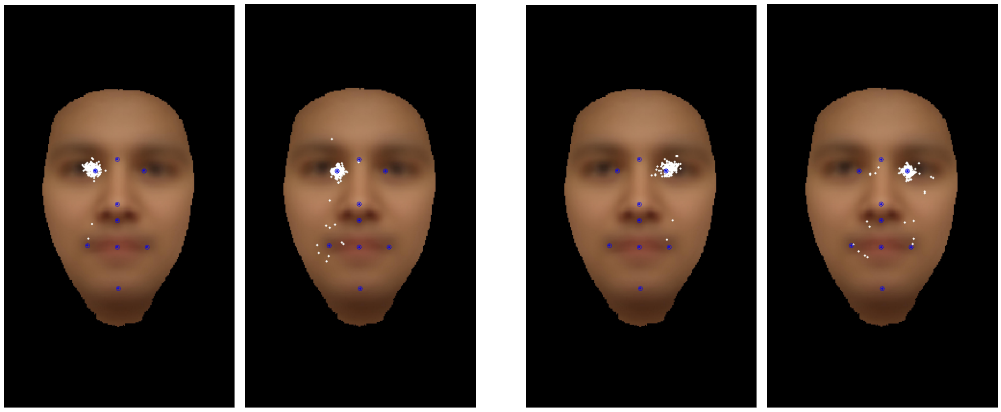
may make verification possible, and correlation may be useful for localizing anchor points.

Another use for the correlation algorithm is for point localization. The correlation algorithm uses a moving window to find the best match between regions around points in the scans. To achieve the best localization results, an operator would need to manually select anchor points on a scan during enrollment. When a new query scan is made, the anchor points on the template are used to help guide the point localization. Figure 6.23 shows the results of point localization on a selection of scans. Notice that there are a larger number of outliers in this algorithm. However, the automatically localized points are better clustered around the manually selected points. Future work will include ways to minimize the number of outliers while maintaining the high quality localization.

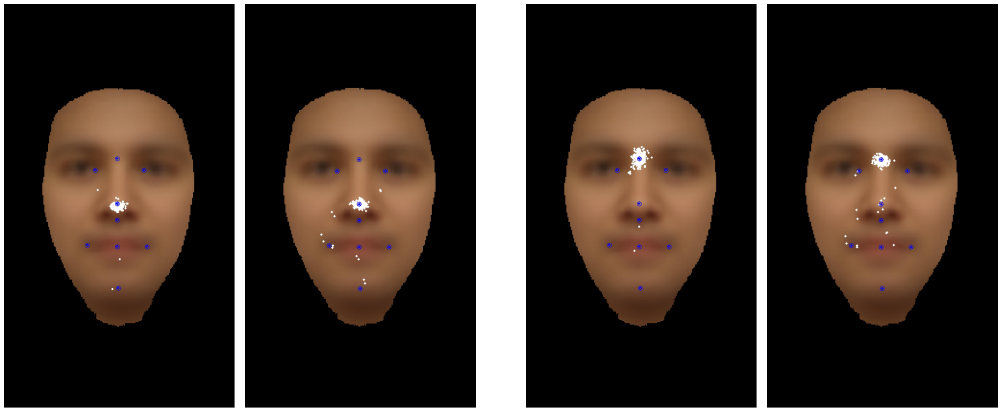
6.6.2 PCA Analysis

In addition to providing a dataset, the Face Recognition Grand Challenge (FRGC) also provides a programming interface called the Biometrics Experimental Environment (BEE). The BEE system gives the FRGC competitors a common interface for running independently-evaluated algorithms. It also provides a version of a PCA matching algorithm to use for baseline comparison.

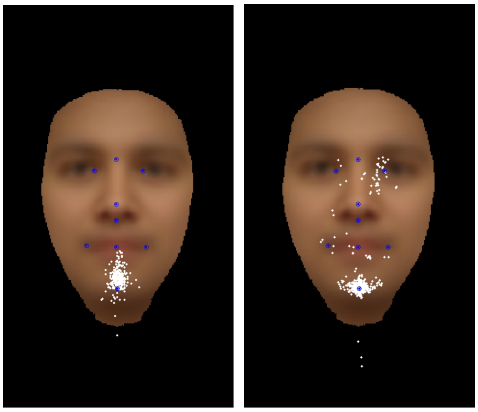
PCA is a well-established 2D color face matching method. It is not state-of-the-art, but has been shown to perform acceptably when there are minor variations in pose and lighting. In addition to using the FRGC PCA algorithm for alignment, it is also possible to use it as a second matching score to be combined with the SAM score. However, this requires some consideration. First, the FRGC PCA algorithm uses *manually selected points*, which is unreasonable for a real-world, fully automatic application. Second, the FRGC PCA algorithm requires some training data. This training data is already available in the FRGC database but needs to be added for



(a) Raw (b) Correlation (c) Raw (d) Correlation



(e) Raw (f) Correlation (g) Raw (h) Correlation



(i) Raw (j) Correlation

Figure 6.23: Point localization using correlation. Average face image with each of the manually selected anchor points (green) and the relative location of the automatically generated points (white) from the MSU 330 image dataset. The left images represents relative the anchor point locations using the raw data and the right images represents the anchor point locations after applying the CFDM.

the MSU database.

The diagram in Figure 6.24 shows the steps involved in using the CFDM alignment algorithm developed in this dissertation as a preprocessor for the FRGC baseline PCA algorithm. The FRGC baseline algorithm was chosen so that a direct comparison with existing results can be shown. For this experiment, the manual FRGC PCA preprocessor was replaced with the Canonical Face Depth Map algorithm. The original FRGC PCA algorithm uses manually selected anchor points for registration and image cropping. However, the Canonical Face Depth Map preprocessor is *fully automatic*. Results are shown in Figure 6.25 for the MSU database of 330 scans. As expected, the manually selected anchor points are more accurate than the automatically selected anchor points. However, the automatically selected system seems to be comparable to the manually selected system.

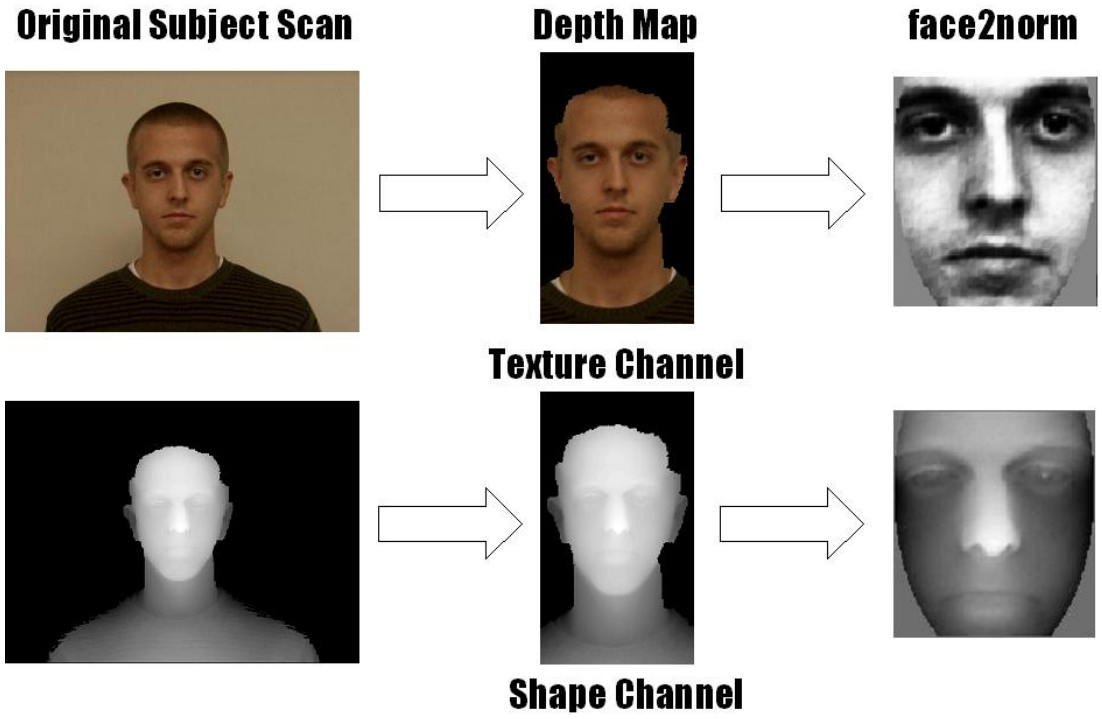


Figure 6.24: Preprocessing the scans for use with the FRGC PCA algorithm is a two step process. First the scan is processed into its CFDM format and then the FRGC normalization algorithm called "face2norm" masks out unwanted regions and normalizes the depth and color.

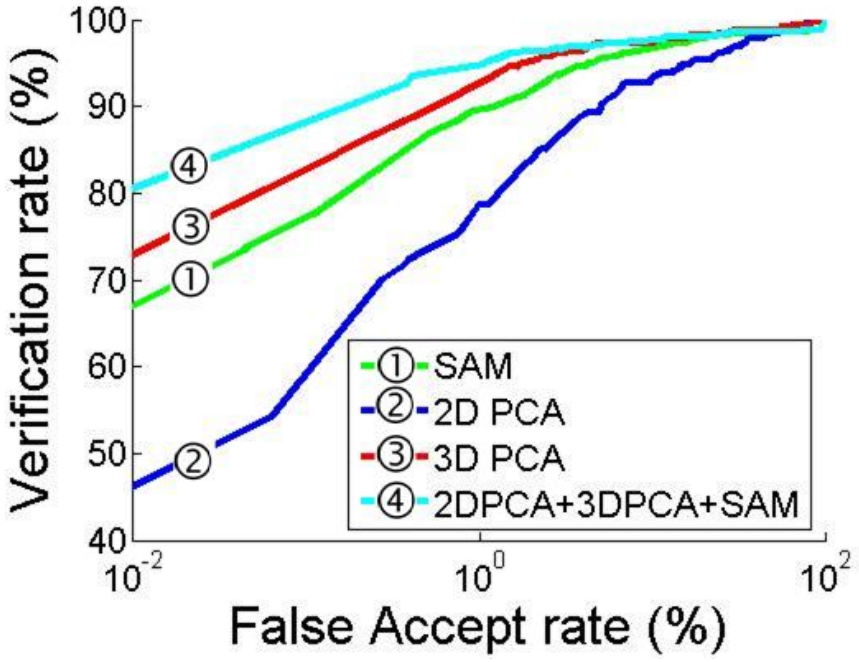


Figure 6.25: Comparing automatic PCA with manual PCA.

Figure 6.25 shows that the normalized PCA algorithm performs even better than the SAM, which is expected because the PCA algorithm is trained on a dataset. It is rather naive to add the PCA and the SAM scores together since they are not in the same units or scale. However, this naive addition produces better results than either score alone, suggesting that a more intelligent, multi-modal combination of scores may produce even better results.

Chapter 7

Research Platform

The previous chapters described different algorithms and methods for manipulating and analyzing 3D faces. This chapter describes the 3D research platform developed for and used to run the experiments described in this dissertation. This platform is considered to be an important contribution of the dissertation work. Section 7.1 describes the history of the programming interface. Section 7.2 discusses the operation of the demo program and its integration with the Vivid 910 scanner. Finally, Section 7.3 details the performance of the different components of the system and compares this performance on databases of different sizes. Appendix A covers the design and use of the research platform in more detail.

7.1 Program History

Research programming must be more fluid than system programming, since it is not possible to design every aspect of the research platform before some of the experiments are conducted and the results analyzed. In the worst case, it may be necessary to completely rewrite portions of the code to account for some aspect of the software that was not initially anticipated. For these reasons, the programs developed for this dissertation have evolved over time to enable the examination of new problems that

may not have been considered when the original code was developed. In developing the research platform, several design goals were pursued:

1. Developing software that was easy to use and enabled quick testing of experimental hypotheses without needing to redevelop redundant support software.

2. Creating a real time demo program capable of demonstrating the viability of 3D face verification.

3. Building a testing platform that was compatible with the NIST Biometric Experimental Environment (BEE).

4. Designing a testing platform that could be compiled and run on a High Performance Computing System.

These design goals occasionally conflicted and care was taken to design a system that could balance these different requirements.

Initially, Matlab was used exclusively to develop a system prototype. There were three main reasons for using Matlab: fast algorithm development and testing; fast data analysis and visualization; and existing software for converting from the FRGC dataset into a Matlab format. The Matlab prototype system was able to demonstrate proof of concept; however, there were some limitations on how fast a program could execute in Matlab and the prototype system took over two minutes to complete a face comparison.

In order to achieve the goal of a real time demo, the Matlab prototype was ported to C++. The C++ version of the research platform is much faster than Matlab and is able to control the VIVID 910 scanner directly. Because C++ is a more difficult environment to develop in, the first goal for the C++ program was to enable the loading and saving of the Matlab data format using the Matlab API. This allowed the C++ system to be built in stages, compared to the original Matlab prototype

code, and allowed the design to take advantage of the benefits of both programming environments.

The real time C++ system allowed for fast data collection and quick visualization of the core algorithms. However, the demo system used the Microsoft Foundation Classes (MFC) for the user interface, which added programming overhead. The MFC classes are not cross platform compatible and the GUI interface made it difficult to run large batch comparisons (although the GUI could be run in batch mode) or to add new experiments.

A stripped-down version of the program was developed to run completely from the command line, which simplified the development and improved the execution time of the experiments. The command line version also allowed the experimenter to write a new, independent program for each type of experiment instead of maintaining one large, bulky program that has to do everything. However, because Visual Studio was used to develop the initial C++ MFC project, it was difficult to add additional command line programs to the project. Consequently, the C++ program was ported into a CMake project that allowed for the simple addition of new programs and enabled the new command line programs to be compiled on different systems (such as Cygwin, Linux, Solaris, etc). Another advantage of using CMake was that the entire project could be represented as text files, instead of the numerous binary files required by Visual Studio. A text-based project also allows for the use of simple project versioning tools such as CVS, which enables large numbers of programmers to work on the same project in completely different areas.

With the ability to write many command line executables, the 3D research platform expanded to include a series of 3D face command line tools. These tools enable quick format conversions, feature detection and data analysis. The command line format also allowed for the development of tools to work directly with the Biometrics Experiment Environment (BEE). The BEE is a software interface used in the

Face Recognition Grand Challenge (FRGC) to normalize the input and output data formats. The BEE system is designed to standardize the method for preprocessing and analyzing biometric data. The BEE system is provided with the FRGC database and not only contains the necessary interface, but also provides a baseline matching algorithm for comparison. Working with the FRGC dataset is not a trivial task, as it takes some time to gain proficiency. The code is exclusively written for Linux, and in order to install the code administrator privileges are required.

Even though the faster C++ system (with command line option) can compare two faces in less than a second, analyzing the entire FRGC dataset requires 16 million comparisons, which would take around 185 days to complete on a standard Pentium four 3Ghz PC; more complex trials that are designed to explore different research possibilities take even longer. To improve experimental speed, programs were developed to enable the FRGC algorithm to run on the MSU High Performance Computing (HPC) system. Because each of the 16 million comparisons are independent of each other, the FRGC experiments are considered "embarrassingly parallel," meaning that the system can take full advantage of the large number of processors available on HPC (128 nodes with 2 duel core processors on each node for a total of 512 cores).

The current 3D research platform runs under Windows, Linux, Solaris, HPC and can integrate directly with Matlab. The research platform is also able to load and save to many different file formats, allowing for maximum research and data options.

7.2 Demo Interface

The demo program is a specific aspect of the 3D Research Platform that runs on Windows. The demo program is designed to gather data using the VIVID 910 and demonstrate the viability of using 3D face matching in a practical system. The three main features of the demo are:

1. Data are gathered using the VIVID 910 in fast mode.

2. The practicality of the baseline matching algorithm is demonstrated by using the SAM matching score and symmetry reject option in real-time.

3. Different visualization methods are provided to demonstrate how the data is processed and what the algorithm is doing.

A subject stands in front of the camera (as shown in Figure 7.1) and must hold still for about 2 seconds to capture a scan. Figure 7.2 is a flow chart representing the operation of the prototype system designed based on the demo system. Many additional features have been incorporated into the demo to increase functionality, such as keyboard shortcuts, batch process capabilities and the ability to turn different aspects of the matching algorithm "on" or "off".

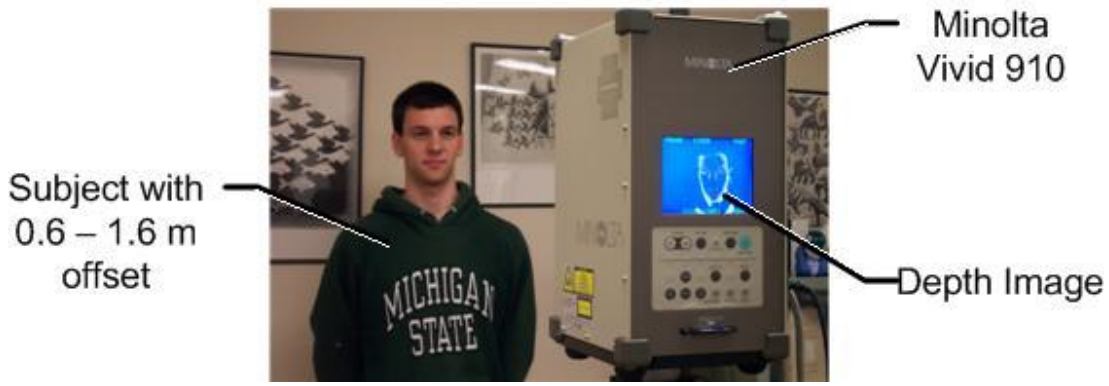


Figure 7.1: Standard operation of the VIVID demo system.

The remainder of this section describes the three main features of the demo in detail. Section 7.2.1 describes the different components of the VIVID 910 interface, Section 7.2.2 outlines the use of the main matching algorithm and Section 7.2.3 discusses the different methods used to visualize the data.

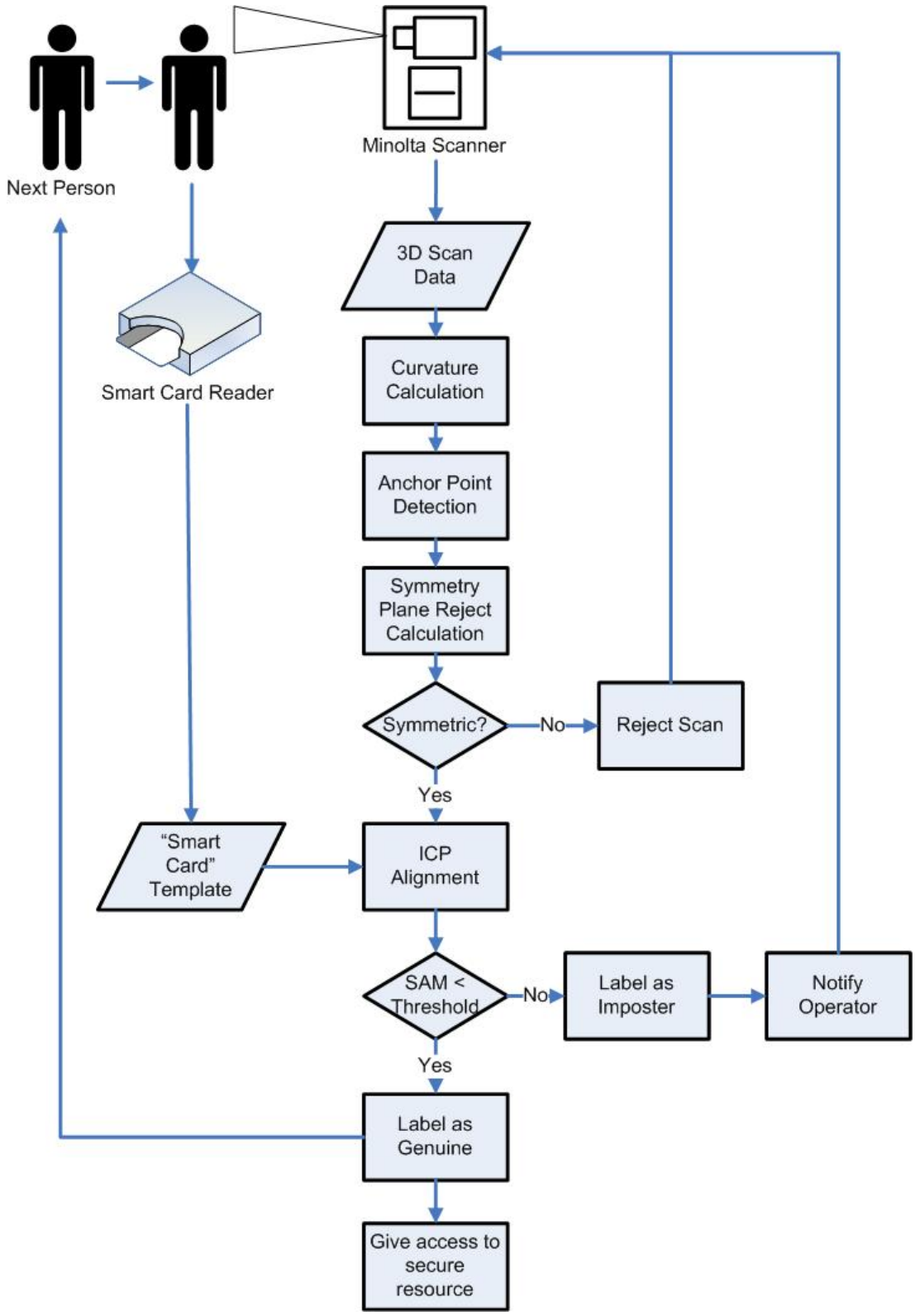


Figure 7.2: Flow chart representing the operation of the prototype 3D face verification system.

7.2.1 VIVID 910 Operation

The VIVID 910 uses a SCSI interface to communicate with the host computer. The interface is programmed with a thread that continuously queries the scanner and updates the viewing window shown in Figure 7.3. The viewing window uses a double buffer system to maintain a smooth display.

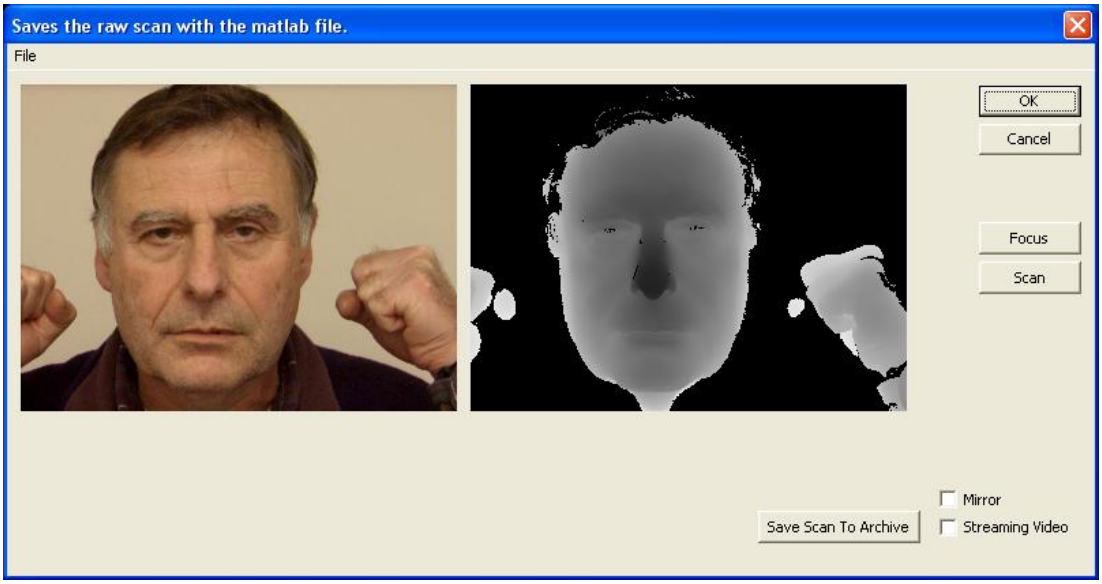


Figure 7.3: VIVID 910 interface

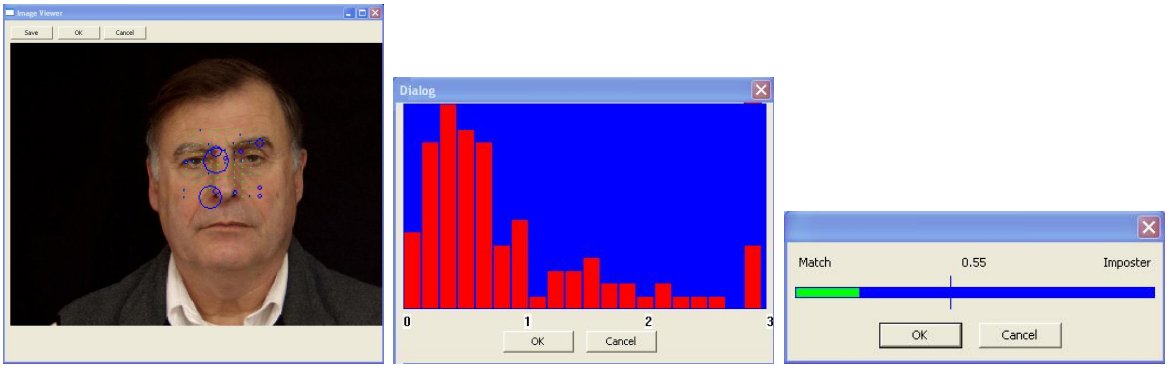
When the demo system is first started, the focus button must be pressed to initialize the scanner. As long as the subjects stay within the same distance window to the scanner, focusing should not need to be repeated. It is a simple matter to put a line on the ground indicating the location where a subject should stand. Once the scanner is properly focused, the scan itself will take about a second.

When a scan is complete, the right window (in Figure 7.3) displays the depth map, while the left window displays a still image of the scan. The streaming video option can be selected if a moving image is desired. The scanner interface includes options to load and save raw VIVID data (cdm files) from the scanner menu. The user interface also includes a mirror scan option, which flips the output window from left to right and makes it easier for subjects to center themselves when viewing the

output window. Once a scan is made and the "ok" button is pressed, the demo system calculates the surface curvatures and anchor points and then measures the symmetry of the scan. If the face is not found to be sufficiently symmetric, the demo system reports a reject and the subject can be rescanned.

7.2.2 Matching Algorithm

The matching algorithm takes two scans (a model and a query) and aligns them using the 3D Face Alignment algorithm described in Chapter 5. Figure 7.4 shows the three windows that appear when executing the matching algorithm. Figure 7.4a shows the model scan with the 100 control points selected from the query scan. The red control points represent the trimmed points, and the blue circles around each control point represents, the current error for each point (1 pixel in radius is approximately one millimeter). Another window (see Figure 7.4b) shows a histogram of these errors (all distances above $3mm$ are put into the $3mm$ bin). Figure 7.4c is a bar graph representing the current SAM score. The vertical line indicates the threshold value. A matching score to the right of the vertical line indicates an impostor, while a matching score to the left of the vertical line indicates a match.

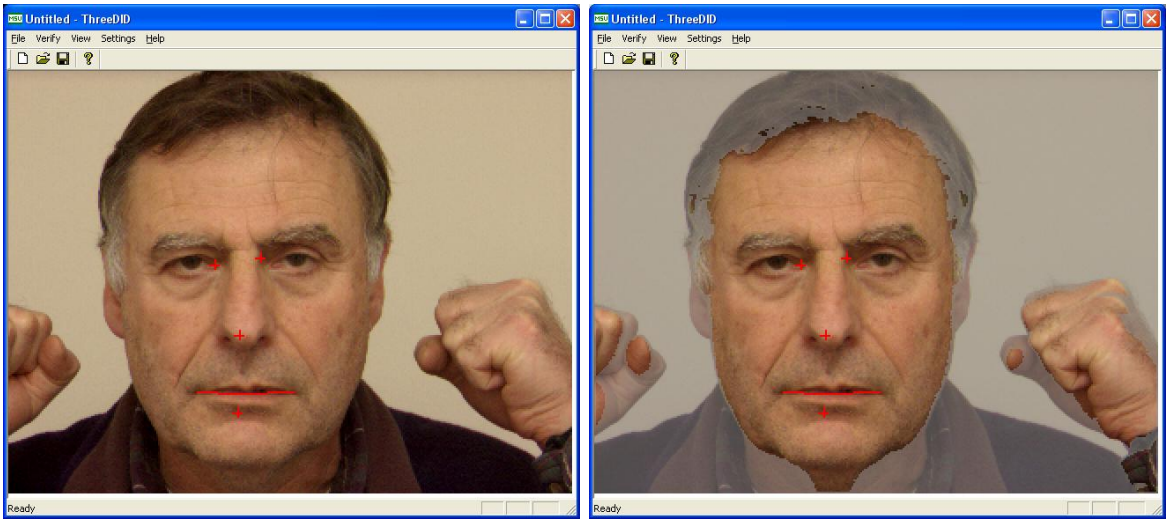


(a) Model Scan with Control Points (b) Point Error Histogram (c) Current SAM

Figure 7.4: Matching algorithm demo visualization.

The matching algorithm compares the model to the query and the query to the model and reports the lowest of the two scores. A final display will appear either accepting or rejecting the subject.

7.2.3 Data Visualization

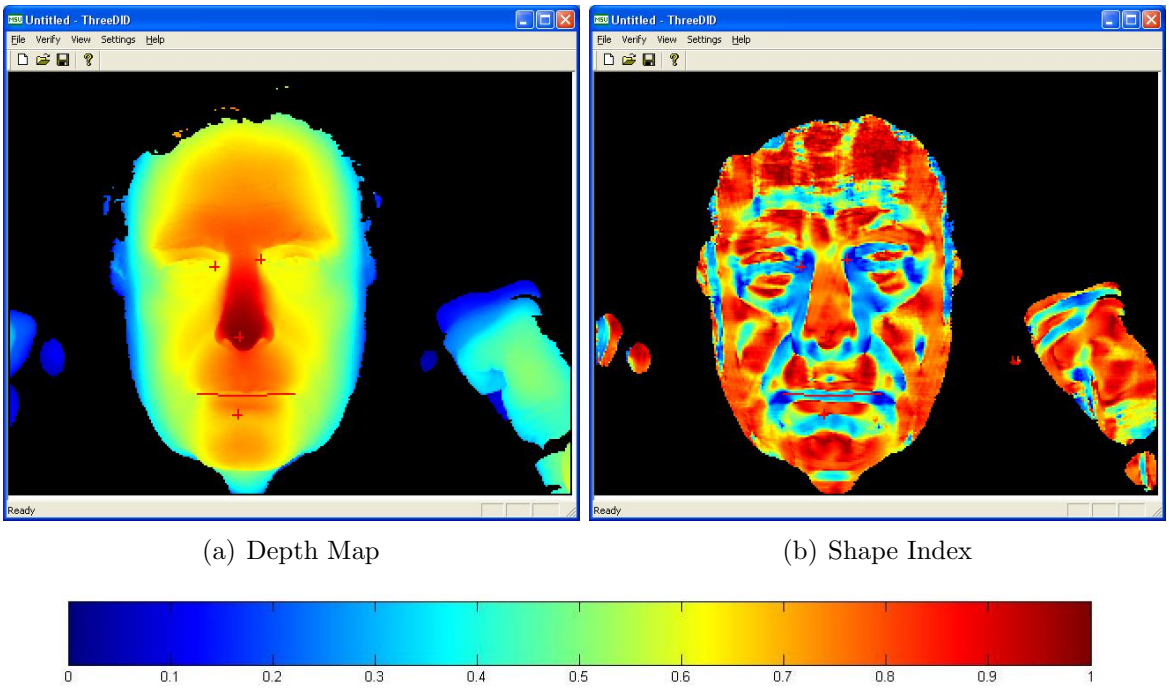


(a) Main Window (b) Main Window With Flag values showing.

Figure 7.5: Main image window showing anchor point location with and without flag values.

Once a scan has been taken or loaded from a file, it is displayed on the main demo window (see Figure 7.5a). The main window has a tool bar for executing the main system commands and setup parameters. By default the image is displayed with the automatically detected anchor points, which can be toggled on and off. The flag values can also be toggled on to indicate where the scanner did not pick up any valid data. In addition to the color image, the depth map (see Figure 7.6a), shape index (see Figure 7.6b) and surface normals (see Figure 7.7) can also be displayed as a 2D image. The depth map and shape index are displayed on a color space represented by the scale shown in Figure 7.6c. The surface normals are shown in red, green and blue. The red component of the surface normal represents the absolute value of the

normal in the x direction. the green component of the surface normal represents the absolute value of the normal in the y direction, and the blue component of the surface normal represents the value of the normal in the z direction.

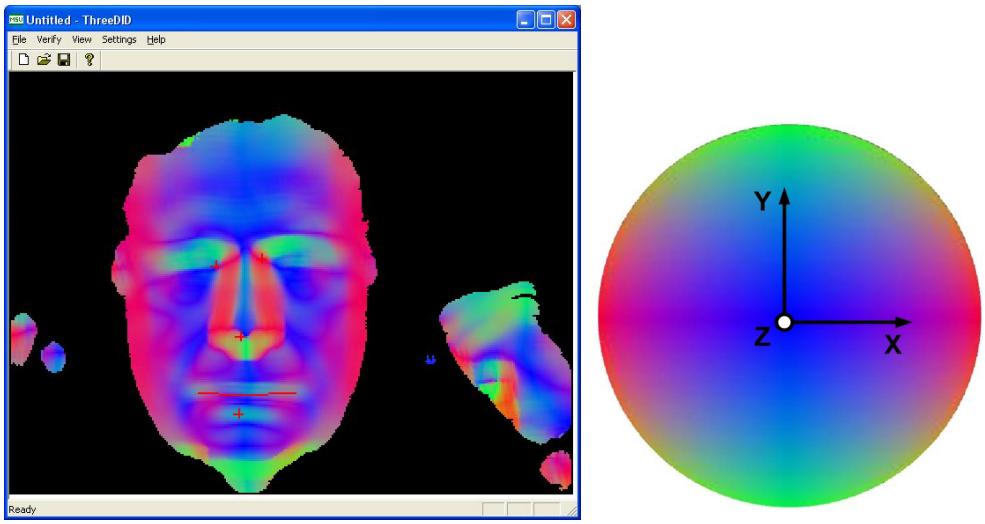


(a) Depth Map (b) Shape Index

(c) Color Space (blue to red)

Figure 7.6: Matching algorithm demo visualization.

Any of the main window visualization modes (shown in Figures 7.5, 7.6 and 7.7) can be exported to an image file. The system can also export the data directly into a VRML file, which will be displayed immediately using an external VRML viewer. This gives the system the ability to output models that can be rotated, in addition to the 2D images. The VRML code is programmed so that an image file is used to map the texture onto the shape. If the image file is overwritten with any of the above visualization modes, this mode will be used by the VRML viewer (see Figure 7.8).



(a) Normal Image (b) Normals on a Unit Sphere

Figure 7.7: Surface normals with the x,y,z directions represented as red, green, and blue.

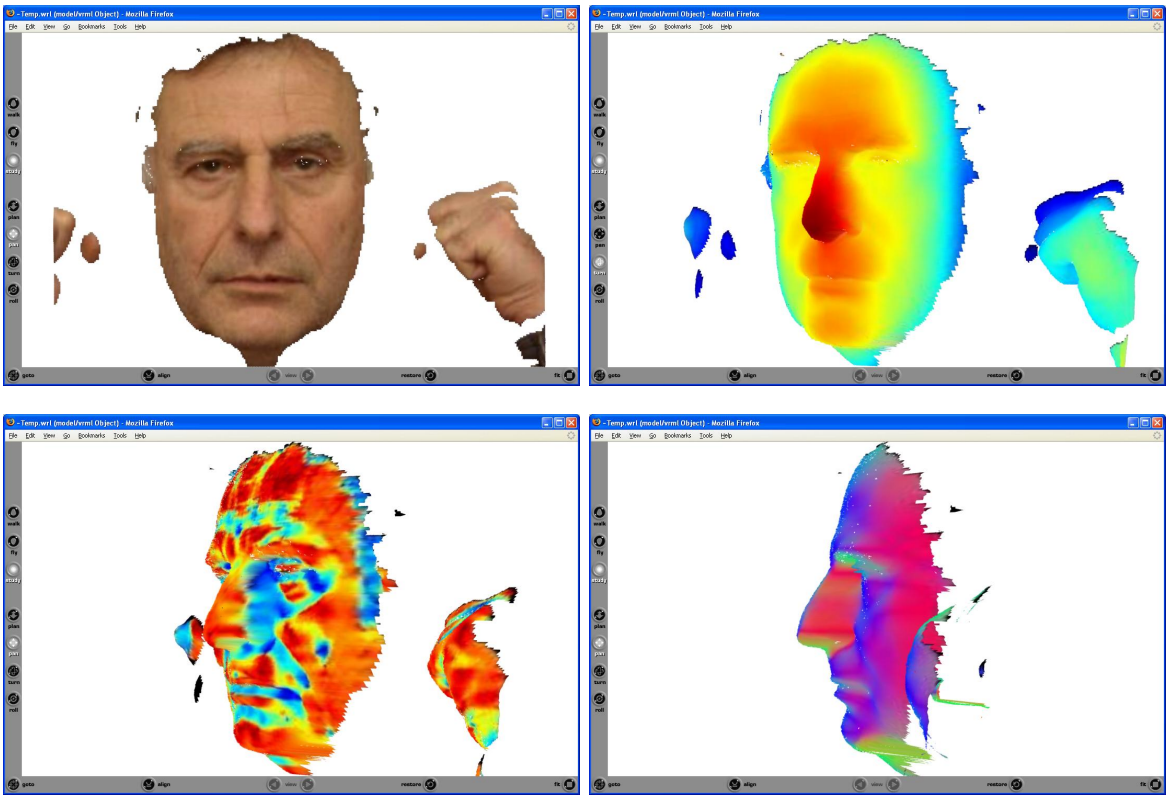


Figure 7.8: 3D VRML viewer with color, depth, shape and normal data representations.

7.3 Performance

This section summarizes the run times for each of the algorithms presented in this chapter. (Previous chapters have presented similar results for individual components of the algorithm.) Table 7.1 offers the running times for the main test program running from Cygwin on a Windows 3.1 GHz dual processor computer with 1Gig of RAM.

Table 7.1: Algorithm processing times in seconds.

	Cols	Rows	Read Model	Curve	Anchor Points	Symmetry	Write Model	Total
MSU MAT	320	240	0.112 ±0.017	2.610 ±0.442	0.052 ±0.009	0.870 ±0.288	0.144 ±0.025	3.790 ±0.590
ND 3BX	640	480	1.207 ±0.074	13.000 ±3.816	0.378 ±0.037	3.582 ±1.672	0.725 ±0.056	18.490 ±3.960

As can be seem in Table 7.1, the entire algorithm takes around 4 seconds to execute using the Matlab file format. The majority of this time is spent calculating the surface curvature. Table 7.2 shows the cycle time for the current demo matching algorithm. These times are all conservative and suggest that this algorithm can be used to create a viable practical system.

Table 7.2: Complete process time for the current demo of a practical verification system system.

	Demo Cycle Time (seconds)
Load Model	0.2
Scan	2.8
CalcCurve	2.4
Anchor Point	0.1
Symmetry Reject	1.1
Alignment	1.3
Swap Alignment	1.3

Chapter 8

Conclusions, Discussion and Future Work

This dissertation studies the use of 3D scanners to improve face verification systems. Traditional 2D face recognition systems are not tolerant to changes in pose and lighting. However, 3D data can be used to reduce the effects of lighting and pose variations for face identification. Table 8.1 summarizes the chapters in this dissertation and the major ideas, objectives, and contributions of each chapter.

The data used in this dissertation was gathered with two different Minolta Vivid 910 scanners over a four year period. This scanner has been shown to have high accuracy ($< 1mm$ in depth), fast capture (under 2 seconds), and it produces data that is consistent across this model of scanners [22]. The preliminary experiments started with as few as 10 subjects and 63 scans, and has grown to include over 625 subjects with over 5500 scans. The combined testing for all of the experiments was run on over 35 computers, including two High Performance Computers (one with 32 processors and the other with 512 core processors). These resources were used to conduct extensive experiments, which totaled over 6 years of computational time.

Table 8.1: Dissertation overview.

Chapter	**Major Content and Contributions**
2. Background	• 3D coordinate system notation • 3D sensor overview • 3D data file types • Data collection and statistics
3. Solution Approach	• Surface fitting • Curvature calculations • Scanner resolution, scale and orthogonal depth maps • Image cleaning
4. 3D Anchor Point Detection	• Fully automatic frontal anchor point detection • Fully automatic arbitrary pose anchor point detection
5. 3D Face Alignment	• Algorithm robustness tests • System error analysis
6. Toward a Canonical Face Format	• Face-based coordinate systems • Canonical Face Depth Map (CFDM) • Preprocessing using CFDM
7. Research Platform	• Real time demo • System performance study
8 . Conclusions	• Anchor point detection and evaluation • Practical 3D face verification overview • Feature space evaluation with fusion results
7. Appendix	• File formats • Cross platform capabilities

Chapter 5 presents an algorithm for 3D surface alignment, used to align the 3D surfaces of two faces. The alignment algorithm is a two step process. In the first step, key anchor points (found using the algorithms presented in Chapter 4) are automatically identified on the surface of the face and used to coarsely align two faces scans. In the second step, an iterative hill climbing algorithm called ICP (Iterative Closest Point) is used to finely align the scans. The quality of this two-step alignment process is studied in detail using a Surface Alignment Measurement (SAM) (see Section 5.3). The SAM is the root mean squared error over all the control points used in the ICP algorithm, after trimming to account for noise in the data. Large SAM values could mean improper alignment between two scans, or they could mean that the surfaces are shaped differently.

Because the SAM value is a measure of the quality of the surface alignment, it is also used as a baseline face matching score. Section 5.4 demonstrates that using the SAM as a matching score can achieve a 1% equal error rate (EER) on a database of frontal scans restricted to neutral expressions. The performance decreases for faces with different poses and expressions. However, these experimental results still suggest that using the SAM score alone can provide a practical face verification solution in a market where subjects are cooperative (such as a security checkpoint).

The surface alignment algorithm requires a number of engineered input parameters, such as the number of control points, the control point selection regions, and the percentage of trimmed points. The experiments in Section 5.2 varied these parameters and compared the resulting variations in the speed of the algorithm and the EER. These experiments show that the surface alignment algorithm is robust to changes in algorithm parameters, and the results determine the optimal operating parameters for the alignment system.

One strength of the fast, fully automatic surface alignment algorithm developed in this dissertation is that it can be used as a general preprocessor for other 3D based

face recognition algorithms. For example, the surface alignment algorithm is used in Section 6.2.3 to robustly identify the plane of symmetry on a face by aligning a face scan with a mirror projection of itself. This plane of symmetry calculation can be used as a reject option to increase the performance of the face verification algorithm, decreasing the EER on the FRGC 1.0 dataset to 1.2% with a 1.5% rejection rate. The plane of symmetry can also be used to help define a face-based coordinate system.

The Canonical Face Depth Map (CFDM) is defined in Chapter 6 as a face-based coordinate system projected onto an orthogonal depth map. The CFDM normalizes and re-samples the 3D data to align all the faces in a database which are taken at different poses and scales. This alignment does not require a generalized face model, and the CFDM can be used to automatically preprocess a database of 3D images. Experiments with a PCA-based algorithm in Section 6.6.2 show that the CFDM provides robust, automatic preprocessing, and performance results are close to those of a database that is normalized by hand.

The high quality alignment and fixed resolution size of the CFDM allows for exploration of feature spaces and matching scores not normally available in arbitrary 3D scans. For example, the CFDM makes it possible to apply correlation to the 3D scans, as shown in Section 6.6.1. The correlation algorithm can be used to localize anchor points, and also provides additional information for the matching process. Experimental results show that by combining simple correlations of the bridge of the nose with the SAM score, better matching performance is achieved over either measurement alone.

Table 8.2 summarizes the major contributions of this dissertation. These findings have been disseminated though a number of publications, as described in Table 8.3.

Section 8.1 combines results from different parts of the dissertation to synthesize and discuss some findings from this dissertation, while Section 8.2 discusses directions for future work.

Table 8.2: Major contributions.

3D face alignment algorithm	Uses the automatic anchor point detector developed in Chapter 5 as a first step to an automatic surface alignment algorithm.
Practical 3D face verification system	Using the SAM value as a matching score, Chapter 5 demonstrated that the face verification system can achieve over 99% accuracy or less than 1% equal error rate on a database of cooperative subjects.
Canonical Face Design	Chapter 6 defines the robust Canonical Face Depth Map and an algorithm to robustly take a 3D face scan and convert it into this standard format.

Table 8.3: Publications based on research in this dissertation.

1. George Stockman, Jayson Payne, Jermil Sadler and Dirk Colbry. ***Error Analysis of Sensor Input Variations in a 3D Face Surface Matching System.*** Sensor Review Journal, Volume 26, No. 2, pages 116–121, 2006.

2. Xiaoguang Lu, Anil K. Jain and Dirk Colbry. ***Matching 2.5D Face Scans to 3D Models.*** IEEE Transactions on PAMI, 28(1):31-43, 2006.

3. Dirk Colbry George Stockman and Anil Jain. ***Detection of Anchor Points for 3D Face Verification.*** In IEEE Workshop on Advanced 3D Imaging for Safety and Security A3DISS, San Diego California, 2005.

4. Xiaoguang Lu, Dirk Colbry and Anil K. Jain. ***Three-Dimensional Model Based Face Recognition.*** In 17th International Conference on Pattern Recognition, pages 362-365, Cambridge, United Kingdom, 2004.

5. Xiaoguang Lu, Dirk Colbry and Anil K. Jain. ***Matching 2.5D Scans for Face Recognition.*** In International Conference on Biometric Authentication, pages 30-36, Hong Kong, 2004.

8.1 Discussion and Synthesis

This dissertation presents methods for aligning faces for face verification. The methods presented in this dissertation could also be used for face recognition tasks where the system searches for a single subject in a database of scans. However, because the pair-wise match between two 3D scans can take as long 4 seconds, searching for a match in a large database would take a long time. Also, a design constraint of many of the matching algorithms presented in this dissertation is that the subjects are being cooperative. Assuming cooperative subjects is reasonable in a verification application, because subjects need to cooperate to gain access to a resource. However, this assumption is not as reasonable in a recognition application, where subjects may not know they are being monitored (e.g., a surveillance application). The ideas and methods presented in this dissertation encompass several major themes, specifically:

- Anchor point detection and evaluation.
- Requirements for a practical 3D face verification system.
- Face verification matching score performance and fusion methods.

These themes span the work presented in this dissertation and the remainder of this section discusses them in detail.

8.1.1 Anchor Point Detection and Evaluation

Anchor points are used throughout this dissertation, and the quality of the anchor point detection is key to the success of many algorithms. In Chapter 5, anchor points are a necessary first step for coarse alignment of the faces. Section 5.2.1 uses anchor points to help establish the location of the control points used by the ICP algorithm. Proper anchor point detection helps to ensure that the control point regions selected avoid areas of the face that change most with expression. Reliable anchor point detection can be used to help model expression [98] or even as a feature for matching

[124]. However, each of these applications require anchor points of various accuracy and repeatability, and it can be difficult to evaluate these qualities in deleted anchor points. Some reasons for the difficulty in evaluating anchor point detection include:

- It is difficult to define a single, salient point on the face.
- The definition of a point (such as the tip of the nose) may be flexible, and using humans or a computer to label these points will produce different results.

The remainder of this section reviews the different methods used in this dissertation to calculate anchor points, and the methods used to evaluate the anchor points after detection.

Detection

In Chapter 4, two algorithms were presented to detect anchor points in 3D face data. The first approach is used for frontal poses and the second algorithm is used for arbitrary poses. The anchor points produced by these detection algorithms were shown to be sufficient to coarsely align two scans such that the ICP algorithm will converge properly. However, there is still some variance in these anchor points, especially around the corners of the mouth. In Chapter 6, the face is put into a canonical format before the anchor point detection algorithm is run, resulting in some minor improvements in point variance.

The anchor point algorithms presented in this dissertation are either fully automatic or they use prior knowledge of the anchor points provided by the template. The fully automatic system is desirable because it does not require that a subject be recorded prior to use. However, if a template is available then it is not unreasonable to use manually selected anchor points on the model, even though manual selection will make the enrollment procedure more time consuming. The manual anchor points can provide better results at the cost of algorithm flexibility and ease of use.

Evalution

Table 8.4 lists methods that can be used to evaluate the quality of anchor point detection. One method is to manually examine the anchor points and evaluate their quality, as was done in Section 4.4.3. The human evaluator has limited choice (good or bad) and makes a Boolean decision. Although this approach is straightforward, manual evaluation is difficult and time consuming, and since only Boolean results are available the results of this approach are not easily quantified and compared.

A second option is to evaluate the quality of the ICP alignment and use this as an indirect measure of the anchor point quality, as was done in Section 4.3. If the anchor points are good enough to converge in ICP, then they are considered to be of acceptable quality. An advantage of the ICP approach is that it constrains the evaluation to a much simpler problem. In addition, it may be possible to develop a threshold for the SAM score, which would allow for automatic evaluation of the anchor points.

A third option is to compare the automatically selected points with manually selected points, as in Sections 4.3.1 and 6.5. However, this approach negatively biases the results because the automatic system may be highly repeatable within a subject, even if it is not finding the same anchor point as the manual (ground truth) point, or because the manually selected points also have errors. At best, a comparison to manually selected ground truth can give an estimate of the variance of the automatically selected point data.

A fully automatic method (i.e., no subjective input is required) to analyze the quality of the anchor points is to compare anchor points from different scans to one another using the ICP algorithm. Each scan is aligned to other scans of the same subject, and the difference between the automatically selected anchor points gives a measure of the variance within the dataset. The problem with this approach is that expressions and surface changes are not taken into consideration.

Table 8.4: Methods for evaluating anchor points.

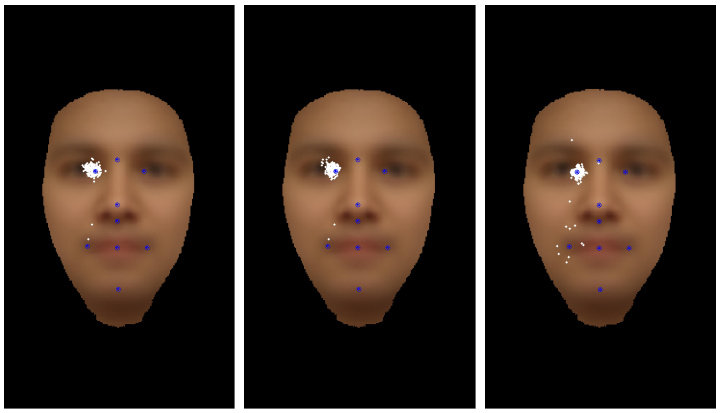
Method	Description	Pros	Cons
Manual Evaluation	Visually inspect the anchor points and give a Boolean result as to their accuracy.	Easy to implement.	Method is not repeatable and is subject to the evaluators' ability to be consistent with their evaluations.
ICP Convergence	Manually see if the ICP algorithm converges given the selected anchor points.	Gives a foundation and criteria that should help the consistency of manual evaluation.	The anchor point accuracy is only useful if ICP is used because it can make up for errors in anchor points locations.
Manually Selected Ground Truth	Manually select points on the face and assume them to be the ground truth.	Distance measurements with statistical comparisons can be made.	Manual points may not be repeatable and the definition of an anchor point may be different between the evaluator and the algorithm.
Cross ICP Comparison	Automatically select anchor points on two different models, align the results and then compare the anchor point locations. Also addresses the issue of algorithm repeatability.	Can be measured easily and avoids human evaluation.	Relies on the ICP algorithm for evaluation and assumes that the two faces are from the same subject and without expression.

Localization

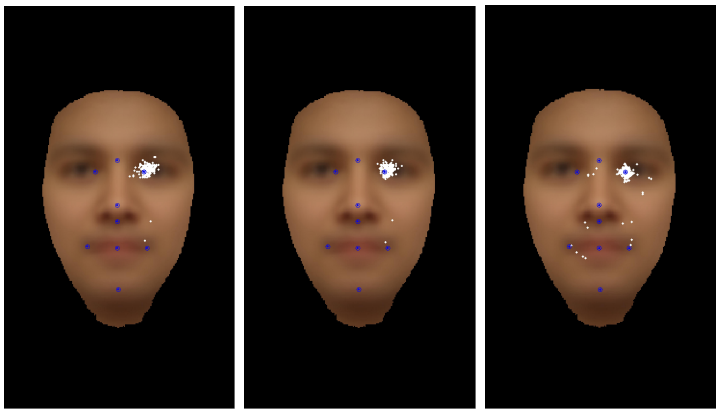
Even though the accuracy of the anchor points are difficult to evaluate, this dissertation has shown that the anchor point detection system presented in Chapter 4 does a good enough job for the ICP alignment system. However, if more accurate anchor points are required, a point localization method could be considered.

Section 6.5 demonstrated that by preprocessing the scans into the CFDM format better anchor point localization could be archived by just re-applying the anchor point detection algorithm presented in Section 4.3. Another option for point localization is the transfer localization method, which assumes that the manually selected anchor points on the model scan are correct and then transfers these points to the nearest neighbor on the query scan after ICP alignment. There are two problems with this point transfer localization method. The first problem is that point transfer localization does not take into account the motion of points within the face due to expression change, because the points are transfered directly. The second problem is that the transfer biases the location of the anchor points and makes it difficult to use them as a biometric measure.

A better option is to use the manually selected anchor points as a guide for point localization, as was done in Section 6.6.1 with point correlation. In this case, a window region is selected around the manually selected anchor point on the template and the same window is searched for on the query scan. This method accounts for minor changes in expression and can do a better job in localizing the points around the desired manual point. However, this localization method still has a larger number of outliers which still need to be accounted for. Figures 8.1 and 8.2 show examples of all three localization methods.

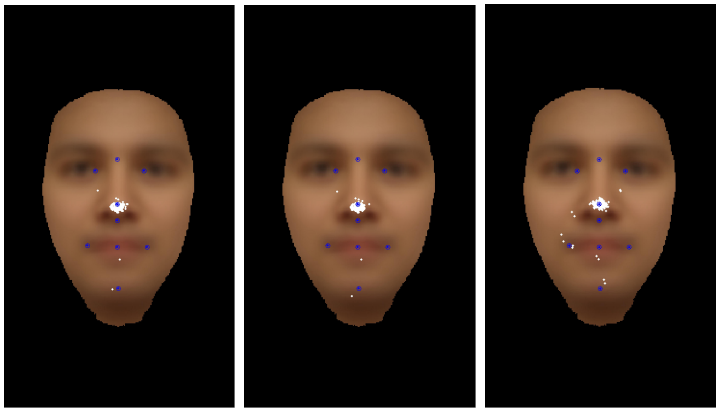


(a) Raw (b) CFDM (c) Correlation

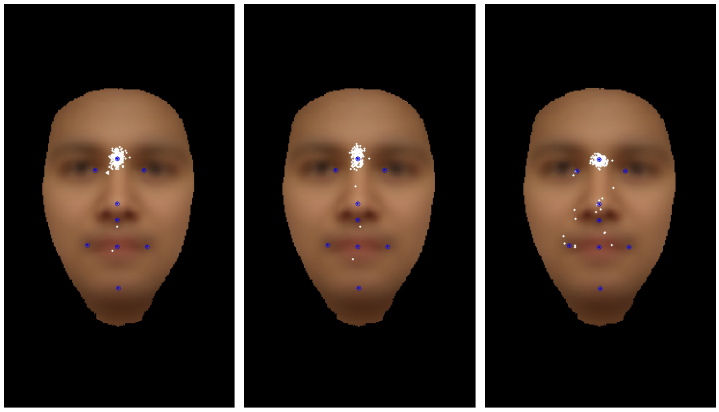


(d) Raw (e) CFDM (f) Correlation

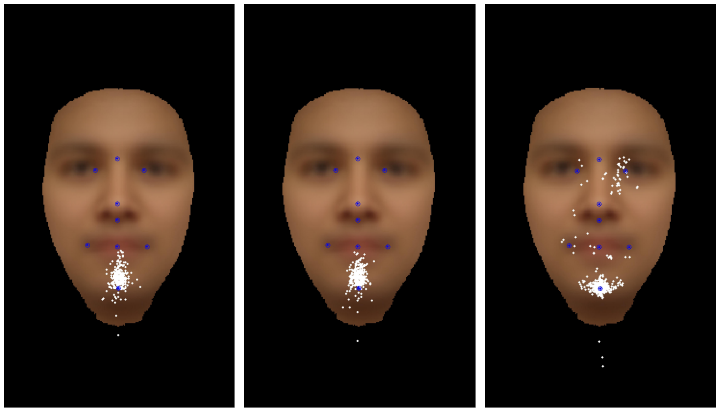
Figure 8.1: Eye corner localization. Average face image with each of the manually selected anchor points (green) and the relative location of the automatically generated points (white) from the MSU 330 image dataset. The left images represent relative anchor point locations using the raw data, the middle images represent the anchor point locations after applying the CFDM, and the right images represent the location after correlation localization.



(a) Raw (b) CFDM (c) Correlation



(d) Raw (e) CFDM (f) Correlation



(g) Raw (h) CFDM (i) Correlation

Figure 8.2: Point localization using correlation. Average face image with each of the manually selected anchor points (green) and the relative location of the automatically generated points (white) from the MSU 330 image dataset. The left images represent the relative anchor point locations using the raw data and the middle images represent the anchor point locations after applying the CFDM and the right image represent the anchor point locations after applying correlation.

8.1.2 Practical 3D Verification System

Many of the methods presented in this dissertation were developed with a practical 3D face verification system in mind. An example application for such a system might be a security checkpoint, where users would have smart cards containing their 3D face data (template). The smart card would be read into the system, the user would get scanned, and the scan would be compared to the template on the smart card. If subjects are not cooperative (e.g., they do not maintain a frontal pose or remove eyeglasses), the quality of the scan my not be within the design specification and the scan may be rejected, which would require the subject to get scanned again. In order to claim that the system is practical, it must be evaluated based on the criteria presented in Table 8.5, which are discussed in the remainder of this section.

Subject Requirements

Most of the research reviewed and developed in this dissertation assumes the goal of matching frontal face scans from cooperative subjects. This is a reasonable assumption in many markets, which can assume that cooperative subjects are trying to gain access to restricted areas or resources, such as an airport, secured workplace or ATM. The cooperative subject assumption gives many additional restrictions. For example, if subjects are being cooperative it is reasonable for them to hold their position during the scan, maintain neutral expression, and face the camera. However, other commonly made assumptions may not be reasonable for even a cooperative subject. For example, it is not unreasonable for minor pose or expression variations to occur as a subject changes mood. It is also not unreasonable for a subject to change hairstyle, grow or shave a beard, or wear makeup.

The experiments in Chapter 5 demonstrated that the frontal matching system developed in this dissertation can easily tolerate pose variations of up to 30 degrees in the yaw direction and 10 degrees in the roll and pitch directions. These variations

Table 8.5: Questions that need to be addressed in order for a 3D face verification system to be considered practical.

Criterion	Description	How the criterion is addressed in this dissertation.
Subject Requirements	1. What are the pose requirements? 2. Are there any lighting restrictions? 3. What are the expression requirements? 4. Can the subjects change their hairstyle? Grow facial hair? Wear glasses or makeup? 5. How long will the enrolled data be valid before re-enrollment is required?	1. $\pm$ 15° in roll and pitch and $\pm$ 30° in yaw. 2. Extreme lighting causes the laser scanner to fail to gather an image. 3. Minimal cooperative expressions (i.e., slight smiles are okay). 4. Most hairstyles and facial hair are acceptable. 5. At least two years seems to be acceptable.
Cycle Time	1. How long does it take (from start to end) to process a single subject in the system? 2. Can this cycle time be improved by specialized hardware? 3. Are the time requirements sufficient for a real world application?	1. 9 seconds, including symmetry calculations and swap testing. 2. Many of the algorithms can be improved and hardware level systems have been investigated. 3. These speeds should be sufficient in applications involving security check points.
Enrollment	1. Can the model be fit onto the smart card? 2. How long does it take to load and save the model? 3. Does the enrollment process require expert intervention?	1. Two or three CFDM models ($<$ 250 kilobytes each) are possible. 2. Less than 0.2 seconds. 3. No.
Hardware Requirements and Cost	1. How much does the entire system cost? 2. Can improvements in hardware improve the functionality and speed of the system? 3. Can the system tolerate noise or holes in the data?	1. The Minolta Vivid 910 scanner is $50,000. 2. New technologies are being developed to improve cycle time and maintain accuracy. 3. The system has been shown to be robust to noise and holes.

are well within the limits of "frontal" scans, as observed in the variations measured in the FRGC database (see Section 5.3.3). Chapter 5 also showed that makeup and beards do not affect detection rates, although the current system can have problems with hair that occludes the upper part of the face or if the hair style captured with the scanner is very large in comparison to the face. (Large hair can cause extra noise in the data, making it difficult to localize the face using the current algorithm.)

Cycle Time

A cooperative subject can be expected to remain still during the scanning process, which can take 2 seconds or more when the Minolta scanner is used in fast resolution mode. With a stereo sensor, such as 3DMD [1], scan time can be less than 0.1 seconds; however, almost a minute of additional processing time is required. A cooperative subject is also assumed to be facing the camera and maintaining a neutral or near neutral expression. For face recognition applications where a cooperative subject cannot be assumed, such as in a surveillance situation, a faster scanner is required and the matching system must be invariant to all poses and lighting conditions. An arbitrary pose system was explored in Sections 4.4 and 5.5.

The entire cycle time for the current demo system was shown in Table 7.2 to be just above 9 seconds. These times are conservative and could easily be sped up by implementing faster algorithms (such as pyramidal methods), faster scanning hardware, or faster computing hardware. The estimate in Table 7.2 also includes the swap alignment step, which was shown to be unnecessary in Section 5.2.5 if the subject is cooperative. Removing these extra 1.3 seconds would reduce the total cycle time to under 8 seconds, which is reasonable in many applications. For example, at an airport security checkpoint subjects need to wait for x-ray scanners to examine their luggage, so an 8 second identity check is not burdensome.

Enrollment

The enrollment procedure for 3DID simply requires a subject to be scanned once. Improved results may be obtained by having an operator inspect the scan and add information, such as manual anchor points. However, as long as the subjects are cooperative, this additional work is not required in the current system.

Section 6.5 showed that the super sampled CFDM template (< 250 kilobytes) can easily fit onto a smart card (512KB [66]). There is room to store two or even three templates on a standard smart card, which could store different expressions or poses and help increase performance.

There is limited evidence to estimate how changes in the subject over time (aging effects) will impact the performance of the face verification system. In Section 5.4 the SAM score did not degrade in performance when the template and the target scan were taken approximately 1 year apart (this was not the case for the PCA algorithm, which degrades in performance). Informal studies in the MSU lab have shown that many subjects match well over a 3 year time period, and these experiments demonstrate that growing or shaving beards or changing hairstyle has almost no effect on the matching results.

Hardware Requirements and Cost

Probably the largest obstacle toward a practical use of a 3D face verification system is the cost of 3D scanners. The current high end scanners (Minolta and 3DMD) cost upwards of $50,000 each. This high price limits the possible applications (such as installing scanners in automatic teller machines). However, as described in Section 2.5.3, there are companies working on new 3D technologies that should be more affordable. If a scanner could be priced in the $2000 range, then 3D face verification may become affordable enough to overtake existing verification technologies, such as fingerprint or iris scanning.

8.1.3 Matching Score Evaluation for Face Verification

The experiments in this dissertation have presented several different matching score spaces that can be used to measure the similarity and difference between subjects. In Section 5.5, the shape index was fused with each control point, which slightly improved performance. In Chapter 6, PCA was applied to both the color and the shape channels of the CFDM modified scan. In each of these cases, the same input data is used but is processed in different ways in order to produce a matching score. Table 8.6 lists the various feature spaces explored in this dissertation.

Table 8.6: Matching Scores.

Feature Spaces	Described	Sections Used
Baseline SAM	Distance from 100 control points on the query scan to the surface of target model after trimming 10% of the the noisy data.	5
Shape Index	Distance measure between the shape index on the surface of the query and target scans.	5.4
Correlation	Root mean squared error over a windowed region on the target model searched over the query scan. Window regions were localized around the anchor points.	6.6.1
PCA	FRGC baseline algorithm after preprocessing using the CFDM.	6.6.2

The baseline matching system presented in this dissertation uses the trimmed root mean square distance (SAM) minimized by the ICP algorithm as the primary matching score for face scans. When subjects are cooperative, the SAM score is shown to be sufficient for doing proper face identification. In most cases, the surface alignment algorithm does a very good job aligning different surfaces, even if their SAM scores are high. However, using the SAM value as a matching measure is a fairly naive approach since the SAM is a measurement of the quality of the surface alignment and the surface of the face changes drastically with expression. What the experiments in Section 5.4 have shown is that comparisons of the same person with

minor changes in expression can produce larger SAM variations than comparisons of some impostors. This is true even when the control points are limited to the "non-deforming" parts of the face, as shown in Section 5.2.1 (although a smile can still push the cheek surface up into this region near the eyes).

Bellon *et al.* [13] showed that the Surface Infusion Measure (SIM) value achieved better performance over the SAM value. Bronstein *et al.* [26] used the surface normals as a matching score, but with limited success. These approaches treat the face as a large, rigid structure. In Section 6.6.1, moving window regions are used to compare the local surface of the face in an approach similar to one developed by Manjunath *et al.* [102], which has been shown to be more tolerant to global changes (such as expression). However, the experiments conducted in Section 6.6.1 show that simply fusing the correlation matching scores is less effective than using the SAM by itself.

The color channel of the Vivid 910 adds additional information about the face that is not available in the 3D surface data alone. It was demonstrated by Chang *et al.* [35] that the information provided in the color space is partially independent of the 3D shape space, and can be used to augment or even improve the shape space recognition. Thus, using the Canonical Face Depth Map as a preprocessor for a 2D color algorithm may achieve better performance by exploring linear subspaces, such as PCA.

The Face Recognition Grand Challenge not only provided a database of faces, but also provided a baseline algorithm for evaluating the faces. The FRGC algorithm is a Principal Component Analysis (PCA) algorithm similar to eigenfaces [133]. The PCA algorithm is applied to both the texture and shape channels separately, which are then fused together. The original PCA algorithm uses manually selected anchor points, which are not practical for an automatic system. In Section 6.6.2, the FRGC algorithm is made fully automatic by first preprocessing the scans using the Canonical Face Depth Map. First, the pose of each scan is corrected and each scan is converted

to a Canonical Face Depth Map using the automatic 3D face preprocessor. To further normalize all of the scans, the FRGC masking algorithm is used to properly crop all of the scans. The resulting normalized and preprocessed scans are fed into the standard FRGC version of the PCA algorithm. Section 6.6.2 shows that automatic PCA on the depth channel (73% at 0.01% FAR) performs better than the SAM score (68% at 0.01% FAR) or PCA (48% at 0.01% FAR) on the color channel. However, by adding all three scores, a verification rate of 81% is achieved at 0.01% FAR. Figure 8.3 plots SAM against the PCA distance measure; notice that the green and red values are seperating nicely. One area of future work is to develop a more advanced multimodal fusion system to combine these scores.

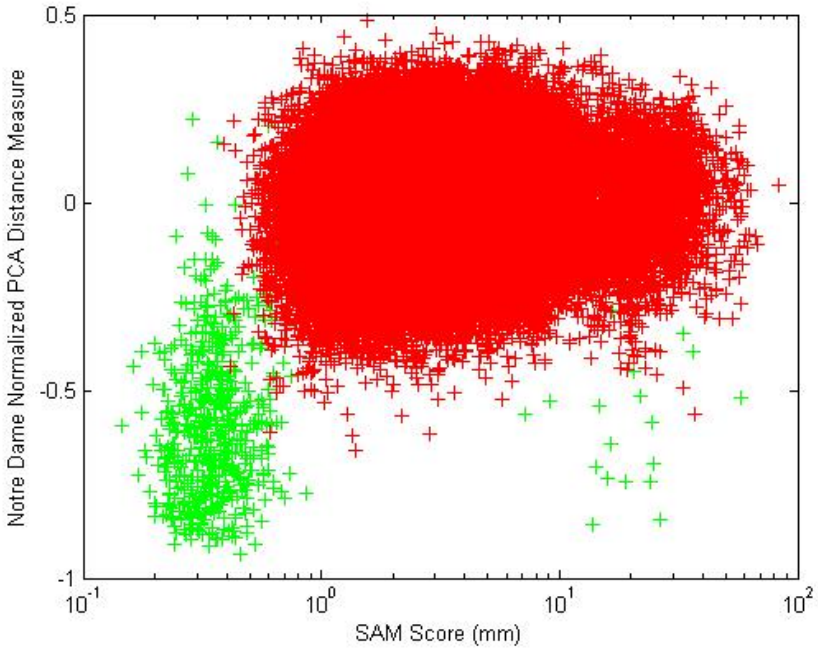


Figure 8.3: SAM score vs. PCA distance score on the MSU dataset. Red (dark) points represent impostors and green (light) points represent genuine.

Some of the outliers in Figure 8.3 represent bad data in the scans. The rejection option described in Section 5.2.6 can be applied to these scans, and the blue scores in Figure 8.4 represent results where one of the two matching scans is rejected.

In Section 6.6.1, correlation distance measures were also explored as matching

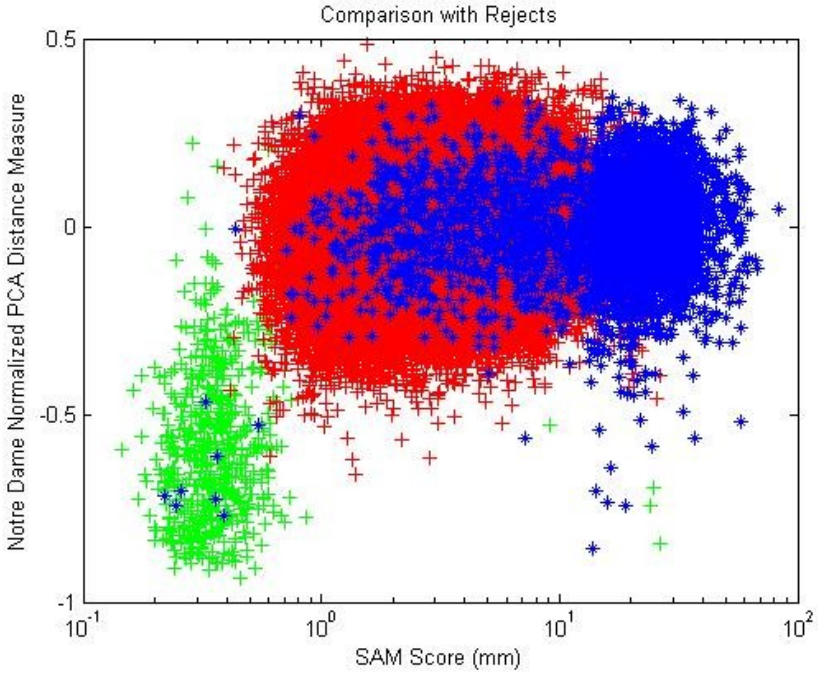


Figure 8.4: SAM score vs. PCA distance score on the MSU dataset. Red (dark) points represent impostors, green (light) points represent genuine and blue (darkest) points represent scores with either the model or the template scan rejected based on the symmetry criteria.

scores. Each individual score provided limited data, but together they produced results similar to the SAM score.

It is possible to construct a matching algorithm that performs even better than all the previous algorithms, as shown in Figure 8.5. In this multimodal matching algorithm, the SAM, bridge and nose correlation results are combined, producing much better results than any of the matching scores alone. Future work will extend these fusion methods to include other feature spaces and matching scores.

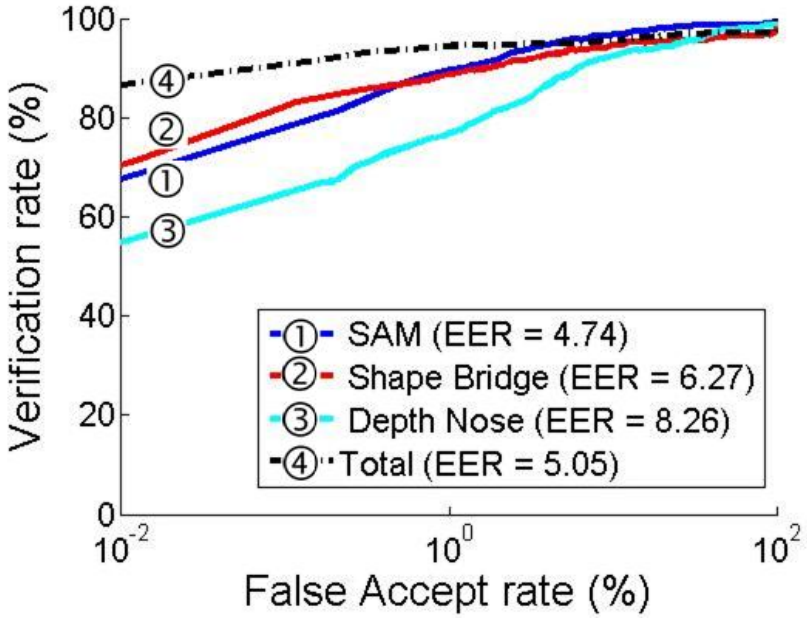


Figure 8.5: Fusion example. By fusing the nose correlation, bridge correlation and SAM score, significantly higher performance is achieved over any of the three matching scores by themselves.

8.2 Directions for Future Work

This dissertation explores the use of 3D data to enhance and extend face modeling and verification methods, which are relevant for a number of security and communications applications. There are a number of areas for future work based on this dissertation research.

The existing 3D databases have been shown to be a reasonable starting point for 3D face verification research; however, in order to further advance the state of the art additional 3D data should be collected. This new dataset should include many different poses as well as lighting conditions, and subjects should be encouraged to make reasonable variations to their appearance (such as growing beards, wearing makeup or changing their hairstyle). It is also important to include more pose and expression variations in the database to evaluate the feasibility of continuing the research into recognition and surveillance domains. No publicly available dataset truly explores all of these image variations.

This dataset with large intra-subject variation should include scans that could occur in any real world situation (indoor, outdoor, minor expressions, pose variations etc.) and should include scans that are invalid or noisy. Future work in this area could examine ways to measure the noise in a scan and identify environmental conditions that affect the quality of the scan. For instance, an improved 3D scanning algorithm could detect changes in the subject's pose or appearance (such as movement or the presence of eyeglasses) that could impact the success of the face verification system. Researching ways to identify these factors would allow the development of a system that could give feedback to the subjects by asking them to adjust their pose, maintain a more neutral expression, or remove eyeglasses and hats, etc.

With a larger, more realistic dataset, new feature spaces can be explored and compared to the existing systems. The CFDM has been shown to work well as an automatic preprocessor to existing face verification algorithms. However, the CFDM preprocessor has only been used for PCA and correlation, and there are many existing algorithms that may benefit from an automatic alignment system.

This dissertation explored the combination of various matching scores, such as SAM, correlation and PCA. These preliminary experiments suggest that combining the results of individual matching scores can improve the performance, yet there

are nearly unlimited ways that new matching scores can be explored. One example would be to further explore the combination of matching scores using weighted sums, which is a naive approach that produced some surprisingly positive results in this dissertation research.

Anchor points are another area where work remains. Improved methods for localization should be explored using the CFDM as a starting point. It is also useful to explore methods for properly evaluating how well anchor point localization is working. One approach would be to evaluate the quality of anchor point detection using ICP to first align the faces, and then calculating the distances between the anchor points. Another method would be to refine the anchor point detection results using manually selected points from the template. Applications using anchor points, such as expression modeling, could also be explored.

The algorithms and experiments presented in this dissertation demonstrate the potential of 3D data to improve face verification. 3D systems are more tolerant to changes in lighting, expression and pose than are 2D approaches, which must rely on color and texture information. However, 3D scanners are still prohibitively expensive for many applications, whereas a 2D face detection or identification system can be created using inexpensive and readily available cameras. 3D verification systems may not become mainstream until the cost of purchasing and maintaining the scanner drops. However, the research presented in this dissertation demonstrates that 3D data can be used to build a robust face verification system capable of achieving more than 99% accuracy for cooperative subjects. In addition to 3D face verification, the methods presented in this dissertation can be used in many fields. For instance, the 3D CFDM could be used in communications and entertainmaint, while the surface alignment algorithm could be helpful in medical domains.

Appendices

Appendix A

Programing Interface

Chapter 7 introduced the operation and use of the Research Platform developed for this dissertation. This Appendix describes the different aspects of the research platform in detail. Section A.1 discusses the various file formats used by the program, and analyzes the load/save times and memory requirements for each file type. Section A.2 describes the different external programing tools used during system development, and Section A.3 lists the programming libraries and SDKs used by the 3D research platform. Section A.4 details the different operating systems and platforms on which the 3D research platform has been successfully used.

A.1 File Formats

Many standards and data formats can be used to store face data. The research platform developed for this dissertation easily loads and saves several different data formats and allows new data formats to be added. The current set of supported data formats includes:

- **Matlab** (mat) - The Matlab file format can store any type of data object file. For the scanner data, the stored object is a struct. The base objects in the

struct were first populated by a program developed by Notre Dame to convert *.abs files into Matlab. The base objects include the x, y, z, and $flag$ matrix. In addition to the base class, the Matlab format can also store the surface normals, shape index, surface curvatures, and anchor points. The Matlab format can be used both by Matlab and by the 3D research platform's C++ code.

- **MSU 3BX** (3BX) - One problem with the Matlab format is that it requires a working copy of Matlab to compile and run, which may not be available on some systems. The 3BX format is a binary format designed to replace the Matlab format. It can store all the same information and is expandable to include future data types.
- **VRML** (wrl) - The Virtual Reality Markup Language (VRML) is a standard, web-based text format that can display all types of 3D information. This is a useful format because there are several VRML viewers available and many 3D applications can load VRML data. However, the VRML language is much more expressive than a 2.5D depth map and therefore the research platform is only able to write VRML files and cannot read them.
- **Vivid 910 File** (cdm) - The Vivid 910 has a native file format called a cdm file. These files contain the raw data from the scanner as well as scanner properties such as mode (fine/fast), focal distance, lens type, etc. These files can only be loaded and saved using the Vivid 910 SDK.
- **Dr. Flynn Format** (abs) - This is a simple ASCII format that stores the x, y, z and $flag$ values. This data, along with an associated image file, is the base format used by the Face Recognition Grand Challenge (FRGC).
- **PhotonX** (xyz) - This is a second ASCII format, similar to abs, developed to simplify data exchange with other researchers (PhotonX). In addition to the x, y, z coordinates, this format also allows surface normal files (nrm), which are the PhotonX scanners' intermediate raw data format.

Throughout this dissertation project all of these file formats have been used; Table A.1 summarizes the file formats available in the 3D research platform and highlights key contents of each format. Figure A.1 displays the formats used most commonly in experiments presented in this dissertation. Notice that the ABS format is much slower than the binary formats. This is because the ABS format is in ASCII and not designed for speed. Refer back to Figure 6.5 for a similar graph on the file memory requirements.

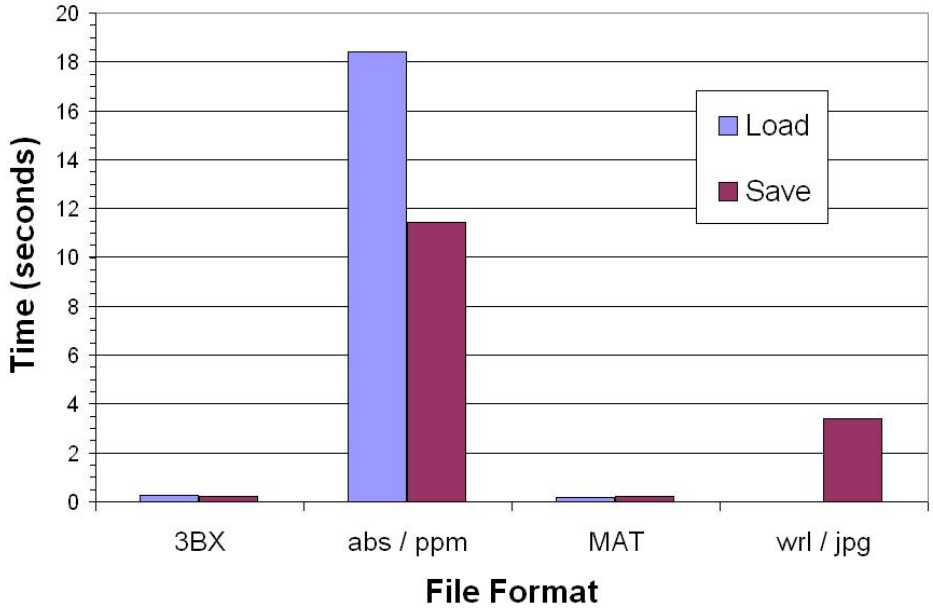


Figure A.1: Results of experiments to compare time taken to read and write common file types.

A.2 Programming Tools

This section describes the programming tools used in the various components of the 3D research platform. These tools enable the project to be multi-platform, simplify program editing, and improve program documentation.

Table A.1: File formats used by the 3DID system. A yes or no indicates whether the format can store a particular data type. A file extension is include if the data type is stored in a separate file.

Extension	Name	Input	Output	Format	XYZ	rgb	Shape	Normals	Expandable
mat	Matlab FaceScan	YES	YES	Binary Point Matrix	YES	YES	YES	YES	YES
3BX	MSU	YES	YES	Binary Point Matrix	YES	YES	YES	YES	YES
wrl	VRML	NO	YES	ASCII Polygonal Mesh	YES	YES	NO	NO	NO
cdm	Vivid 910 File	YES	Limited	Binary Point Matrix	YES	YES	NO	NO	NO
abs	Dr. Flynn format	YES	Yes	ASCII Point Matrix	YES	(ppm)	NO	NO	NO
xyz	PhotonX File	YES	YES	ASCII Point Matrix	YES	(ppm)	NO	(nrm)	NO
vla	Digistar	NO	YES	ASCII Point Cloud	YES	YES	NO	NO	NO

- **CMake** - CMake is an open source make utility for managing cross platform code trees [87]. CMake is a meta-level "make" program that takes configuration files and outputs working environments based on the operating system. For example, in Windows CMake takes the raw source code and outputs a working Visual Studio solution file; while in Linux CMake identifies the system compiler (gcc etc.) and outputs the makefiles required to build the system. The following list highlights key properties of the CMake system:

  1. Organizes cross platform builds in a text file.
  2. Works well with Windows Visual Studio, Cygwin, Linux, Solaris, HPC, etc.
  3. Organizes the code so that there is a clear separation between the source tree and the binary tree.
  4. Identifies and links common code packages, such as Image Magick and Matlab.
  5. Can be configured to work well with doxygen.

  One drawback of CMake is that the CMake program must be installed on each platform where it will be used. However, this is generally not problematic because CMake is a freely available, open source project that compiles easily on most systems.

- **Visual Studio** - The visual studio programming system is an excellent environment for tracking down programming bugs [105]. The 3D research platform can be compiled as a debug program and Visual Studio tracks which line of code is being executed, as well as the function call stack and local variables. Also, within the Visual Studio editor each function is hyper-linked to declaration, allowing for quick navigation within a large project.

The downside of using Visual Studio is that it requires a specialized project solution file. This file is simple to generate using Visual Studio wizards, but it is difficult and time consuming to manipulate because all of the settings are controlled by specialized dialog boxes within Visual Studio. Fortunately, Visual Studio solution files can also be generated using CMake, which allows for all of the advantages of the Visual Studio interface while avoiding many of the shortcomings.

- **Concurrent Versioning System** - The Concurrent Versioning System (CVS) is an open source program used to sync text files between many editors [60]. CVS runs on a system server and client programs on each user's system download the current version of the source code. After a user makes changes, he/she updates the code on the server using the client program. The CVS server keeps track of all of the changes and notifies the user if changes conflict with other programmers. Even though CVS can handle binary files, it cannot keep track of the types of changes within these files. This makes CVS difficult to use with Visual Studio files. However, CMake files and source code files work ideally with CVS.

- **Doxygen** - Doxygen is an open source, automatic documentation generation tool [136]. Doxygen uses special tags to strip out comments from code and format them to a human readable form, such as HTML, man pages, latex, etc. The advantage of using Doxygen is that code commenting and documentation are completed in the same step. Another advantage of Doxygen is that it creates an incentive to keep the code comments up to date.

A.3 Algorithm Components and Required SDKs

At a minimum, the 3D research platform developed in this dissertation requires a running copy of CMake and a C++ compiler. This section describes the set of additional packages and SDKs that can be used by the research platform. If the package is optional, CMake searches the host computer to find the package and, if found, links the new code.

- **Microsoft Foundation Classes** - The GUI in the demo program is designed as an MFC project in Microsoft Visual Studio [106]. This part of the system code will only run on Windows. CMake will identify the operating system and only loads the Demo program if the system is run on Windows. The main reason for limiting the demo program to a Windows environment is that the VIVID SDK is only designed to work in a Windows environment.

- **VIVID SDK** - The VIVID SDK is a software interface library designed to work directly with the VIVID 910 scanner and comes freely available with a new scanner [107]. The VIVID 910 uses a SCSI interface to communicate with the computer. However, the required SCSI drivers are only available for Windows. The current software interface between the VIVID 910 and the 3D research platform works only with MFC because the software uses a special MFC class to convert from the native cdm data structure into the 3D research platform's facescan data structure. There is no reason that the MFC classes could not be eliminated, but because of the limitations of the VIVID SDK eliminating the MFC would only enable the scanner to be used though Cygwin and the command line. Command line operation of the scanner is not particularly useful because there in no GUI interface to show the user where he/she is standing.

- **Template Numerical Toolkit** - The Template Numerical Toolkit (TNT) is a collection of public domain interfaces useful for mathematical computations

in C++ [123]. There are three primary data structures used throughout the code: Array2D, Array3D and Eigenvalues. Array2D and Array3D are used in the research platform to store all scanner matrices, while the Eigenvalue calculations are used for surface fitting. The TNT collection is required for almost all of the programs provided with the 3D research platform.

- **Matlab SDK** - Matlab comes installed with a Matlab SDK, which allows Matlab code to be executed from within other programs [104]. The SDK also allows programs to be developed that can load and save to the native Matlab format. This SDK was important for maintaining a database that could be loaded and run in both the Matlab version and C++ versions of the 3D Research Platform.

- **Image Magick SDK** - Image Magick is a freely available, open source SDK used for loading and saving digital images [97]. Image Magick is available across many platforms and is useful mostly because it can load a wide variety of image formats. However, installing and linking Image Magick is not always simple, and most of the functionality of Image Magick is also available with the other programming interfaces. For example, both Matlab and MFC provide a class for loading and saving images; however, neither of these classes are open source or freely available.

- **Biometrics Experiment Environment** - The Biometrics Experiment Environment (BEE) was developed by the National Institute of Standards and Technology (NIST) for use within the Face Recognition Grand Challenge (FRGC) [49]. BEE is a system of programs, data types, and interfaces that allows all the FRGC participants to communicate in a common language and submit their results to NIST in a common format. FRGC includes many experiments, of which Experiment 3 is the only one to deal with 3D data.

There are two types of programs required to run with BEE: preprocessors and "BioBoxes." A preprocessor is a program, such as the CFDM, that preprocesses each file. Strictly speaking, the preprocessor is not needed but is commonly used to normalize the data. The "BioBox" is a program for comparing two sets of scans (target and query). The output of the "BioBox" is a signature set that contains the matching scores for the target and query sets. There are several types of files that must be read and used by the research platform to work with BEE, including:

1. Signature Sets - XML files that contain the list of data files used by both the preprocessor and the biobox.
2. Workspace File - An XML file that tells BEE how to run the specific programs for specific datasets.

- **Xerces** - Xerces is an open source validating XML parser [59]. XML is extensively used by the Biometrics Experiment Environment (BEE), which also uses the Xerces parser. The current version of the 3D Research Platform only has the Xerces parser working in Cygwin and Linux. A BEE signature set XML reader was also developed independent of Xerces in order to allow this code to be run on HPC and Windows. If Xerces is not found, CMake will not include it in the project.

A.4 Cross Platform Computing

The 3D Research Platform developed for this dissertation is primarily designed for running scientific experiments. Its secondary functions are to demonstrate the algorithms and to provide a viable prototype for a a practical 3D verification system. Not all system functionality is available on all system platforms. For example, the

Biometrics Experiment Environment (BEE) will only run on a Linux system with administrator privileges. Table A.2 lists the various platform features that are available on the different systems.

Table A.2: Cross platform functionality.

	GUI Demo	VIVID 910 Control	Command Line Programs	Scripting Tools	XML Tools (Xerces)	BEE
Windows	Yes	Yes	Yes	No	No	No
Cygwin	No	No	Yes	Yes	Yes	No
Linux	No	No	Yes	Yes	Yes	Yes
HPC	No	No	Yes	Yes	Yes	Some

Due to development and system restrictions, none of the platforms shown in Table A.2 have all of the features available. So, it may be necessary to use more than one system when using the 3D Research Platform. Key features of the 3D Research Platform on each system include:

- **Windows** - In Windows, the complete 3D Research Platform can be compiled in Visual Studio. Because the demo program requires the Microsoft Foundation Classes (MFC), Windows is the only platform able to compile and run the demo. This is also true for operation of the VIVID 910 scanner, which requires a SCSI driver file only available in Windows. Operating the camera also requires a limited GUI interface that was also developed using MFC. In the future, the MFC library should be avoided if cross platform computability is desired; a library such as wxWidgets may be more useful in these circumstances [142].

- **Cygwin** - Another problem with Windows is that it cannot easily run the numerous scripting tools (such as grep, bash, perl, python, etc.) that are used to facilitate testing [69]. These tools are common on Linux systems but have to be installed independently for a Windows machine. The easiest way to install these programs is to install Cygwin. Cygwin is a Linux compatible interface written for Window; thus, a Windows system installed with Cygwin has the

most functionality for the 3D research platform. However, the combination of Windows and Cygwin does not allow BEE to be run.

- **Linux** - Installing the 3D Research Platform on Linux enables all of the research trial tools, including BEE, but it is not possible to run the Demo or gather data with the VIVID 910 scanner. Proper installation of BEE requires full administrator access on the host computer because the installation script for BEE assumes that some library and SDKs (such as Image Magick, Xerces, Java, etc.) will be installed in fixed paths. A new installation script was developed to work around this limitation by using a fake root path to install the required software. This method allowed the software to be installed on HPC without administrator privileges, but the script is a work in progress and does not always perform as expected.

- **High Performance Computing** - The 3D Research Platform was compiled on the two High Performance Computer (HPC) systems at MSU. The first system is a 64 1.6GHz Itanium2 processor SGI Altix 3700 BX2 with 512 Gigabytes of RAM. The second system is a 128 Node cluster from Western Scientific with Dual AMD Opteron 27, 2.2GHz, 2MB Cache, and dual core processors for each node (for a total of 512 processor cores). Both HPC computers are running Linux, which works well with the 3D Research Platform. However, many of the command line programs in the 3D Research Platform were rewritten to take full advantage of the parallel system.

Bibliography

Bibliography

[1] 3dMD. http://www.3dmd.com/. 2005.

[2] A4Vision. http://www.a4vision.com/. 2005.

[3] ABCNews. Scan suggests ex-sailor was WWII smoocher. http://abcnews.go.com/GMA/Technology/story?id=1036576. *ABC News Internet Ventures*, 2006.

[4] Bernard Achermann and Horst Bunke. Classifying range images of human faces with hausdorff distance. In *Proceedings of the 15th International Conference on Pattern Recognition*, volume 2, page 2809, Barcelona, Spain, 2000.

[5] Jörgen Ahlberg. Candide-3 – an updated parameterized face. Technical Report LiTH-ISY-R-2326, Linköping University, Sweden, 2001.

[6] Soon-Jeong Ahn, Jaechil Yoo, Byung-Gook Lee, and Joon-Jae Lee. 3D surface reconstruction from scattered data using moving least square method. In *Proccedings of the 13th International Conference on Image Analysis and Processing*, pages 719–726, Cagliari, Italy, 2005.

[7] Animetrics. http://www.animetrics.com/. 2006.

[8] A-Nasser Ansari and Mohamed Abdel-Mottaleb. Automatic facial feature extraction and 3D face modeling using two orthogonal views with application to 3D face recognition. *Pattern Recognition*, 38(12):2549–2563, 2005.

[9] Vera Bakic and George Stockman. Menu selection by facial aspect. In *Proceedings of Vision Interface '99*, Trois-Riviéres, Canada, 1999.

[10] Philippe Ballard and George Stockman. Controlling a computer via facial aspect. *IEEE Transactions on Systems, Man and Cybernetics*, 25(4):669–677, 1995.

[11] Roman Barták. Theory and practice of constraint propagation. In *Proceedings of the 3rd Workshop on Constraint Programming in Decision and Control*, pages 7–14, Gliwice, Poland, 2001.

[12] Marian Stewart Bartlett, Gwen Littlewort, Ian Fasel, and Javier R. Movellan. Real time face detection and facial expression recognition: Development and applications to human computer interaction. In *Proceedings of CVPR Workshop on Computer Vision and Pattern Recognition for Human-Computer Interaction*, volume 05, page 53, Los Alamitos, CA, USA, 2003.

[13] Olga Regina Pereira Bellon, Luciano Silva, and Chauã C. Queirolo. 3D face matching using the surface interpenetration measure. In *Proceedings of the 13th International Conference on Image Analysis and Processing*, pages 1051–1058, Cagliari, Italy, 2005.

[14] Chiraz BenAbdelkader and Paul A. Griffin. Comparing and combining depth and texture cues for face recognition. *Image and Vision Computing*, 23:339–352, 2005.

[15] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[16] Barry K.B. Berkovitz and Bernard J. Moxham. *A Textbook of Head and Neck Anatomy*. Year Book Medical Publishers, Inc., 1988.

[17] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[18] Charles Beumier and Marc Acheroy. Automatic face identification. In *Applications of Digital Image Processing XVIII, SPIE*, volume 2564, pages 311–323, 1995.

[19] Charles Beumier and Marc Acheroy. Automatic 3D face authentication. *Image and Vision Computing*, 18(4):315–321, 2000.

[20] J. Ross Beveridge, David Bolme, Bruce A. Draper, and Marcio Teixeira. The CSE face identification evaluation system. *Machine Vision and Applications*, 16(2):128–138, 2005.

[21] Volker Blanz and Thomas Vetter. Face recognition based on fitting a 3D morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1063–1074, 2003.

[22] Chris Boehnen and Patrick Flynn. Accuracy of 3D scanning technologies in a face scanning scenario. In *3DIM '05: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, pages 310–317, Ottawa, Ontario Canada, 2005.

[23] Chris Boehnen and Trina Russ. A fast multi-modal approach to facial feature detection. In *Proceedings of 7th IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)*, volume 01, pages 135–142, Breckenridge, Colorado, 2004.

[24] Kevin W. Bowyer, Kyong Chang, and Patrick Flynn. A survey of approaches to three-dimensional face recognition. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 01, pages 358–361, Cambridge, United Kingdom, 2004.

[25] Kevin W. Bowyer, Kyong Chang, and Patrick Flynn. A survey of approaches and challenges in 3d and multi-modal 3d + 2d face recognition. *Compututer Vision and Image Understanding*, 101(1):1–15, 2006.

[26] Alexander M. Bronstein, Micheal M. Bronstein, and Ron Kimmel. Expression-invariant 3D face recognition. In *IEEE International Workshop on Analysis and Modeling of Faces and Gestures (AMFG 2003)*, pages 232–233, Nice, France, 2003.

[27] Alexander M. Bronstein, Micheal M. Bronstein, and Ron Kimmer. Three-dimensional face recognition. *International Journal of Computer Vision*, 64(1):5–30, 2005.

[28] Myron Z. Brown, Darius Burschka, and Gregory D. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):993–1008, 2003.

[29] Roberto Brunelli and Tomaso Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.

[30] Canesta. http://www.canesta.com/, 2005.

[31] J. Y. Cartoux, J.T. LaPreste, and M. Richetin. Face authentification or recognition by profile extraction from range images. In *Proceedings of Workshop on Interpretation of 3D Scenes*, pages 194–199, Austin, Texas, November 1989.

[32] Marco La Cascia and Stan Sclaroff. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. *IEEE Transactions on Pattern Recognition and Image Processing*, 22(2):322–336, 2000.

[33] Kyong I. Chang, Kevin W. Bowyer, and Patrick J. Flynn. Face recognition using 2D and 3D facial data. In *Workshop on Multimodal User Authentication*, pages 25–32, Santa Barbara, California, 2003.

[34] Kyong I. Chang, Kevin W. Bowyer, and Patrick J. Flynn. Multi-modal 2D and 3D biometrics for face recognition. In *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, page 187, France, 2003.

[35] Kyong I. Chang, Kevin W. Bowyer, and Patrick J. Flynn. An evaluation of multimodal 2D+3D face biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):619–624, 2005.

[36] Hong Chen and Anil K. Jain. Dental biometrics: Alignment and matching of dental radiographs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1319–1326, 2005.

[37] Yang Chen and Gérard Medioni. Object modeling by registration of multiple range images. *International Journal of Image and Vision Computing*, 10(3):145–155, 1992.

[38] Yi Chen, Sarat C. Dass, and Anil K. Jain. Localized iris image quality using 2D wavelets. In *Proceedings of the International Conference on Biometrics (ICB)*, pages 373–381, Hong Kong, January 2006.

[39] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Kresek. The trimmed iterative closest point algorithm. In *Proceedings of the 16th International Conference on Pattern Recognitionn*, volume 3, pages 545–548, Québec City, Canada, 2002.

[40] Amit K. Roy Chowdhury and Rama Chellappa. Statistical bias in 3D reconstruction from a monocular video. *IEEE Transactions on Image Processing*, 14:1057–1062, 2005.

[41] C.S. Chua, F. Han, and Y. Ho. 3D human face recognition using point signature. In *Proceedings of IEEE 4th International Conference on Automatic Face and Gesture Recognition*, pages 233–238, Grenoble, France, 2000.

[42] C.S. Chua and R.A. Jarvis. Point signatures: A new representation for 3D object recognition. *International Journal of Computer Vision*, 21(1):63–85, 1997.

[43] Cognitec. FaceVACS face recognition software. http://www.cognitec-systems.de/, 2006.

[44] Philippe Colantoni, Nabil Bolukala, and Jérôme Da Rugna. Fast and accurate color image processing using 3D graphics cards. In *In Proceedings of 8th International Fall Workshop: Vision Modeling and Visualization*, Munich, Germany, 2003.

[45] Dirk Colbry, Xiaoguang Lu, Anil Jain, and George Stockman. 3D face feature extraction for recognition. Technical Report MSU-CSE-04-39, Michigan State University, September 2004.

[46] Dirk Colbry, George Stockman, and Anil Jain. Detection of anchor points for 3D face verification. In *Proceedings of Workshop Advanced 3D Imaging for Safety and Security*, San Diego California, 2005.

[47] Robert T. Collins, Alan J. Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, Osamu Hasegawa,

Peter Burt, and Lambert Wixson. A system for video surveillance and monitoring. *Technical Report CMU-RI-TR-00-12, Carnegie Mellon University Robotics Institute*, 2000.

[48] Cristina Conde and Ángel Serrano. 3D facial normalization with spin images and influence of range data calculation over face verification. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, page 115, 2005.

[49] Science Applications International Corporation. Biometrics experimentation environment (BEE) http://www.bee-biometrics.org/, 2004.

[50] CyberExtruder. http://www.cyberextruder.com/. 2005.

[51] Cyberware. http://www.cyberware.com/, 2006.

[52] Maltoni David, Dario Maio, Anil K. Jain, and Salil Prabhaker. *Handbook of Fingerprint Recognition*. Springer Verlag, 2003.

[53] Douglas DeCarlo, Dimitris Metaxas, and Matthew Stone. An anthropometric face model using variational techniques. In *Proceedings of International Conference on Computer Graphics and Intractive Techniques*, pages 67–74, 1998.

[54] Chitra Dorai and Anil K. Jain. Cosmos - a representation scheme for 3D free-form objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(10):1115–1130, 1997.

[55] Ioannis Douros and Bernard F. Buxton. Three-dimensional surface curvature estimation using quadric surface patches. In *Scanning 2002*, Paris, 2002.

[56] Nicolae Duta, Anil K. Jain, and Kanti V. Mardia. Matching of palmprints. *Pattern Recognition Letters*, 23(4):477–485, 2002.

[57] Leslie G. Farkas and Ian R. Munro. *Anthropometric Facial Proportions in Medicine*. Charles C Thomas, Springfield, Illinois, 1987.

[58] Patrick Joseph Flynn. *CAD-Based computer vision: Modeling and Recognition Strategies*. PhD thesis, Michigan State University, 1990.

[59] Apache Software Foundation. Xerces xml parser http://xml.apache.org/xerces-c/, 2006.

[60] Free Software Foundation. CVS - concurrent versions system. http://www.nongnu.org/cvs/, 2006.

[61] Jerome H. Freidman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3:209–226, September 1977.

[62] James Fung and Steve Mann. Computer vision signal processing on graphics processing units. In *Proceedings of Acoustics, Speech, and Signal Processing*, volume 5, pages 93–96, 2004.

[63] Sir Francis Galton. Personal identification and description. *Nature*, pages 173–177, 1888.

[64] Natasha Gelfand, Leslie Ikemoto, Szymon Rusinkiewicz, and Marc Levoy. Geometrically stable sampling for the icp algorithm. In *Proceedings of International Conference on 3D Digital Imaging and Modeling*, pages 260–267, Banff, 2003.

[65] Geometrix. http://www.geometrix.com/. 2005.

[66] Giesecke and Devrient. Big SIM cards from 512 kb to 512 mb. Technical report, http://www.gi-de.com/, 2006.

[67] Gaile Gordon. Face recognition based on depth and curvature features. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 108–110, 1992.

[68] A. Ardeshir Goshtasby. *2-D and 3-D Image Registration for Medical, Remote Sensing, and Industrial Applications*. Wiley, 2005.

[69] Inc. Red Hat. Cygwin DLL and utilities http://www.cygwin.com/, 2005.

[70] Curt Hesher, Anuj Srivastava, and Gordon Erlebacher. PCA of range images for facial recognition. In *Proceedings of International Multiconference in Computer Science*, Las Vegas, Nevada, 2002.

[71] Erik Hjelmas. Face detection: A survey. *Computer Vision and Image Understanding*, 83:236–274, 2001.

[72] Horace H.S.Ip and Lijun Yin. Constructing a 3D individualized head model from two orthogonal views. *The International Journal in Computer Graphics*, 12(5), 1996.

[73] Rein-Lien Hsu. *Face Detection and Modeling for Recognition*. Ph.D. Dissertation, Michigan State University, 2002.

[74] Rein-Lien Hsu and Anil K. Jain. Face modeling for recognition. In *Proceedings of the International Conference on Image Processing*, pages 693–696, Greece, 2001.

[75] Rein-Lien Hsu and Anil K. Jain. Generating discriminating cartoon faces using interacting snakes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1388–1398, 2003.

[76] Robert A. Hummel and Steven W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:267–287, 1983.

[77] Identix. FaceIT surveillance SDK http://www.identix.com/, 2005.

[78] ATI Technologies Inc. Visual technology. http://www.ati.com/technology/index.html, 2006.

[79] Anil Jain, Yi Chen, and Meltem Demirkus. Pores and ridges: Fingerprint matching using level 3 features. In *Proceedings of the 18th International Conference on Pattern Recognition*, Hong Kong, August 2006.

[80] Anil Jain, S. Prabhaker, and S. Pankanti. Twin test: On discriminability of fingerprints. In *Proceedings of the 3rd International Conference on Audio- and Video-Based Person Authentication*, pages 211–216, 2001.

[81] Anil K. Jain, Lin Hong, and Yatin Kulkami. A multimodal biometric system using fingerprint, face, and speech. In *Proceedings of the 2nd International Conference on Audio- and Video-based Biometric Person Authentication*, pages 182–187, Washington D.C., March 1999.

[82] Anil K. Jain, Arun Ross, and Sharath Pankanti. A prototype hand geometry-based verification system. In *Proceedings of 2nd International Conference on Audio- and Video-based Biometric Person Authentication*, pages 166–171, 1999.

[83] Anil K. Jain, Arun Ross, and Umut Uludag. Biometric template security: Challenges and soutions. In *Proceedings of the 14th European Signal Processing Conference (EUSIPCO)*, Antalya, Turkey, September 2005.

[84] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999.

[85] Javier Ortega-Garcia Julian Fierrez-Aguilar, Stephen Krawczyk and Anil K. Jain. Fusion of local and region approaches for on-line signature verification. In *Proceedings of the International Workshop on Biometric Recognition Systems (IWBRS)*, pages 188–196, Beijing, China, October 2005.

[86] Takeo Kanade. *Picture Processing System by Computer Complex and Recognition of Human Faces*. PhD thesis, Kyoto University, 1973.

[87] Insight Consortium Kitware Inc. Cmake cross-platform make. http://www.cmake.org/HTML/Index.html, 2002.

[88] Rotem Kowner. Psychological perspectives on human developmental stability and fluctuating asymmetry; sources, applications and implications. *British Journal of Psychology*, 92(3):447–469, 2001.

[89] Mitsubishi Electric Research Laboratories. http://www.merl.com/. Technical report, 2006.

[90] J. Lee and E. Milios. Matching range images of human faces. *Proceedings of IEEE International Conference on Computer Vision*, pages 722–726, 1990.

[91] Yonguk Lee, Hwanjong Song, Ukil Yang, Hyungchul Shin, and Kwanghoon Sohn. Local feature based 3D face recognition. In *Proceedings of the 5th International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 909–918, 2005.

[92] Yuencheng Lee, Demetri Terzopoulos, and Keith Walters. Realistic modeling for facial animation. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 55–62, New York, NY, USA, 1995. ACM Press.

[93] Alexandre Lemieux and Marc Parizeau. Experiments on eigenfaces robustness. In *Proceedings of the 16th International Conference on Pattern Recognition*, Quebec City,QC, Canada, 2002.

[94] Stan Z. Li and Anil K. Jain, editors. *Handbook of Face Recognition*. Springer Verlag, 2005.

[95] James J. Little and Jeffrey E. Boyd. Recognizing people by their gait: The shape of motion. *Videre: Journal of Computer Vision Research*, 1(2), 1998.

[96] Yanxi Liu, Robert T. Collins, and William E. Rothfus M.D. Automatic extraction of the central symmetry (mid-sagittal) plane from neuroradiology images. In *Proceedings of Medical Imaging Computing and Computer Assisted Intervention (MICCAI 2000)*, Pittsburgh, 2000.

[97] ImageMagick Studio LLC. http://www.imagemagick.org/, 2006.

[98] Xiaoguang Lu and Anil Jain. Deformation analysis for 3D face matching. In *Proceedings of IEEE Workshops on Applications of Computer Vision*, pages 99–104, Breckenridge, Colorado, 2005.

[99] Xiaoguang Lu and Anil K. Jain. Ethnicity identification from face images. In *Proceedings of SPIE International Symposium on Defense and Security : Biometric Technology for Human Identification*, volume 5404, Orlando, FL,, 2004.

[100] Ye Lu, Jason Z. Zhang, Q. M. Jonathan Wu, and Ze-Nian Li. A survey of motion-parallax-based 3-d reconstruction algorithms. *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews.*, 34(4):532–548, 2004.

[101] Sotiris Malassiotis and Michael G. Strintzis. Robust face recognition using 2D and 3D data: Pose and illumination compensation. *Pattern Recognition*, 38:2537–2548, 2005.

[102] B.S. Manjunath, R. Chellappa, and C. von der Malsburg. A feature based approach to face recognition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–378, 1992.

[103] Giovianni Marola. On the detection of the axes of symmetry of symmetric and almost symmetric planar images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(1):104–108, 1989.

[104] MathWorks. Matlab http://www.mathworks.com/, 2006.

[105] Microsoft. Visual studio. http://msdn.microsoft.com/vstudio/, 2005.

[106] Microsoft. MFC Microsoft Foundation Class Library. http://msdn.microsoft.com/, 2006.

[107] Minolta. Vivid software development kit II version 2.03, 2003.

[108] Minolta. Vivid 910 non-contact 3D laser scanner. http://konicaminolta.com.sg/products/industrial/instrument/vivid/vivid910.html, 2006.

[109] NIST. Report to the United States Congress. summary of NIST standards for biometric accuracy, tamper resistance, and interoperability. Technical report, National Institute of Standards and Technology, 2002.

[110] nVidia. Programmable graphics processor technolgies. http://www.nvidia.com/page/technologies.html. Web, 2006.

[111] Ojos. Internet photo search engine http://riya.com/, 2006.

[112] Javier Ortega-García, Joaquín González-Rodríguez, Danilo Simn-Zorita, and Santiago Cruz-Llanas. *From Biometrics Technology to Applications Regarding Face, Vocie, Signature and Fingerprint Recognition Systems*, chapter 12, pages 289–337. Kluwer Academic Publishers, 2002.

[113] Jörn Ostermann. Animation of synthetic faces in mpeg-4. In *Computer Animation*, pages 49–51, Philadelphia, Pennsylvania, 1998.

[114] Alice J. OToole, Fang Jiang, Dana Roark, and Hervé Abdi. *Face processing: Advanced modeling and methods.*, chapter 9, pages 293–319. 2006.

[115] Chia-Jung Pai, Hsiao-Rong Tyan, Yu-Ming Liang, Hong-Yuan Mark Liao, and Sei-Wang Chen. Pedestrian detection and tracking at crossroads. *Pattern Recognition*, 37:1025–1034, 2004.

[116] Gang Pan and Zhaohui Wu. 3D face recognition from range data. *International Journal of Image and Graphics*, 5(3):573–593, 2005.

[117] Igor S. Pandzic and Robert Forchheimer, editors. *MPEG–4 Facial Animation: The Standard, Implementation and Applications*. Wiley, 2002.

[118] P. Jonathon Phillips, Patrick J. Flynn, Todd Scruggs, Kevin W. Bowyer, Jin Chang, Kevin Hoffman, Joe Marques, Jaesik Min, and William Worek. Overview of the Face Recognition Grand Challenge. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[119] P. Jonathon Phillips, Patrick Grother, Ross J. Micheals, Duane M. Blackburn, Elham Tabassi, and Mike Bone. Face Recognition Vendor Test (FRVT), overview and summary. Technical report, National Institute of Standards and Technology, 2003.

[120] PhotonX. http://www.photon-x.com/, 2005.

[121] PixelVelocity. http://www.pixelvelocity.com/, 2006.

[122] Martha E. Pollack, Colleen E. McCarthy, Sailesh Ramakrishnan, Ioannis Tsamardinos, Laura Brown, Steven Carrion, Dirk Colbry, Cheryl Orosz, and Bart Peintner. Autominder: A planning, monitoring, and reminding assistive agent. In *Proceedings of 7th International Conference on Intelligent Autonomous Systems*, pages 265–272, 2002.

[123] Roland Pozo. TNT - template numerical toolkit. http://math.nist.gov/tnt/. NIST, 2004.

[124] Daniel Riccio and Jean-Lue Dugelay. Asymmetric 3D/2D processing: A novel approach for face recognition. In *Proceedings of the International Conference on Image Analysis and Processing*, pages 986–993, 2005.

[125] Arun Ross, Karthik Nandakumar, and Anil K. Jain. *Handbook of Multibiometrics*. Springer Verlag, 2006.

[126] Ashok Samal and Prasana A. Iyengar. Automatic recognition and analysis of human faces and facial expressions: A survey. *Pattern Recognition*, pages 65–77, 1992.

[127] Peter T. Sander and Steven Zucker. *Tracing surfaces for surfacing traces*. McGill University: McGill Research Centre for Intelligent Machines, 1988.

[128] Renata Scognamillo, Gillian Rhodes, Concetta Morrone, and David Burr. A feature-based model of symmetry detection. *Proceedings Biological Sciencees / Royal Society*, 270(1525):1727–1733, 2003.

[129] Dan Smith. Caricature of Jay Leno. http://www.dansmithartist.com/, 2006.

[130] Geomagic Studio. http://www.geomagic.com/en/, 2004.

[131] H. Tanaka, M. Ikeda, and H. Chiaki. Curvature-based face surface recognition using spherical correlation. In *Proceedings of IEEE 3rd International Conference on Automatic Face and Gesture Recognition*, pages 372–377, 1998.

[132] Carlo Tomasi and Takeo Kanade. Shape and Motion from Image Streams: A Factorization Method Part 2. Detection and Trcking of Point Features. Technical Report CMU-CS-91-132, April 1991.

[133] Mattew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[134] Alexander V. Tuzikov, Olivier Colliot, and Iisabelle Bloch. Brain symmetry plane computation in MR images using inertia axes and optimization. In *Proceedings of the 16th International Conference on Pattern Recognition*, Quebec City, Canada, 2002.

[135] Umut Uludag, Sharath Pankanti, and Anil K. Jain. Fuzzy vault for fingerprints. In *International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 577– 581, 2005.

[136] Dimitri van Heesch. Doxygen. http://www.doxygen.nl/, 2006.

[137] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[138] Paul Viola and Micheal Jones. Robust real-time object detection. In *Second International Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing, and Sampling*, Vancouver, Canada, 2001.

[139] Yingjie Wang, Chin-Seng Chua, and Yeong-Khing Ho. Facial feature detection and face recognition from 2D and 3D images. *Pattern Recognition Letters*, 23:1191–1202, 2002.

[140] James Wayman, Anil Jain, Davide Maltoni, and Dario Maio, editors. *Biometric Systems : Technology, Design and Performance Evaluation.* Springer, 2005.

[141] David M. Weinstein. The analytic 3-d transform for the least-squared fit of three pairs of corresponding points. School of Computing Technical Report UUCS-98-005, University of Utah, March 1998.

[142] wxWidgets. Cross-platform GUI library. http://wxwidgets.org/. SourceForge Project, 2006.

[143] Jing Xiao, Simon Baker, Iain Matthews, and Takeo Kanade. Real-time combined 2D+3D active appearance models. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 535–542, Washington D.C., 2004.

[144] Chenghua Xu, Yunhong Wang, Tieniu Tan, and Long Quan. Face recognition based on 3D mesh models. In *Proceedings of SPIE - The International Society for Optical Engineering.*, Denver, Colorado, 2004.

[145] Ping Yan and Kevin W. Bowyer. Ear biometrics using 2D and 3D images. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, volume 3, pages 121–121, June 2005.

[146] Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[147] Alper Yilmaz and Mubarak Shah. Automatic feature detection and pose recovery for faces. In *Proceedings of the Fifth Asian Conference on Computer Vision*, pages 284– 289, 2002.

[148] Lijun Yin and Matt T. Yourst. 3D face recognition based on high-resolution 3D face modeling from frontal and profile views. In *WBMA '03: Proceedings of the 2003 ACM SIGMM workshop on Biometrics methods and applications*, pages 1–8, New York, NY, USA, 2003. ACM Press.

[149] Hagit Zabrodsky, Shmuel Peleg, and David Avnir. Symmetry as a continuous feature. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:1154–1166, 1995.

[150] Haihong Zhang, Zhiyong Huang, Weimin Huang, and Liyuan Li. Kernel-based method for tracking objects with rotation and translation. In *Proceedings of the 17th International Conference on Pattern Recognition*, pages 728–731, Cambridge, United Kingdom, 2004.

[151] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706, August 1999.

[152] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(1):119–152, 1994.

[153] Zhi-Hua Zhou and Xin Geng. Projection functions for eye detection. *Pattern Recognition*, 37:1049–1056, 2004.