Face View Synthesis Using A Single Image

Jiang Ni

CMU-RI-TR-07-38

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

November 2007

Thesis Committee:
Henry Schneiderman, Chair
Martial Hebert
Alexei A. Efros
David Kriegman, University of California, San Diego

Abstract

Face view synthesis involves using one view of a face to artificially render another view. It is an interesting problem in computer vision and computer graphics, and can be applied in the entertainment industry for animated movies and video games. The fact that the input is only a single image, makes the problem very difficult. Previous approaches learn a linear model on pair of poses from 2D training data and then predict the unknown pose in the test example. Such 2D approaches are much more practical than approaches requiring 3D data and more computationally efficient. However they perform inadequately when dealing with large angles between poses. In this thesis, we seek to improve performance through better choices in probabilistic modeling. As a first step, we have implemented a statistical model combining distance in feature space (DIFS) and distance from feature space (DFFS) [28] for such pair of poses. Such a representation leads to better performance. As a second step, we model the relationship between the poses using a Bayesian network. This representation takes advantage of the sparse statistical structure of faces. In particular, we have observed that a given pixel is often statistically correlated with only a small number of other pixel variables. The Bayesian network provides a concise representation for this behavior reducing the susceptibility to over-fitting. Compared with the linear method, the Bayesian network more accurately predicts small and localized features.

Acknowledgements

First of all, I thank my advisor Henry Schneiderman, for inspiring and guiding me through these years of study and research at Carnegie Mellon. Henry encourages me to study many fundamental materials to understand the theories, and he gives me much freedom in choosing the thesis topic. I am also grateful for his revision of my thesis and helping my job hunting.

I also thank my thesis committee, Martial Hebert, Aloysha Efros and David Kriegman, for giving me valuable feedback and advices on my thesis work. They have also recommended me to talk to experts in their related fields. Such way of doing research does benefit me a lot.

I thank Eric Xing and Simon Lucey for the discussion of my thesis work. They give me very good advices, in Bayesian network and face research respectively. I have enjoyed their many discussions. I also thank Ralph Gross, Simon Baker, Leon Gu, and the University of South Florida, for providing me the databases, so that I can get a large set of training data in order to build a statistical model.

Finally, I thank my husband, my sister and my parents, for supporting and encouraging me with their endless love and care.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Problem

Face view synthesis is an interesting problem in computer vision and computer graphics. It involves using a photo of a person's face in one pose, to synthesize a photo of the person's face in a different pose. For example, if we know the frontal view of a face, such as a driver's license photo, we may wonder what its profile will look like (figure 1.1).

Face view synthesis can be used in the entertainment industry, such as animated movies or video games, for example, synthesizing a movie involving a historical figure such as Abraham Lincoln for whom a limited set of realistic images exist. Also, face view synthesis is one of the ways to aid face recognition for different poses. When the gallery images belong to a different pose other than the pose in the probe image, one way is to use face view synthesis to synthesize the desired pose in order to do the matching process.

Face view synthesis can be accomplished in a number of ways. If multiple images are available, one approach would be to extract and use the 3D geometry of the face. According to the stereopsis theory in computer vision, to recover the

Figure 1.1: A pair of frontal view and profile images of the same person, from CMU PIE database

precise 3D geometry of an object, we need at least two images of this object. This is why some approaches use multi-view images [13], or even video sequences [46], to synthesize new views. But in real life applications, often only a single image is available.

In this thesis, we are interested in the case where the input is only one single photo, which makes the problem difficult, because the information contained in one photo is insufficient to precisely recover the 3D geometry of the face. One may think this task impossible. But human brain is very good at performing this task. Suppose you are given a stranger's driver's license photo, by looking at the photo, you will probably have an idea what this person looks like. The trick is that you have seen many people's faces of different poses and have learned some relationship between poses. This prior knowledge helps you deal with strangers' photos. That is why driver's licenses only require one photo instead of multiple photos.

Machine learning techniques provide a natural method for incorporating prior knowledge into the synthesis problem. From a training set, we can learn the statistics of the geometric and appearance relationships between different poses, and apply these statistical relationships onto the single input image in order to synthe-

size new views. In particular, Vetter and his colleagues have developed several methods with this idea. Initially, they used only 2D images and applied linear object class method to solve for a partial least square problem [43]. Then they used one 3D face model to perform texture mapping between poses, with the occluded part synthesized by the linear object class method in separate regions of faces [42]. Later on, they built a 3D morphable model learned from 3D training examples, to synthesize novel views from a single image [5, 6, 7].

We aim to research face view synthesis with a single input image, building upon the prior knowledge from machine learning. The main ideas of the thesis are to find better representations of probability distributions relating facial appearance across pose:

1. We have implemented a regularized holistic linear probability model using distance in feature space (DIFS) and distance from feature space (DFFS). This representation improves upon previous methods that only use DFFS and leads to improved performance as described in Chapter 2 [30].

2. We have observed that statistical dependency varies among different groupings of pixel variables. In particular, a given pixel variable is often statistically correlated with only a small number of other pixel variables. We exploit this statistical structuring by modeling the synthesis problem using graphical probability models such as Bayesian Networks. Such representations concisely describe the synthesis problem, providing a rich model with reduced susceptibility to over-fitting. This idea is explained in Chapter 3.

1.2 Background and Related Work

Because of a face's 3D geometry, a single 2D image does not directly provide enough information to synthesize another view of the same face. In fact, Belhumeur et al. [1] showed that different 3D-shaped faces obtained via generalized

bas-relief transformations can look the same in 2D, given different lighting conditions. The extreme case is that from a single view it is not possible to differentiate an image of an object from an image of a flat photograph of the object [48].

Only by making various assumptions in combination with prior information is it possible to generate a prediction of what a face will look like from a different viewpoint. Below we list various assumptions made in earlier work.

1.2.1 Linear Object Class

In 1992 Poggio and Vetter proposed the concept of linear object class [34, 43]. This method relates feature locations in two viewpoints of the same object through a linear equation:

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \sum_{j=1}^{N} \alpha_j \begin{bmatrix} \phi_{1,j} \\ \phi_{2,j} \end{bmatrix}$$

where $\mathbf{y}_1$ and $\mathbf{y}_2$ are the feature locations in the two viewpoints. $\phi_{1,j}$ and $\phi_{2,j}$ are training examples or vectors derived from the training examples (e.g., principal components). This model involves two assumptions. The first assumption is that given a basis set ($\phi_{1,1}$, $\cdots$, $\phi_{1,N}$), there exists another basis set ($\phi_{2,1}$, $\cdots$, $\phi_{2,N}$) such that for any instance represented by the combination $\mathbf{y}_1 = \sum_{j=1}^{N} \alpha_j \phi_{1,j}$ in the first view, the instance in the second view will be accurately represented by $\mathbf{y}_2 = \sum_{j=1}^{N} \alpha_j \phi_{2,j}$. In order for this to hold, the object must be viewed under orthographic projection. In practice, this assumption requires that the variation in depth across a face is relatively small compared to the distance of the face to the camera. The second assumption is that the space of linear combinations of the training examples sufficiently describe the general class; that is, given a training set of the faces, any other face can be generated through a particular choice of $\alpha = (\alpha_1, \cdots, \alpha_N)^T$.

To use this model to synthesize $\mathbf{y}_2$ given $\mathbf{y}_1$, they solve for α in the standard

view using least squares

$$\alpha = (\alpha_1, \cdots, \alpha_N)^T = (\mathbf{\Phi}_1)^+ \mathbf{y}_1,$$

where $\mathbf{\Phi}_1 = (\phi_{1,1}, \cdots, \phi_{1,N})$ and $(\mathbf{\Phi}_1)^+$ is the pseudo-inverse of $\mathbf{\Phi}_1$. They then use the coefficients α to synthesize the virtual shape

$$\mathbf{y}_2 = (\phi_{2,1}, \cdots, \phi_{2,N})\, \alpha = \mathbf{\Phi}_2 \, (\mathbf{\Phi}_1)^+ \mathbf{y}_1,$$

where $\mathbf{\Phi}_2 = (\phi_{2,1}, \cdots, \phi_{2,N})$. Note that this method does not require correspondence between the two viewpoints, i.e., the feature points in $\mathbf{y}_1$ and $\mathbf{y}_2$ can be different landmarks. With no need for correspondence, this method can be applied to large rotations.

Gross et al. [17] used the similar math in their implementation of appearance-based face recognition and light-fields . This method has also been discussed in some related problems [24, 3, 26]. Hwang and Lee [24] use exactly the same method as above in predicting occluded parts of human faces. Black et al. [3] and Leonardis et al.[26] slightly modify the approach, by either excluding [26] or putting less weight [3] on some rows of $\mathbf{\Phi}_1$ that they assume are outliers.

However, experiments show that when the number of training examples is limited, the results are usually not good. The reason could be that a small number of training examples do not span the space of the class, therefore there exists a significant error term δ_1 and δ_2 in equations 1.1 and 1.2 below, in describing the novel object as linear combination of the examples.

$$\mathbf{y}_1 = \sum_{j=1}^{N} \alpha_j \phi_{1,j} + \delta_1 \tag{1.1}$$

$$\mathbf{y}_2 = \sum_{j=1}^{N} \alpha_j \phi_{2,j} + \delta_2 \tag{1.2}$$

The original linear object class method ignores those error terms, which leads to

the poor results.

Considering the error terms, Beymer and Poggio [2] assumed that δ_1 and δ_2 are highly correlated between the known view and the novel view. So they proposed that simply subtracting equation (1.2) from equation (1.1) can cancel out the error terms and get $\mathbf{y}_2$.

$$\mathbf{y}_2 = \mathbf{y}_1 + \sum_{j=1}^{N} \alpha_j \left(\phi_{2,j} - \phi_{1,j}\right)$$

Furthermore, instead of assuming δ_1 and δ_2 are equal, Sali and Ullman [35] found a way of computing how the error terms change from δ_1 to δ_2 in the rotation, in order to predict the error term δ_2 in the novel view. Assuming $\delta_2 = \mathbf{A}\delta_1$, they have

$$\mathbf{y}_2 = \sum_{j=1}^{N} \alpha_j \phi_{2,j} + \mathbf{A}\delta_1$$

where matrix $\mathbf{A}$ can be learned from a given transformation from the prototype examples. The cost in these two papers is the "cross correspondence", i.e., corresponding elements in $\mathbf{y}_1$ and $\mathbf{y}_2$ must describe the same physical point on the object. The correspondence between different views has to be computed in order to cancel out the error terms or determine the new error term. The "cross correspondence" may also restrict the method from dealing with large rotation angles, since it requires a set of feature points visible in both views.

Blanz et al. [4] proposed a regularization framework to solve this math problem, and applied in 3D shape reconstruction when part of the landmarks are known, with the requirement of lots of 3D training data obtained by a 3D laser scanner. They use Bayes rule to derive the tradeoff between DIFS and DFFS, leaving the weight factor η undetermined in the equation and having to try different η in the experiments. We describe in Chapter 2 a different way of derivation and reached the similar tradeoff between DIFS and DFFS, and in our derivation the weight factor η can be determined in the equation.

1.2.2 Lambertian Assumption and Texture Mapping

In 1998, Vetter introduced two variations [42] to the pure linear class method [43]. First, the linear class approach was applied to the parts of a face separately. Second, they used a 3D laser scanner to record 3D data of human heads, and averaged them to get a single 3D-model, which was used to establish pixelwise correspondence between the two reference face images in the two different poses. This correspondence field allows texture mapping across the view point change, which leads to better quality of synthesized texture in the visible pixels, while the occluded pixels are still synthesized using linear class method.

Wei [44] also used texture mapping techniques to synthesize novel views of faces. He built the 3D model for each input image by finding the symmetric landmarks and estimated depth. This method does not require 3D training data, because all the 3D information is recovered by the symmetry. The symmetric landmarks were labeled by hand. He also tried automatic detecting the symmetric landmarks by neural networks trained from 2D images, but did not integrate this automatic procedure into the whole framework.

1.2.3 3D Parametric Models

In 1999, Blanz and Vetter [5] proposed using a 3D morphable model in face view synthesis. With a database of many 3D human heads recorded via a 3D laser scanner, they performed principal component analysis (PCA) on these 3D data to learn the statistics in the shape and texture, to build a 3D morphable face model. When a new input face image is given, the coefficients of the 3D morphable model are optimized along with a set of rendering parameters such that they produce an image as close as possible to the input image. Once the optimization is done, the 3D model is rotated to give the 2D rendered face image in the desired view. Later they applied this approach to face recognition [6, 7].

The above two methods [42, 5] gave better results than [43], but incorporating

a 3D model requires hardwares such as 3D laser scanners, and the 3D model makes them computationally expensive, especially the optimization process of fitting the 3D model in [5] is very time-consuming.

Gu and Kanade [19] also proposed aligning a 3D deformable model to a single face image. They adjust the 3D deformable model to fit the image by comparing the view-based patches at the sparse points.

1.2.4 User-Specified Constraints

Li Zhang et al. [48] proposed a method that solves for single view shape reconstruction for free-form objects, such as portraits, sculptures, mountains, candies, under the assumption of orthographic projection. Their method requires user-specified constraints: point constraints, depth discontinuities, creases, planar region constraints, and fairing curve constraints. Point constraints specify the position or the surface normal of any point on the surface. Surface discontinuity constraints identify tears in the surface, and crease constraints specify curves across which surface normals are not continuous. Planar region constraints determine surface patches that lie on the same plane. Fairing curve constraints allow users to control the smoothness of the surface along any curve in the image. They also define a surface objective function as a measure of surface smoothness penalizing for large derivatives,

$$\begin{aligned} Q_0(g) &= \tfrac{1}{2d^2} \textstyle\sum_{i,j} \Big[\alpha_{i,j} \left(g_{i+1,j} - 2g_{i,j} + g_{i-1,j} \right)^2 \\ &\quad + 2\beta_{i,j} \left(g_{i+1,j+1} - g_{i,j+1} - g_{i+1,j} + g_{i,j} \right)^2 \\ &\quad + \gamma_{i,j} \left(g_{i,j+1} - 2g_{i,j} + g_{i,j-1} \right)^2 \Big] \end{aligned}$$

where $\alpha_{i,j}$, $\beta_{i,j}$, $\gamma_{i,j}$ are weights, $g_{i,j}$ is the depth value at grid (id, jd) on the depth map and d is the distance between adjacent grid cells. With the constraints and the objective function, they solve for a constrained optimization problem to find the 3D shape reconstruction that satisfies these constraints and optimizes the surface

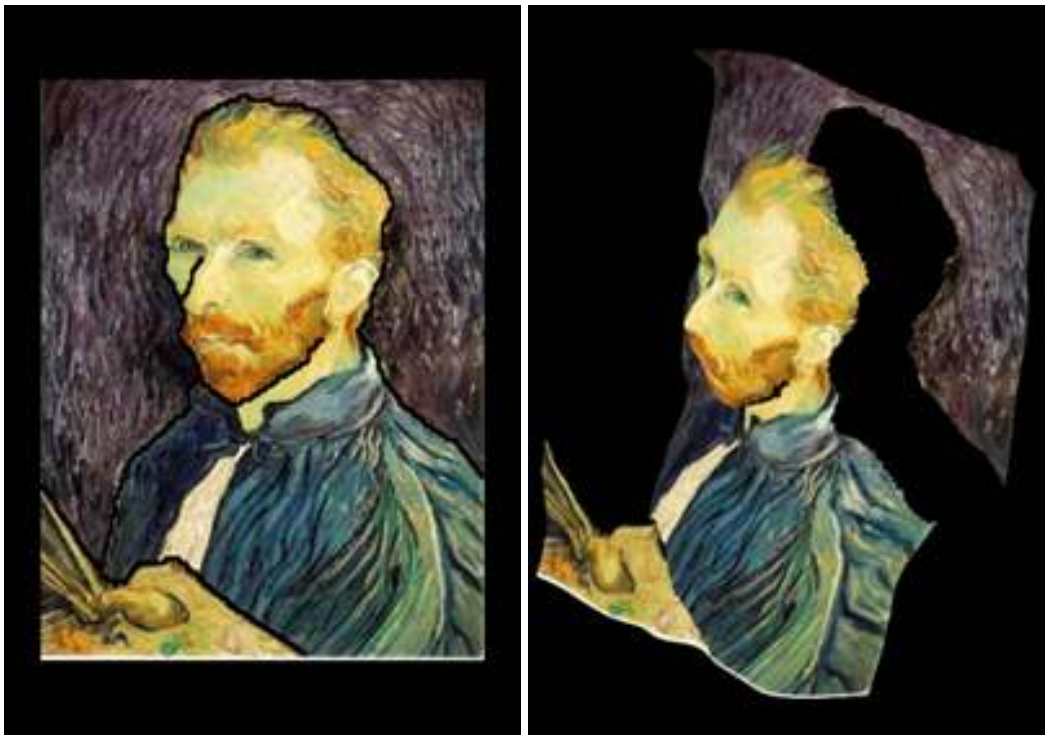

Figure 1.2: Results of Li Zhang et al.'s method[48]. The first image is a painting. The second image is its 3D reconstruction. Courtesy of Li Zhang.

objective function. Figure 1.2 is an example of their results.

1.2.5 Bilateral Symmetry

Wei Hong et al. [22] uses the bilateral symmetry as an assumption on some symmetric objects under perspective projection. They pointed out that one image of a symmetric object is equivalent to multiple images, and proved that symmetric objects can be recovered given only a single view. Assuming the symmetry relationship is fully known, e.g., let x be the 2D projection of a 3D point X, $g(x)$ denotes the symmetric pixel of x in 2D, where x, $g(x)$, X are all homogeneous coordinates. Matrix g represents the plane of symmetry in 3D. Then the projection

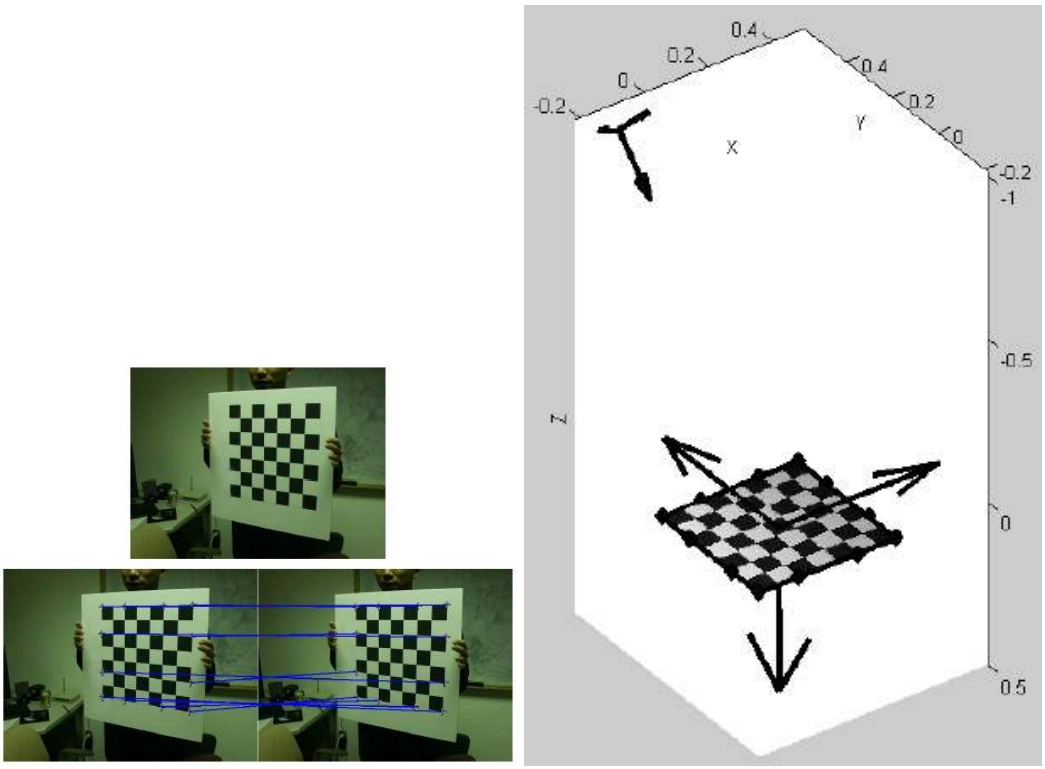

Figure 1.3: Results of Wei Hong et al.'s method[22]. The left images show a symmetric checker board and the corresponding points. The right image shows the 3D reconstruction. Courtesy of Wei Hong.

functions are

$$\lambda x = \Pi_0 g_0 X = [R_0, T_0]\, X$$

$$\lambda' g(x) = \Pi_0 g_0 g X$$

where $\Pi_0 = [I, 0] \in \mathcal{R}^{3\times 4}$, and g_0, R_0 and T_0 represent the camera parameters, i.e., the 3D relationship between camera and the object. This is a set of 6 equations with only 5 unknowns: 3 entries of X, λ and λ', and can be solved. Figure 1.3 is an example.

1.2.6 View-Based Active Appearance Model

Cootes et al.'s view-based active appearance model (AAM) [9] assumes a formula that describe the relationship between AAM parameters and the viewing angle.

$$c = c_0 + c_c cos(\theta) + c_s sin(\theta) \tag{1.3}$$

where c is a vector of the AAM parameters, and θ is the orientation angle. The constants c_0, c_c and c_s in the formula are learned from regression between $\{c_i\}$ and $\left\{(1,\ cos(\theta_i),\ sin(\theta_i))'\right\}$ on the training data. When given a first example image with parameters c, first they estimate θ in equation (1.3) as $tan^{-1}(y_a/x_a)$, where $(x_a, y_a)' = R_c^{-1}(c - c_0)$ and $R_c^{-1}$is the left pseudo-inverse of the matrix $(c_c|c_s)$. In order to do view synthesis at a new angle α, they compute the residual vector not explained by the rotation model at the first image as

$$c_{res} = c - (c_0 + c_c cos(\theta) + c_s sin(\theta))$$

and add this to the computed parameters at α:

$$c(\alpha) = c_0 + c_c cos(\alpha) + c_s sin(\alpha) + c_{res}.$$

1.2.7 Shape from Shading

In many AAM-based methods, Lambertian surfaces are assumed, and also the relative position between the light source and the object is supposed to be unchanged during the rotation. Therefore after warping, the pixel intensities are directly copied to the corresponding locations on the new pose. However, Zhao et al. [49] discussed the case where the absolutely position of the light source remains unchanged while the object is rotated. Under the shape-from-shading (SFS) theory, the pixel intensity changes between the two poses. The new pixel intensity can be computed given the rotation angle and the normals on both surfaces.

Figure 1.4: Results in Zhao et al.'s paper[49]. Column 1 and 4 are original frontal views. Column 2 and 5 are original rotated views. Column 3 and 6 are synthesized frontal views. Courtesy of Zhao.

However, in their experiments they did not use this SFS theory. Instead they directly copied the pixel intensities after warping, using a general 3D face model. Figure 1.4 shows their results, in which the synthesized frontal views usually have one side a little distorted because some pixels on that side are occluded in the rotated view.

1.2.8 Scene Geometry in Perspective Projection

Instead of assuming orthographic projection as the methods described in section 1.2.4, there is another class of single view reconstruction methods that use per-

spective projection cues to recover the 3D structures. These methods rely on assumptions in scene geometry. Horry et al. [23] and Criminisi et al. [12] reconstructed piecewise planar models based on user-specified vanishing points and geometric invariants. For example, they compute the perspective distortion of the parallel lines and other easy cues, in order to do the single view reconstruction. Their results are very interesting, for example, showing a virtual museum where you can walk into the 2D paintings such that the structures have 3D appearance. Hoiem et al. [21] added automatic segmentation to separate objects in the scene. However, these methods require that certain geometric easy cues exist, such as lines, planar surfaces, squares and so on. It is not clear, however, how such methods could be generalized to general objects of free-form, such as faces.

1.3 Our Approaches: Probabilistic Modeling

We aim to research face view synthesis with a single input image, building upon the prior knowledge from machine learning. The main ideas of the thesis are to find better representations of probability distributions relating facial appearance across pose:

1. We have implemented a regularized holistic linear probability model using distance in feature space (DIFS) and distance from feature space (DFFS). This representation improves upon previous methods that only use DFFS and leads to improved performance as described in Chapter 2.

2. We have observed that statistical dependency varies among different groupings of pixel variables. In particular, a given pixel variable is often statistically correlated with only a small number of other pixel variables. We exploit this statistical structuring by modeling the synthesis problem using graphical probability models such as Bayesian Networks. Such representations concisely describe the synthesis problem, providing a rich model with

reduced susceptibility to over-fitting. This idea is explained in Chapter 3

Chapter 2

Regularized Holistic Linear Model

The performance of face view synthesis depends on the accuracy and appropriateness of the probability model. In this thesis, we will explore several models beginning with an improved version of Vetter and Poggio's linear object class method [43]. We show how simple representation improves its performance. We then explore several observations that may lead to further improvements.

In the notations, we use uppercase bold fonts for matrices, lowercase bold fonts for vectors, and regular fonts for scalars.

2.1 Introduction

Vetter and Poggio's linear-object-class method [43] models images as linear sums of other images, and solves a set of linear equations with missing data:

$$\begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix} \mathbf{y} = \begin{pmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \end{pmatrix}, \tag{2.1}$$

where $\tilde{\mathbf{b}}_2$ is the unknown pose and $\tilde{\mathbf{b}}_1$ is the known pose of the test example. $\mathbf{\Phi} = \begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix}$ is the training set (or vectors formed from a linear combination of the training set, i.e., PCA of the training set) containing the two poses. $\begin{pmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \end{pmatrix}$ is represented as a linear combination of the columns in $\begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix}$. The vector $\mathbf{y}$ contains the parameters describing the linear combination. The linear-object-class method solves for $\mathbf{y} = \arg\min \left\| \mathbf{\Phi}_1 \cdot \mathbf{y} - \tilde{\mathbf{b}}_1 \right\|^2$, then uses it to predict $\tilde{\mathbf{b}}_2 = \mathbf{\Phi}_2 \mathbf{y}$. (In the view synthesis problem for faces, shape and appearance are usually analyzed and predicted separately.)

We believe that the problems with the linear-object-class method lie with an incorrect assumption: there are no errors inherent in the solution for $\mathbf{y}$ in $\mathbf{y} = \arg\min \left\| \mathbf{\Phi}_1 \cdot \mathbf{y} - \tilde{\mathbf{b}}_1 \right\|^2$. However, as it is well known, there are measurement errors in the training data due to many factors. These errors will propagate into the solution for $\tilde{\mathbf{b}}_2$, using the linear-object-class method. We can improve upon this solution with a probabilistic formulation. This formulation combines "distance-from-feature-space" (DFFS) and "distance-in-feature-space" (DIFS) [28], whereas the linear-object-class solution is purely based on DFFS. By considering DIFS, our method penalizes for points within the subspace, $\begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix}$, that have low probability (figure 2.1). Our representation leads to solutions that have higher probability and, as we will show, significantly better empirical performance.

Blanz et al. [4] also proposed a regularization framework to solve this math problem. The way they derive the probabilistic formulation was through Bayes rule, which is different from our derivation, although both lead to similar results. We will illustrate the difference in the chapter. Also we will analyze how to determine the weight factor which was undetermined in Blanz et al.'s approach.

This chapter is organized as follows. In Section 2.2, a probabilistic model

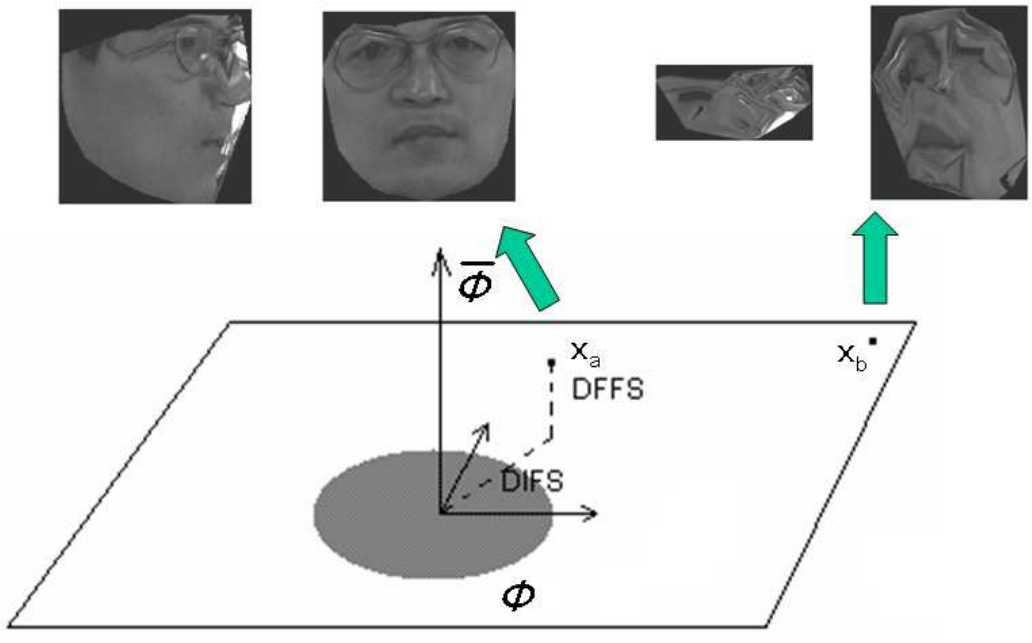

Figure 2.1: Linear object class will choose $\mathbf{x}_b$, whereas our method chooses $\mathbf{x}_a$

combining DFFS and DIFS is introduced, and the solution for equation (2.1) is derived. In Section 2.3, we explain the necessary steps of separating the shapes from the appearance of faces, and apply the solution in Section 2.2 to predict a new view of faces. Section 2.4 shows experimental results of synthetic face images at new views. In Section 2.5, we discuss this approach.

2.2 Probabilistic Modeling

The problem of linear equations with missing data described in equation (2.1) is restated in the following way:

We have N_T training vectors $\{\mathbf{x}_i\}_{i=1}^{N_T}$, each of which is an N-by-1 vector. Usually $N \gg N_T$. A test example $\mathbf{b} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$ belongs to the same class defined

by the training set. $\mathbf{b}$ is N-by-1. We only know $\mathbf{b}_1$, which contains the first N_1 elements of $\mathbf{b}$. The task is to predict $\mathbf{b}_2$, given $\mathbf{b}_1$ and $\{\mathbf{x}_i\}_{i=1}^{N_T}$.

2.2.1 Probabilistic Modeling

Let's first discuss the ideal case that we have enough independent training examples to span the whole N-dimensional space, i.e., $N_T \geq N$.

Here we apply several assumptions:

1. The class defined by the training set is an M-dimensional linear subspace, denoted as Φ. $M < N$ and determined by PCA from the training set. PCA is computed from the training set $\{\mathbf{x}_i\}_{i=1}^{N_T}$, and the M largest eigenvalues of the principal components $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$ are the variances along the M dimensions of Φ.

2. In this subspace Φ, samples are drawn from an M-dimensional Gaussian distribution with zero mean.

3. If the complete space has N dimensions, there is another $(N - M)$ dimensional linear subspace $\bar{\Phi}$, which is orthogonal and complementary to the eigenspace Φ (Fig. 2.2). We assume Φ and $\bar{\Phi}$ are statistically independent.

4. The samples also contain random noise distributed over all the $(N - M)$ dimensions of $\bar{\Phi}$. Each of the $(N - M)$ dimensions of $\bar{\Phi}$ has approximately equal non-zero variance, i.e., $\lambda_{M+1} \approx \lambda_{M+2} \approx \cdots \approx \lambda_N > 0$.

Under these assumptions, the probability of $\mathbf{x}$ is

$$P(\mathbf{x}|\Omega) = \left[\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{N/2}\prod_{i=1}^{N}\lambda_i^{1/2}} \right]$$

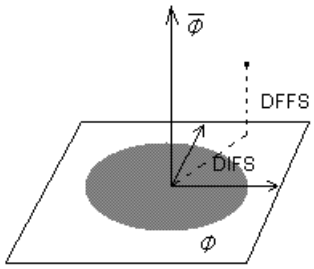

Figure 2.2: Decomposition into the eigenspace Φ and its orthogonal subspace $\bar{\Phi}$. The DFFS and DIFS are also shown.

$$
\begin{aligned}
&= \left[\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{M/2}\prod_{i=1}^{M}\lambda_i^{1/2}}\right]\cdot\left[\frac{\exp\left(-\frac{1}{2}\sum_{i=M+1}^{N}\frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{(N-M)/2}\prod_{i=M+1}^{N}\lambda_i^{1/2}}\right] \\
&= P_{\Phi}(\mathbf{x}|\Omega)\cdot P_{\bar{\Phi}}(\mathbf{x}|\Omega),
\end{aligned}
$$

where Ω denotes the class described by the training set. $\mathbf{x}$ is a random point from this class, and its projection onto each dimension is denoted as $\{y_i\}_{i=1}^{N}$. $P_{\Phi}(\mathbf{x}|\Omega)$ and $P_{\bar{\Phi}}(\mathbf{x}|\Omega)$ are two marginal Gaussian distributions, in Φ and $\bar{\Phi}$ respectively. In Blanz et al.[4]'s paper, they modeled the posterior probability $P(\mathbf{c} \mid \mathbf{x}) = \nu \cdot P(\mathbf{x} \mid \mathbf{c}) \cdot p(\mathbf{c})$ by Bayes rule where $c_i = \frac{y_i}{\lambda_i}$, $i = 1, \cdots, M$.

In practice, since N is very large and $N \gg N_T$, we lack sufficient data to compute each $\{\lambda_i\}_{i=M+1}^{N}$ in $P_{\bar{\Phi}}(\mathbf{x}|\Omega)$. Recall the assumption that $\{\lambda_i\}_{i=M+1}^{N}$ are about the same magnitude. Then it is reasonable to use the arithmetic average $\rho = \frac{1}{N-M}\sum_{i=M+1}^{N}\lambda_i$ [28] to get an estimation of $P(\mathbf{x}|\Omega)$, which is

$$
\hat{P}(\mathbf{x}|\Omega) = P_{\Phi}(\mathbf{x}|\Omega)\cdot\hat{P}_{\bar{\Phi}}(\mathbf{x}|\Omega)
$$

$$= \left[\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right)}{(2\pi)^{M/2}\prod_{i=1}^{M}\lambda_i^{1/2}}\right]\cdot\left[\frac{\exp\left(-\frac{1}{2\rho}\cdot\sum_{i=M+1}^{N}y_i^2\right)}{(2\pi\rho)^{(N-M)/2}}\right].$$

The distance characterizing the $\hat{P}(\mathbf{x}|\Omega)$ is

$$\hat{d}(\mathbf{x}) = \left[\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right] + \frac{1}{\rho}\cdot\left[\sum_{i=M+1}^{N}y_i^2\right]. \tag{2.2}$$

In Blanz et al.[4]'s derivation, they maximized the posterior probability $P(\mathbf{c} \mid \mathbf{x})$ and reached similar characteristic distance $E = \|\mathbf{c}\|^2 + \eta\cdot\|\mathbf{Q}\mathbf{c}-\mathbf{x}\|^2$. In fact $\|\mathbf{c}\|^2 = \left[\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right]$, $\|\mathbf{Q}\mathbf{c}-\mathbf{x}\|^2 = \left[\sum_{i=M+1}^{N}y_i^2\right]$. In their derivation, the weight factor η is left undetermined, while we determine this weight factor in Section 2.2.4.

In our problem, we only know the upper part of $\mathbf{b} = \begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix}$, and know it is from class Ω. In order to solve for the unknown part $\mathbf{b}_2$, we want to maximize the likelihood of $\hat{P}(\mathbf{b}|\Omega)$ by choosing $\{y_i\}_{i=1}^{N}$, where $\{y_i\}_{i=1}^{N}$ are the measurements of projecting $\mathbf{b}-\bar{\mathbf{x}}$ onto each of the N dimensions and $\bar{\mathbf{x}}$ is the mean of the training set. We then generate $\mathbf{b}_2 = \bar{\mathbf{x}}_2 + \mathbf{\Phi}_2\cdot\mathbf{y}$. This optimization depends upon three quantities: $\left[\sum_{i=1}^{M}\frac{y_i^2}{\lambda_i}\right]$, $\left[\sum_{i=M+1}^{N}y_i^2\right]$ and the weight ρ. Let's look at them one by one.

2.2.2 Distance in Feature Space

This is the Mahalanobis distance, also called the "distance-in-feature-space" (DIFS) [28]. It describes how far the projection of $\mathbf{x}$ onto Φ is from the origin. Let

$$\mathbf{\Lambda}_M^{-1} = \begin{bmatrix} \frac{1}{\lambda_1} & & & 0 \\ & \frac{1}{\lambda_2} & & \\ & & \ddots & \\ 0 & & & \frac{1}{\lambda_M} \end{bmatrix}, \text{ and } \mathbf{y} = (y_1, y_2, \cdots, y_M)^T, \text{ then}$$

$$\left[\sum_{i=1}^{M} \frac{y_i^2}{\lambda_i}\right] = \mathbf{y}^T \mathbf{\Lambda}_M^{-1} \mathbf{y}. \tag{2.3}$$

2.2.3 Distance from Feature Space

The residual reconstruction error, also called DFFS [28] is $\sum_{i=M+1}^{N} y_i^2 = \epsilon^2(\mathbf{x}) = \|\mathbf{x}' - \mathbf{x}\|^2$, where $\mathbf{x}'$ is the projection of $\mathbf{x}$ on Φ.

The linear-object-class method [43] minimizes the DFFS to find $\mathbf{y}$ in order to predict $\mathbf{b}_2$. Split the eigenvector matrix $\mathbf{\Phi}$ containing the first M eigenvectors into $\mathbf{\Phi} = \begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix}$ and split the mean $\bar{\mathbf{x}}$ of training data into $\bar{\mathbf{x}} = \begin{pmatrix} \bar{\mathbf{x}}_1 \\ \bar{\mathbf{x}}_2 \end{pmatrix}$, where $\mathbf{\Phi}_1$ and $\bar{\mathbf{x}}_1$ have the same number of rows as $\mathbf{b}_1$. No matter what method we use to solve for $\mathbf{y}$, since $\mathbf{b}_2$ is defined as $\bar{\mathbf{x}}_2 + \mathbf{\Phi}_2 \cdot \mathbf{y}$, the residual reconstruction error of resulting $\mathbf{b} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$ is

$$\begin{aligned} \sum_{i=M+1}^{N} y_i^2 = \epsilon^2(\mathbf{b}) &= \|\mathbf{b} - (\bar{\mathbf{x}} + \mathbf{\Phi} \cdot \mathbf{y})\|^2 = \left\| \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix} - \begin{pmatrix} \bar{\mathbf{x}}_1 + \mathbf{\Phi}_1 \cdot \mathbf{y} \\ \bar{\mathbf{x}}_2 + \mathbf{\Phi}_2 \cdot \mathbf{y} \end{pmatrix} \right\|^2 \\ &= \|\mathbf{b}_1 - (\bar{\mathbf{x}}_1 + \mathbf{\Phi}_1 \cdot \mathbf{y})\|^2 \\ &= \left\|\tilde{\mathbf{b}}_1 - \mathbf{\Phi}_1 \cdot \mathbf{y}\right\|^2 , \end{aligned} \tag{2.4}$$

where $\tilde{\mathbf{b}}_1 = \mathbf{b}_1 - \bar{\mathbf{x}}_1$. Thus the linear-object-class method [43] solves a least square problem to solve for $\mathbf{y}$:

$$\mathbf{y} = \arg\min \left\|\mathbf{\Phi}_1 \cdot \mathbf{y} - \tilde{\mathbf{b}}_1\right\|^2 .$$

2.2.4 Determining the Weight

Moghaddam and Pentland [28] define $\rho = \frac{1}{N-M}\sum_{i=M+1}^{N}\lambda_i$, under the assumption that the number of training examples $N_T \geq N$, and that $\{\lambda_i\}_{i=M+1}^{N}$ are about the same magnitude. However, in practice, N is very large and we have $N_T \ll N$. These N_T training examples can only span an $(N_T - 1)$ dimensional subspace, resulting in that $\lambda_{N_T} = \lambda_{N_T+1} = \cdots = \lambda_N = 0$.

We use the non-zero eigenvalues, $\{\lambda_i\}_{i=M+1}^{N_T-1}$, to guess what $\{\lambda_i\}_{i=N_T}^{N}$ would be like had we been given sufficient training data. Here we add another assumption:

- We assume that the actual values of $\{\lambda_i\}_{i=N_T}^{N}$ will be about the same magnitude as the average of the known eigenvalues $\{\lambda_i\}_{i=M+1}^{N_T-1}$.

Under this assumption, $\rho = \frac{1}{N_T-M-1}\sum_{i=M+1}^{N_T-1}\lambda_i$.

2.2.5 Solving the Optimization Problem

Given $\mathbf{b}_1$, we want to find $\mathbf{b}_2$ that minimizes $\hat{d}(\mathbf{b})$. Substituting equations (2.3) and (2.4) into (2.2),

$$\begin{aligned}
\hat{d}(\mathbf{b}) &= \sum_{i=1}^{M}\frac{y_i^2}{\lambda_i} + \frac{\epsilon^2(\mathbf{b})}{\rho} \\
&= \mathbf{y}^T\mathbf{\Lambda}_M^{-1}\mathbf{y} + \frac{1}{\rho}\left\|\tilde{\mathbf{b}}_1 - \mathbf{\Phi}_1\cdot\mathbf{y}\right\|^2 \\
&= \mathbf{y}^T\mathbf{\Lambda}_M^{-1}\mathbf{y} + \frac{1}{\rho}\left(\tilde{\mathbf{b}}_1 - \mathbf{\Phi}_1\cdot\mathbf{y}\right)^T\left(\tilde{\mathbf{b}}_1 - \mathbf{\Phi}_1\cdot\mathbf{y}\right) \\
&= \frac{1}{\rho}\left(\mathbf{y}^T\rho\mathbf{\Lambda}_M^{-1}\mathbf{y} + \mathbf{y}^T\mathbf{\Phi}_1^T\mathbf{\Phi}_1\mathbf{y} - 2\left(\mathbf{\Phi}_1^T\tilde{\mathbf{b}}_1\right)^T\cdot\mathbf{y} + \tilde{\mathbf{b}}_1^T\tilde{\mathbf{b}}_1\right).
\end{aligned}$$

Letting the partial derivative to be zero,

$$\begin{aligned} 0 = \frac{\partial \hat{d}(\mathbf{b})}{\partial \mathbf{y}} &= 2\rho\mathbf{\Lambda}_M^{-1}\mathbf{y} + 2\mathbf{\Phi}_1^T\mathbf{\Phi}_1\mathbf{y} - 2\mathbf{\Phi}_1^T\tilde{\mathbf{b}}_1 \\ &= 2\left[\left(\rho\mathbf{\Lambda}_M^{-1} + \mathbf{\Phi}_1^T\mathbf{\Phi}_1\right)\mathbf{y} - \mathbf{\Phi}_1^T\tilde{\mathbf{b}}_1\right]. \end{aligned}$$

The solution of $\mathbf{y}$ is

$$\mathbf{y} = \left(\rho\mathbf{\Lambda}_M^{-1} + \mathbf{\Phi}_1^T\mathbf{\Phi}_1\right)^{-1} \cdot \mathbf{\Phi}_1^T\tilde{\mathbf{b}}_1. \tag{2.5}$$

And the unknown $\mathbf{b}_2$ can be predicted as $\mathbf{b}_2 = \bar{\mathbf{x}}_2 + \mathbf{\Phi}_2 \cdot \mathbf{y}$.

2.3 Separating Shape and Appearance

Let's use the above technique to solve the problem of synthesizing new views of human faces. The problem is described as follows. Given a probe face image $\mathbf{I}$ under pose 1, we need to synthesize a new image $\mathbf{J}$ of this person's face under pose 2. The training set consists of N_T pairs of face images, $\{[\mathbf{I}_1, \mathbf{J}_1], [\mathbf{I}_2, \mathbf{J}_2], \cdots, [\mathbf{I}_{N_T}, \mathbf{J}_{N_T}]\}$. $\mathbf{I}_i$ and $\mathbf{J}_i$ are faces of the ith subject in the training set. $\{\mathbf{I}_i\}_{i=1}^{N_T}$ are under pose 1, and $\{\mathbf{J}_i\}_{i=1}^{N_T}$ are under pose 2.

In our approach, we make the common assumption [10, 43, 17] that the characteristics of shape can be separated from appearance.

2.3.1 Shape

On each face image, a set of landmarks are labeled by hand. For the ith training image under pose 1, denote the coordinates of each landmark as (x_j, y_j), $j = 1, \cdots, L_1$, where L_1 is the number of landmarks on the faces under pose 1. Define the shape vector of this ith face image under pose 1 as

$$\mathbf{s}_{i,1} = (x_1, x_2, \cdots, x_{L_1}, y_1, y_2, \cdots, y_{L_1})^T.$$

A similar vector $\mathbf{s}_{i,2}$ can also be defined in the same way for pose 2. Concatenating these two vectors, we get a vector

$$\mathbf{s}_i = \begin{pmatrix} \mathbf{s}_{i,1} \\ \mathbf{s}_{i,2} \end{pmatrix}$$

as a combined shape vector for the ith subject in the training set.

Thus, for the N_T subjects in the training set, we get a training set of shape vectors $\{\mathbf{s}_i\}_{i=1}^{N_T}$.

2.3.2 Appearance

For each pose, a reference face is chosen by averaging over the training set, so that every face under this pose is warped to the shape of the reference face, giving a normalized image (Figure 2.3). The warping is done by first connecting the landmarks into a set of triangles via Delaunay triangulation algorithm[1] and then doing an affine transform to each triangle [10] (Figure 2.4), assuming that the faces have Lambertian surfaces. On each normalized image, only the pixels within the convex hull of the landmarks are kept and all other pixels are discarded. This is done to remove the unnecessary variations of the hair or the background scenery. Let's call the resultant normalized images under pose 1 as $\widetilde{\mathbf{I}_1}, \widetilde{\mathbf{I}_2}, \cdots, \widetilde{\mathbf{I}_{N_T}}$, and those under pose 2 as $\widetilde{\mathbf{J}_1}, \widetilde{\mathbf{J}_2}, \cdots, \widetilde{\mathbf{J}_{N_T}}$. Reshape them into vectors as $\{\mathbf{t}_{i,1}\}_{i=1}^{N_T}$ and $\{\mathbf{t}_{i,2}\}_{i=1}^{N_T}$ for pose 1 and pose 2 respectively.

For the ith subject in the training set, define

$$\mathbf{t}_i = \begin{pmatrix} \mathbf{t}_{i,1} \\ \mathbf{t}_{i,2} \end{pmatrix}$$

[1]Given a set of data points, the Delaunay triangulation algorithm connects them as a set of triangles such that no data points are contained in any triangle's circumscribed circle. See http://www.mathworks.com/access/helpdesk/help/techdoc/ref/delaunay.html

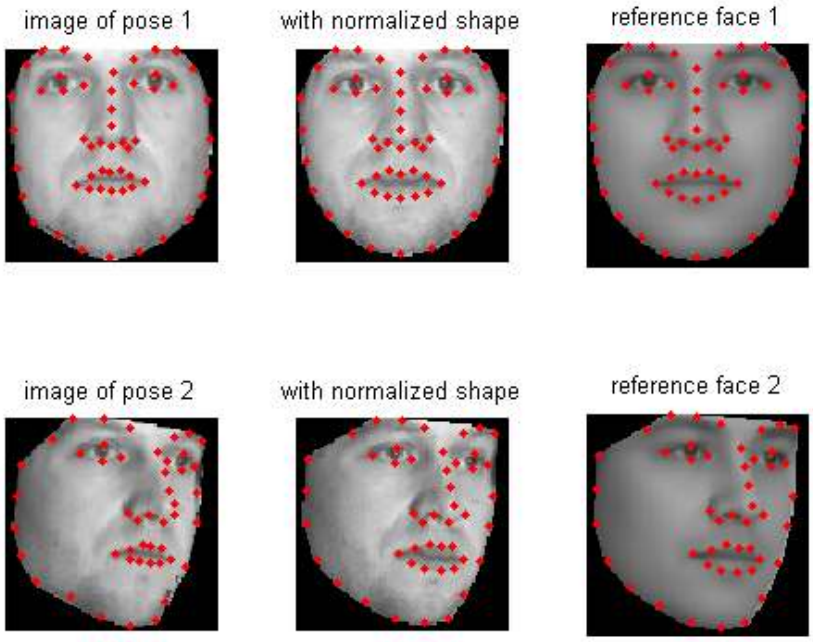

Figure 2.3: An example showing the separation of shape and appearance. In each of the two poses, the image is labeled with some landmarks and then warped to the normalized shape of the reference face in that pose. The reference faces are obtained by averaging the shapes and appearances of the training images.

as a combined appearance vector. Thus, for the N_T subjects in the training set, we get a training set of appearance vectors $\{\mathbf{t}_i\}_{i=1}^{N_T}$.

2.3.3 Probe Image and Prediction

Given a probe face image $\mathbf{I}$ under pose 1, we need to synthesize a new image $\mathbf{J}$ of this person's face under pose 2. With a set of landmarks on $\mathbf{I}$ and the reference face under pose 1, we can again decompose $\mathbf{I}$ into its shape vector $\hat{\mathbf{s}}_1$ and appearance vector $\hat{\mathbf{t}}_1$. The landmarks can be obtained using AAM fitting [10]. In our

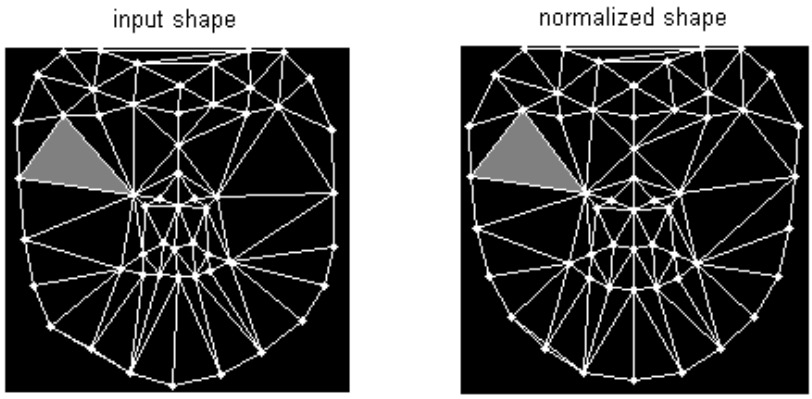

Figure 2.4: The Delaunay triangulation and affine warping. In the input image and the reference image, the landmarks are connected as triangular meshes via Delaunay triangulation algorithm, and each triangle is warped via affine transformation to its corresponding part in the reference face. The gray area shows an example of such corresponding pair of triangles.

experiments, we hand labeled these landmarks on the probe image $\mathbf{I}$.

If we can predict the shape vector $\hat{\mathbf{s}}_2$ and the appearance vector $\hat{\mathbf{t}}_2$ of the unknown image $\mathbf{J}$, by warping $\hat{\mathbf{t}}_2$ from the reference face under pose 2 back to the shape defined by $\hat{\mathbf{s}}_2$, we will be able to get the synthesized new image $\mathbf{J}$.

So the problem turns into: How to predict $\hat{\mathbf{s}}_2$, given $\hat{\mathbf{s}}_1$ and the training set $\{\mathbf{s}_i\}_{i=1}^{N_T}$? And how to predict $\hat{\mathbf{t}}_2$, given $\hat{\mathbf{t}}_1$ and the training set $\{\mathbf{t}_i\}_{i=1}^{N_T}$? They are the same mathematical problem. Using exactly the equation (2.5) that we described in Section 2.2 will predict the unknown shape $\hat{\mathbf{s}}_2$ and the unknown appearance $\hat{\mathbf{t}}_2$.

Figure 2.5: The 13 poses in CMU PIE database.

Then we can combine them to get the synthesized new image **J**, which is the new view of the probe face under pose 2 .

2.4 Experimental Results

2.4.1 Experiments on PIE database

We tested the performance of this method on the CMU PIE database [39]. The database contains 68 subjects. We chose 64 subjects as the training set, and 4 subjects (04016, 04022, 04026 and 04029) as the test set. Our experiments were performed on the "expression" subset including those images with neutral expressions, and those images containing glasses if the subject normally wears glasses. All images were converted to gray-scale images. The database contains 13 poses, illustrated in Fig. 2.5. We used combinations of 'c27' (frontal view), 'c37' (45° view) and 'c22' (profile) to test our algorithm. The landmarks were provided courtesy of Ralph Gross [17]. The number of landmarks vary depending on the pose, from 39 landmarks to 54 landmarks. So the number of shape variables, including both x and y coordinates and a pair of poses, is around 200. The sizes of the reference faces in each pose are around 180x180, so the number of appearance variables is around 60,000.

We performed 3 sets of experiments, including predicting frontal view from

profile (Fig. 2.6), predicting profile from frontal view (Fig. 2.7), predicting 45° view from frontal view (Fig. 2.8). These experiments all involve large out-of-plane rotations, such as 90° or 45°. In each experiment, the result of our approach is compared with that of linear-object-class method. We also computed the PCA reconstruction of the ground truth, by projecting the true $b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$ onto the eigenspace, to show the best possible reconstruction under the linear eigenspace assumption. In each experiment, for either the shape or the appearance, we always choose the number of principal eigenvectors that occupies 98% of energy. Although the number of shape variables is around 200 and the number of appearance variables is around 60,000, we only have 64 pairs of training images, so the maximum number of dimensions is 63. With 98% energy, the dimensions for shape and appearance are reduced to around 40 and 54 respectively.

Each synthesis takes an average of 4 - 5 seconds on a PC with a 3GHz Pentium 4 processor, including predicting shape and appearance and also warping the appearance to the shape. More specifically, the prediction of shape and appearance takes about 0.3 second, and the warping takes about 4 seconds.

Fig. 2.6 - 2.8 show how our approach improves upon the results of the linear-object-class method, especially in predicting the shapes and handling large out-of-plane rotations. Although our synthetic images are not perfect replicas of the ground truth, they are similar to the PCA reconstructions of the ground truth, which are the best possible synthetic images under the linear eigenspace assumption. We performed these experiments using a training set of only 64 subjects. With more training data, the eigenspace would be more accurately described and better results could be expected.

We have also included the numerical comparison of errors for shape prediction in each set of experiments (Table 2.1, 2.2, 2.3). In each table, we compare the square roots of the mean-squared-errors in the coordinates of the predicted shape, using the linear-object-class method, our approach, and the PCA reconstructions of ground truth, respectively. From the numerical errors, we can see our approach

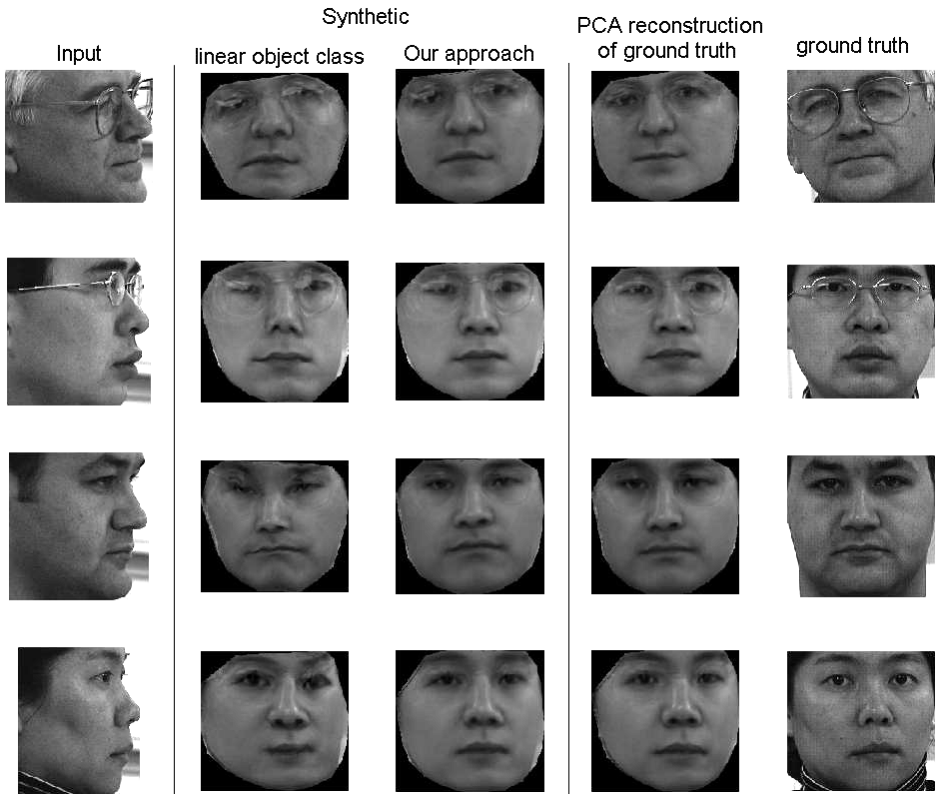

Figure 2.6: Synthesizing a frontal view from a given profile. Column 1 to 5: (1) input image under pose 1 (2) synthetic image using linear-object-class. (3) synthetic image using our approach. (4) PCA reconstruction of ground truth of pose 2. (5) ground truth of pose 2.

is efficient in reducing the errors by at least 30%.

2.4.2 Experiments on Multi-PIE and FERET databases

The above experiments have a training set of only 64 subjects. Such a small training set has limited capability to span a feature space to represent a number of possible human faces. So we did the experiments again with a much larger training set. We use data from two databases: Multi-PIE [18] and Color-FERET [32, 33],

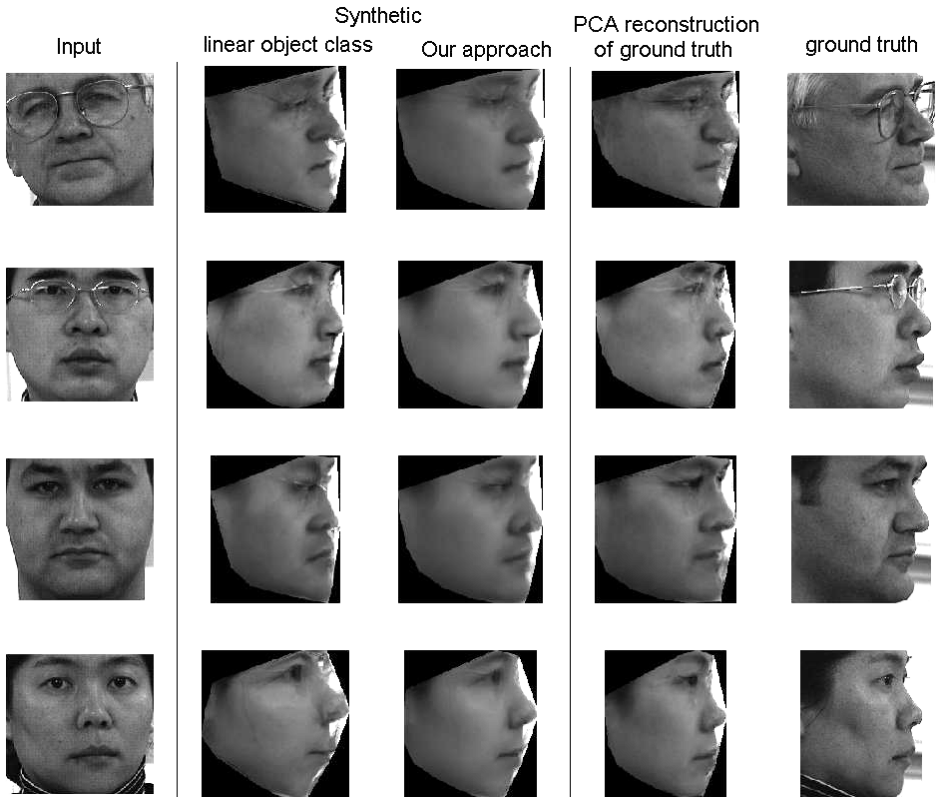

Figure 2.7: Synthesizing a profile from a given frontal view. Column 1 to 5: (1) input image under pose 1 (2) synthetic image using linear-object-class. (3) synthetic image using our approach. (4) PCA reconstruction of ground truth of pose 2. (5) ground truth of pose 2.

Table 2.1: Square roots of mean squared errors in pixel location for shape prediction from profile to frontal view

Linear-object-class method	Our approach	PCA reconstruction of ground truth
4.7714	3.5099	1.2343
3.0689	2.6584	1.1250
4.9929	3.2950	1.1216
7.2097	4.7792	1.2631

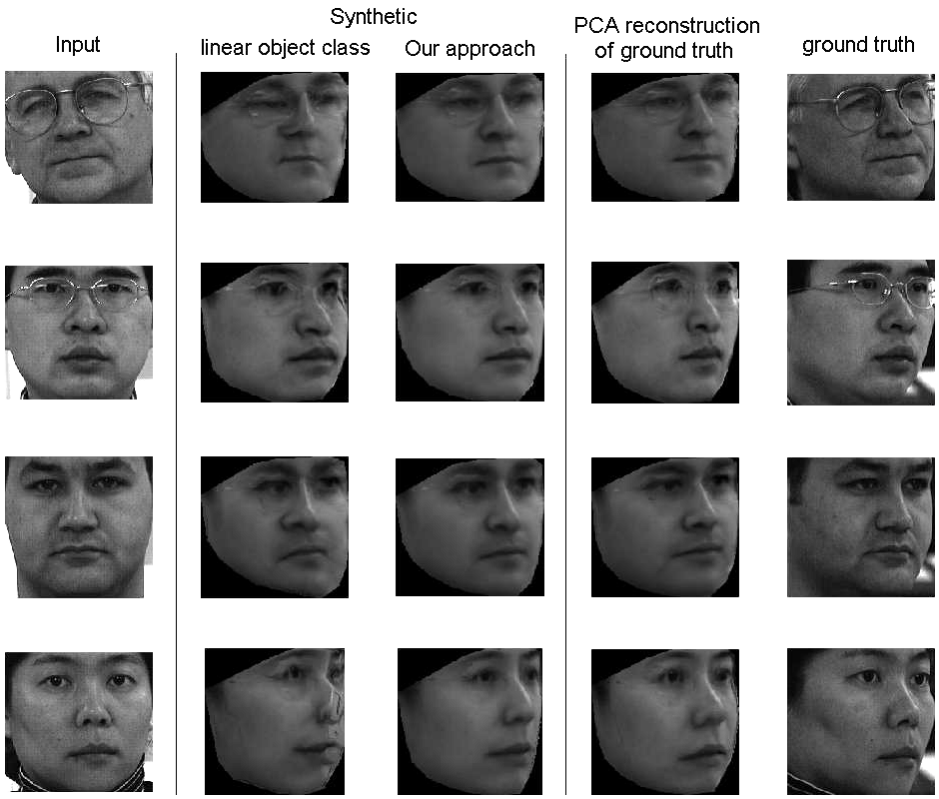

Figure 2.8: Synthesizing a 45° view from a given frontal view. Column 1 to 5: (1) input image under pose 1 (2) synthetic image using linear-object-class. (3) synthetic image using our approach. (4) PCA reconstruction of ground truth of pose 2. (5) ground truth of pose 2.

Table 2.2: Square roots of mean squared errors in pixel location for shape prediction from frontal view to profile

Linear-object-class method	Our approach	PCA reconstruction of ground truth
7.4199	4.2115	1.7578
4.7118	3.5389	1.3191
5.2369	3.4810	1.4949
5.5709	3.1199	1.7967

Table 2.3: Square roots of mean squared errors in pixel location for shape prediction from frontal view to 45° view

Linear-object-class method	Our approach	PCA reconstruction of ground truth
4.4719	3.3408	1.5467
4.1070	3.0246	1.1708
5.4397	3.3834	1.7336
6.2868	3.9477	1.4638

of frontal view (0°) and half profile (30°). We chose 1842 pairs of faces from 337 subjects in Multi-PIE database, and 1582 pairs of faces from 738 subjects in Color-FERET database, to form a training set of 3352 pairs of faces from 1075 subjects. We also chose 20 frontal images from 10 subjects Multi-PIE and 20 images from 15 subjects in Color-FERET, to form the test set. Our goal is to synthesize a half profile face from its frontal view.

We labeled 54 landmarks for each frontal view face and 47 landmarks for each half profile face. So the number of shape variables is 108+94=202. We chose the reference faces to be 100x100 size, so the number of appearance variables is 2x10000 = 20000. With 3352 pairs of training images, the maximum possible dimension of appearance is 3351. We also adjusted the pixel intensities of each warped image to be zero mean and unit variance. With eigen decompositions and keeping 98% energy, the dimensionality of shape variables reduces from 202 to 113, and the dimensionality of appearance variables reduces from 3351 to 1485.

In linear-object-class method, in order to make the set of equations as an over-constrained problem in shape prediction, we limit the number of eigenvectors to 1/3 of the number of known shape variables, i.e., 1/3 x 108 = 36. Without such limitation, had we kept 98% energy which are 113 dimensions, the equations would have been an underconstrained problem and the result would have been unreliable. For appearance prediction, we still keep 98% energy because 1485 dimensions will still make the equations as an over-constrained problem.

Below (figure 2.9) we list some results of the two approaches for comparison,

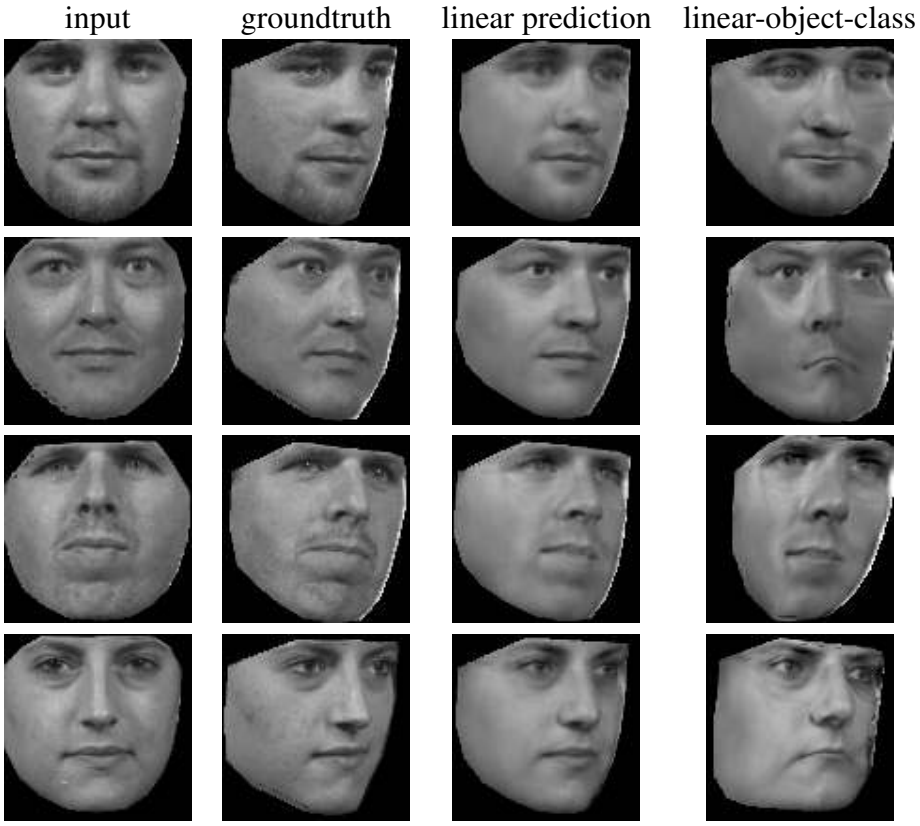

Figure 2.9: Comparison of prediction results of linear method and linear-object-class method on Multi-PIE and FERET database.

where it shows the linear method outperforms the linear-object-class method. Especially the linear-object-class method has severe distortion in shape prediction. The average square roots of mean-squared-error in shape prediction for the 40 test images is 1.4536, which is pretty small compared to the results of PIE databases. This shows with a large training set, the shape prediction can be quite accurate and satisfactory. And the average square roots of mean-squared-error in appearance prediction for the 40 test images is 13.1247, which means approximately the error in predicting the intensity of each pixel.

Although the linear method outperformed the linear-object-class method, it still has some failure examples (figure 2.10). It seems the linear method has some

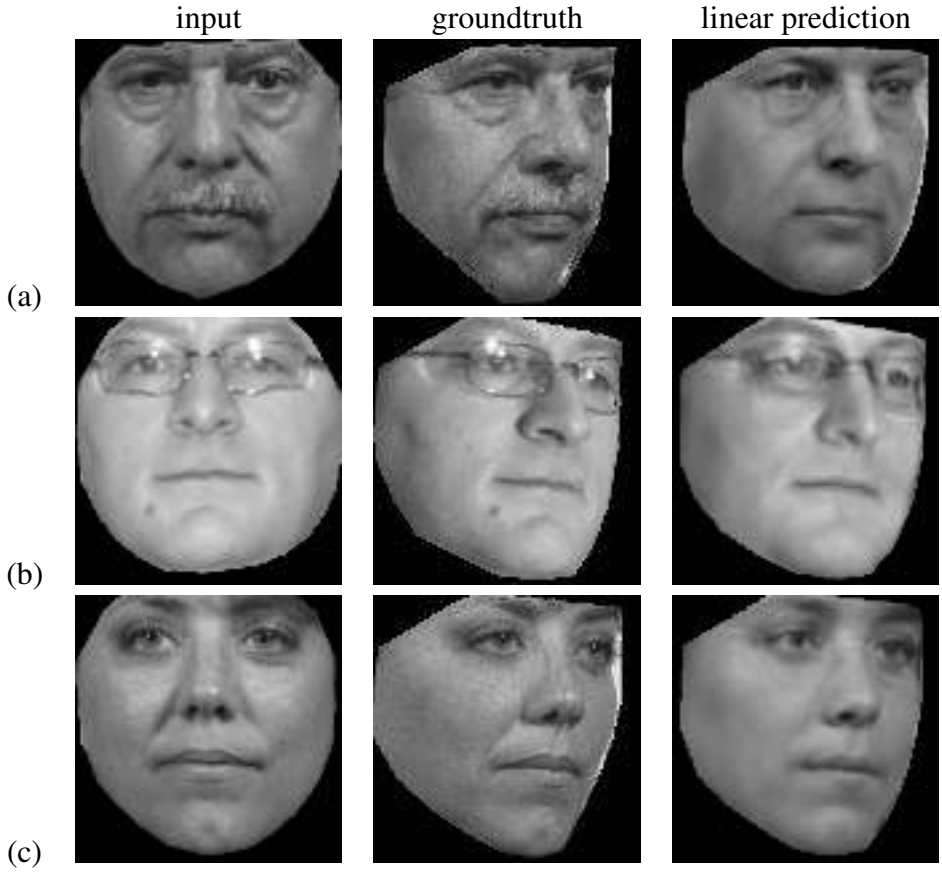

Figure 2.10: Examples of failure in the linear method. In each example, the left image is the ground truth and the right image is the synthetic image of this pose. (a) The subject's mustache is removed in the synthetic image. (b) The subject's mole is removed in the synthetic image. (c) The subject's eyes are surrounded by a partially hallucinated pair of glasses in the synthetic image.

problems in predicting small and localized features, such as mustache or moles, or determining whether the subject wear glasses.

2.5 Conclusions

In this chapter, we proposed an approach that can efficiently synthesize accurate new views of faces across large out-of-plane rotation, given only a single image. In our approach, we formulate a probabilistic model combining the "distance-from-feature-space" and the "distance-in-feature-space", and minimize the weighted sum of the two distances, in order to maximize the likelihood of the test example with missing data. Experimental results show that our approach produces more accurate results than the commonly used linear-object-class approach which is the basis of many 2D approaches. However, this approach is not always reliable at predicting localized features as shown in Figure 2.10. In the next chapter, we explore a method designed to improve localized prediction.

Chapter 3

Bayesian Network Approach

3.1 Motivation

One of the challenges of image synthesis is high dimensionality; that is, it is a prediction problem over many variables. In the last chapter, we manually separated shape and appearance, and analyzed the variables in the appearance holistically, as well as the variables in the shape; that is, that method does not decompose the variables any further. However, such a representation may be ill-suited to this problem. Examples of failure are shown in figure 2.10, where the subject's distinctive characteristics such as mustache or moles are removed in the synthetic image, or the subject's eyes are surrounded by artifacts of glasses in the synthetic image while the subject actually does not wear glasses.

The holistic approach is ill-suited to this problem because sparse statistical structure usually exists among the random variables [36, 37, 20]. Each random variable usually has strong statistical dependency with a few variables, and has weak dependency with the others (Figure 3.1). If we ignore such statistical structure and group all the variables together, the high dimensionality of the model will be susceptible to over-fitting. Such a holistic representation also models

relationships among variables that have no statistical dependency and is, therefore, wasteful. Instead, we propose to take advantage of the statistical structure which will allow us to formulate our solution as a combination of smaller models and thereby be less susceptible to over-fitting and to devote more representational power to true dependencies among the variables.

We aim to build models that exploit this sparse statistical structure among the random variables. Graphical models, such as Bayesian networks, are well matched to this task. In such models, each node represents a single variable. Connections between nodes exist only if there are direct statistical dependencies. Variables are unconnected if they are statistical independent or conditionally independent. Such a graphical structure concisely captures sparseness. In doing so, it greatly reduces the dimensionality of the problem. Instead of a single high-dimensional model, a graph consists of a collections of lower-dimensional models at each node. The dimensionality of each such model is given by the number of edges flowing into the node. Such a representation makes good use of the training data we have and reduces over-fitting.

In particular, a Bayesian network represents a probability distribution over a group of variables $X_1, \ldots, X_n$ as:

$$p(\Omega) = \prod_{i=1}^{n} p(x_i \mid Pa_{x_i}) \tag{3.1}$$

The probability representation in each lower dimensional model is $p(x_i \mid Pa_{x_i})$, representing the probability of variable X_i conditioned on all the variables flowing into it on the graph, Pa_{X_i}.

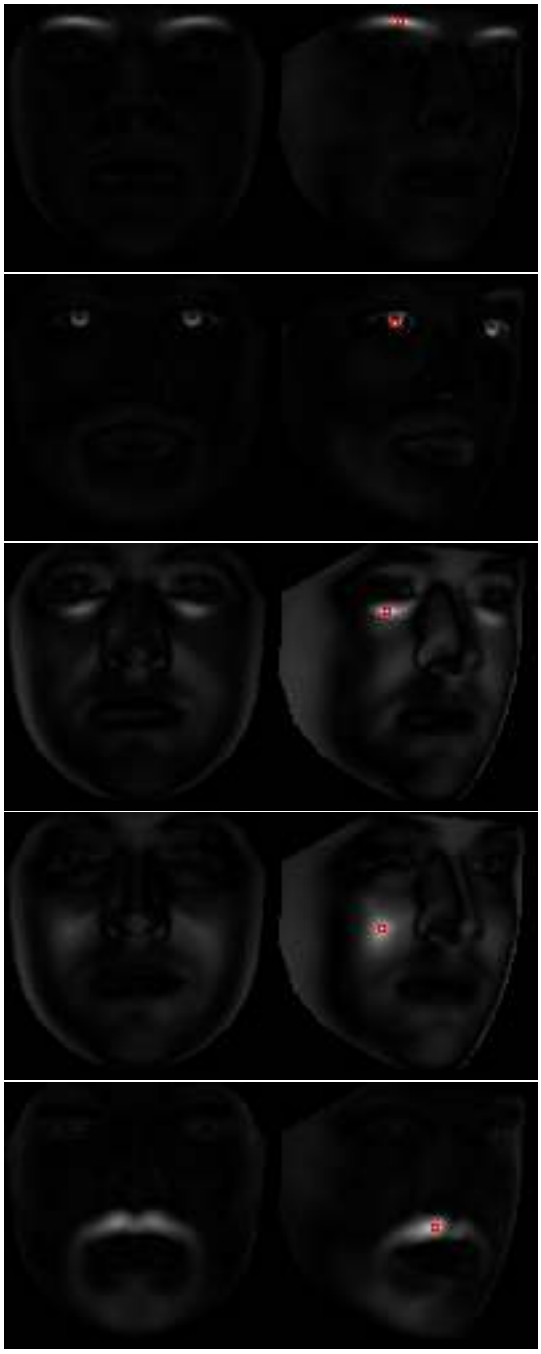

Figure 3.1: Mutual information between pixels on pairs of human faces. Each red rectangle denotes the pixel we are analyzing, and the brightness of other pixels show the relative magnitude of mutual information with that pixel visually. The five pictures show areas of eyebrows, eyes, eye glasses, cheekbones and mustache respectively.

3.2 Overview

3.2.1 Three Steps

The Bayesian network approach for prediction contains three steps.

- The first step is to learn the structure of the Bayesian network, which means the connectivity among the nodes in the graph. With the network structure learned, it can represent the joint probability distribution using factorization (Eq. 3.1).
- The second step estimates the parameters in the representation of the probability distribution functions, for example, the mean and covariance if the distribution were represented as a Gaussian. With the structure and parameters learned, the Bayesian network is fully known.
- Then in the third step, given the known variables, we predict the unknown variables by maximizing the joint probability. That means, we can predict the random variables in pose 2, denoted as $\mathbf{b}_2$, from the random variables in pose 1, denoted as $\mathbf{b}_1$, by maximizing the probability $p(\mathbf{b}_2 \mid \mathbf{b}_1)$, using standard inference techniques in the graphical models.

3.2.2 Approaches

Learning the structure of the Bayesian network is difficult when the number of variables is large, because the number of all possible structures is exponential to the number of variables. There are two general approaches: score-based approaches and constraint-based approaches. The score-based approach uses a score function to evaluate many possible structures selecting the structure with the largest score, and is efficient when the score function is decomposable. Since searching for the optimal structure is NP-hard, it applies heuristics in the search, and is susceptible to local optima. The constraint-based approach is very intuitive in that it

does conditional independence test (CI test) to every pair of nodes to determine whether the pair should be connected with an edge. However, a CI test with a large size condition set may not be reliable because it may cause overfitting in the data when the size of training set is not large enough to estimate the probability distribution. In this research, we decided to use a constraint-based method because it is intuitive and easy to implement.

In the constraint-based approaches, we need to choose some measurement for the conditional dependency between nodes. χ^2 and conditional mutual information are both good measurements. We choose conditional mutual information. The conditional mutual information of X and Y given U is defined as:

$$I\left(X;Y \mid C\right) = \sum_{c} P(c) \sum_{x} \sum_{y} P\left(x,y \mid c\right) log \frac{P\left(x,y \mid c\right)}{P\left(x \mid c\right) P\left(y \mid c\right)}$$

3.2.3 Challenges

There are many challenges of using constraint-based approach of Bayesian network structure inference for face view synthesis problem.

First of all, the number of variables is huge. With 108 and 94 shape variables in the two poses, and 100x100 pixels in the appearance variables in each pose, we have a total of 20,202 variables. Many constraint-based approaches require the number of CI test being exponential to the number of variables, which makes the computation very expensive.

Also in each CI test, the size of the condition set might be very large if the variable has many possible parents. CI tests over large condition sets are unreliable because of overfitting.

Another challenge is that the mutual information across pose is usually smaller than the mutual information within pose, which makes it hard to compare them directly.

3.3 Constraint-Based Search for Bayesian Network Structure

3.3.1 Overview

The goal of structure learning is to achieve a perfect map (P-map). A graph G is a P-map of a probabilistic distribution P if every independence relationship in G is true in P and every dependence relationship in G is true in P.

There are several constraint-based approaches that find the P-map of a distribution. These include the SGS algorithm [40], PC algorithm [40], Cheng's TPDA and TPDA-Π algorithms [8], RAI algorithm [47], etc. Computational complexity vary among these. SGS algorithm does an exhaustive set of CI tests which is exponential in the number of variables. PC algorithm has complexity of $O\left(\frac{N^2(N-1)^{k-1}}{(k-1)!}\right)$, where N is the number of variables and k is the maximum in-degree of the graph. Cheng's TPDA algorithm has complexity of only $O\left(N^4\right)$ when there is no prior knowledge of the node ordering, and TPDA-Π algorithm has complexity of only $O\left(N^2\right)$ when the ordering is given. The RAI algorithm combines edge removal and edge orientation iteratively, with the graph being divided into autonomous subgraphs. It assumes no prior knowledge of node ordering and experimental results show the number of CI tests required is less than PC algorithm.

In terms of minimizing the number of CI tests and lowering computational complexity, it seems that TPDA-Π algorithm is the best choice, in that it only requires $O\left(N^2\right)$ CI tests. However, as we have mentioned, CI tests with large condition sets are unreliable. Each CI test in TPDA-Π algorithm uses all the parents of a node as the condition set, which could be very large a set and give unreliable results of conditional independence. On the other hand, PC algorithm controls the size of condition set in each loop, to increase from zero to some value until no more edges are eliminated. In this sense, we can modify the PC algorithm

to set an upperbound of the size of condition set, in order to get reliable results of CI tests. The modification also includes adding prior ordering information, to eliminate the step of orienting the edges by some rules.

3.3.2 Prior Node Ordering

In our problem, the variables are grouped as "shape1, appearance1, shape2, appearance2". "shape1" and "appearance1" are from pose 1, which are known variables. "shape2" and "appearance2" are from pose 2, which are unknown variables. It is reasonable to assume variables from pose 1 appear earlier in the ordering than variables from pose 2, because this is how the inference works. Moreover, it is also reasonable to assume that in each pose, shape variables appear earlier than appearance variables. Within each part, the natural ordering from the first variable to the last variable can serve as an ordering. So we assume the ordering of nodes is known as "shape1$\rightarrow$ appearance1 $\rightarrow$ shape2 $\rightarrow$ appearance2".

3.3.3 Modified PC Algorithm with Node Ordering

The original PC algorithm [40] assumes no prior knowledge of node ordering. The algorithm starts from a fully-connected graph over the nodes. Then with the degree n starting from 0 and increasing by 1 at a time, for each connected pair of nodes (X, Y) such that Y has more than n parents, try all the subsets S of cardinality n among its parents excluding X, and perform CI tests $X, Y|S$. If the CI test succeeds, remove the edge $Y - X$ and record the witness set S. The degree n is increased until there are no nodes with at least $n + 1$ parents. Then, in this undirected graph, some rules are applied to orient the edges in the graph.

There are some heuristics in selecting the order of the tests in speeding up the algorithm. One of them is that, for a given variable A, first test those variables B that are least probabilistically dependent on A, conditioned on those subsets of

variables that are most probabilistically dependent on A. We are interested in this heuristic and will apply it in the following paragraph.

If the node ordering is given, it is straightforward to modify the PC algorithm as follows.

1. The algorithm starts from a fully-connected graph over the nodes. The directions of edges are defined as pointing from higher order to lower order.

2. With the degree $n = 0$, eliminate all the edges with mutual information less than a threshold ε_1.

3. With the degree $n = 1$, for each node X, sort its parents with mutual information in descending order $\{Y_k\}$ such that $I(X, Y_k) > I(X, Y_{k+1})$. First try the parent Y_1 with the largest mutual information $I(X, Y_1)$, to perform CI tests on the rest of parents Y_k, i.e., $X, Y_k|Y_1$. If the CI test succeeds, which means $I(X, Y_k|Y_1) > \varepsilon_2$ where ε_2 is a threshold, remove the edge $Y_k \rightarrow X$. Then try the parent Y_2 with the second largest mutual information as condition set and perform CI tests on the rest of parents, and so forth.

4. The degree n can be increased to a certain upperbound.

In our experiments, we choose the upperbound to be 1, which means we always try CI tests with condition set only containing one variable. The complexity of this modified algorithm is $O(N^3)$, where N is the number of variables. So it has a reasonable complexity and reliable results of CI tests. Although the upperbound may make the final graph only an I-map instead of a P-map, an I-map is still good for factorization and is a great reduction in dimensionality for a holistic representation.

3.3.4 Dependency Measures

The mutual information and conditional mutual information are used as the measure of dependency. As mentioned in Section 3.2.2, we use conditional mutual information as a dependency measure. Conditional mutual information is a generalization of mutual information which is defined as

$$I(X;Y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \,, \tag{3.2}$$

where $I(X;Y)$ is the mutual information, and X, Y are two random variables. Note that upper cases denote random variables and lower cases denote particular instantiations of the random variables. Mutual information is a measure of how much information can be obtained about one random variable by observing another [11]. The larger the mutual information is, the stronger dependency exists between X and Y.

Both the mutual information and the conditional mutual information need the estimation of joint probabilities, such as $p(x,y)$, $p(x,y,c)$, $p(x,c)$ and $p(y,c)$. Since x, y, c each have many possible values, the combination of them would be very large a table and the number of training set will be too small to estimate such a large table. So we did scalar quantization for the variables. Each variable is quantized to 4 levels so the maximum size of the probability table will be $4^3 = 64$, which makes the training set (3352 examples in our experiments) sufficient.

In order to make dynamic range of each $I(X,Y)$ comparable, we quantize each variable to a uniform distribution because a uniform distribution has the largest entropy. So the quantization of each variable is done such that each of the 4 bins has the same amount of examples. This is because $I(X,Y) = H(X) + H(Y) - H(X,Y)$, where $I(X,Y)$ is the mutual information, $H(X)$ and $H(Y)$ are the entropy, $H(X,Y)$ are the joint entropy. When X and Y are deterministic to each other, $I(X,Y) = H(X) = H(Y)$. When X and Y are independent, $I(X,Y) = 0$. And note that the joint entropy $H(X,Y) \geq H(X)$. So the dynamic

range of $I(X,Y)$ is from 0 to $max\,(H(X), H(Y))$. A uniform distribution will make the upperbound $max\,(H(X), H(Y))$ the same.

3.3.5 Thresholds

As we described in Section 3.3.3, there are two thresholds, ε_1 and ε_2. ε_1 is the threshold for mutual information, and ε_2 is the threshold for conditional mutual information. In practice, we found the dynamic range of conditional mutual information is quite different from mutual information. That is why we choose ε_1 and ε_2 to be different. In mutual information, we choose $\varepsilon_1 = 0.5 \times max(MI)$, where $max(MI)$ is the maximum mutual information for each variable with all its parents. In conditional mutual information, we choose $\varepsilon_2 = 0.03$.

3.4 Probability Representation

Suppose the Bayesian network has a total of m nodes $[x_1, x_2, \cdots, x_m]$. Our assumption is that these m nodes have a joint Gaussian distribution, i.e., $[x_1, x_2, \cdots, x_m] \sim N\,(\mu,\, \Sigma)$. As we know, if a vector is jointly Gaussian distributed, then the conditional distribution within this vector is also Gaussian. More specifically, if μ and Σ are partitioned as follows

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{ with sizes } \begin{bmatrix} q \times 1 \\ (N-q) \times 1 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \text{ with sizes } \begin{bmatrix} q \times q & q \times (N-q) \\ (N-q) \times q & (N-q) \times (N-q) \end{bmatrix}$$

then the distribution of x_1 conditional on $x_2 = a$ is multivariate normal $X_1 \mid X_2 = a \sim N\left(\bar{\mu},\, \bar{\Sigma}\right)$ where

$$\bar{\mu} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}\,(a - \mu_2)$$

and covariance matrix

$$\bar{\Sigma} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

For node y_j, its parents are denoted as $Pa(y) = [x_{j1}, x_{j2}, \cdots, x_{jk}]'$. Under Gaussian assumption, the conditional probability distribution is $y_j \mid x_{j1}, x_{j2}, \cdots, x_{jk} \sim N\left(\bar{\mu}_j, \bar{\Sigma}_j\right)$, where

$$\bar{\mu}_j = \mu_j + \Sigma_{j,\,(j1,\cdots,jk)}\Sigma^{-1}_{(j1,\cdots,jk),\,(j1,\cdots,jk)}\left(\begin{pmatrix} x_{j1} \\ \vdots \\ x_{jk} \end{pmatrix} - \begin{pmatrix} \mu_{j1} \\ \vdots \\ \mu_{jk} \end{pmatrix}\right)$$

$$\bar{\Sigma}_j = \Sigma_{j,\,j} - \Sigma_{j,\,(j1,\cdots,jk)}\Sigma^{-1}_{(j1,\cdots,jk),\,(j1,\cdots,jk)}\Sigma_{(j1,\cdots,jk),\,j}$$

So if we estimate the overall μ and Σ from the training data, we will be able to compute the parameters $\bar{\mu}_j$ and $\bar{\Sigma}_j$ of the conditional distribution at each node y_j.

The parameters can also be rewritten in another form $y_j \sim N\left(\hat{\mu_j} + \hat{W_j}' \cdot Pa(y),\, \hat{\sigma_j}\right)$, where

$$\hat{\mu_j} = \mu_j - \Sigma_{j,\,(j1,\cdots,jk)}\Sigma^{-1}_{(j1,\cdots,jk),\,(j1,\cdots,jk)}\begin{pmatrix} \mu_{j1} \\ \vdots \\ \mu_{jk} \end{pmatrix}$$

$$\hat{W_j}' = \Sigma_{j,\,(j1,\cdots,jk)}\Sigma^{-1}_{(j1,\cdots,jk),\,(j1,\cdots,jk)}$$

$$\hat{\sigma_j} = \bar{\Sigma}_j$$

$\hat{\mu_j}$ and $\hat{\sigma_j}$ are scalars, while $\hat{W_j} = [w_1, w_2, \cdots, w_k]'$ is a vector, meaning the weights for the k parents of node y.

3.5 Parameter Estimation

Once we have learned the structure of the Bayesian network, we need to estimate the parameters of the probability distributions in the Bayesian network. We assume the variables are adequately described by a Gaussian distribution. Under this assumption, given training data, a closed form solution to the probability distribution's parameters exists. Suppose the Bayesian network has a total of m nodes. For node y, its parents are denoted as $Pa(y) = [x_1, x_2, \cdots, x_k]'$. Under Gaussian assumption, the conditional probability distribution at this node is $y \sim N(\mu + W' \cdot Pa(y),\ \sigma)$, where μ and σ are scalars, while $W = [w_1, w_2, \cdots, w_k]'$ is a vector, meaning the weights for the k parents of node y.

The number of training examples is N. We need to first estimate the parameters using the training examples, and then apply these parameters and the known part of test example, to predict the unknown part of the test example, as a closed-form solution.

We have N training examples for node y, where $y \sim N(\mu + W' \cdot Pa(y),\ \sigma)$. If we write the total probability as the product of each probability and maximize it, we will have

$$\begin{aligned} P(y) &= \prod_{i=1}^{N} P(y_i \mid Pa(y_j)) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(y_i - \mu - W' \cdot Pa(y_i))^2}{2\sigma^2}\right) \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^N exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \mu - W' \cdot Pa(y_i))^2\right) \quad (i = 1, 2, \cdots, N) \end{aligned}$$

So maximizing $P(y)$ means minimizing $\sum_{i=1}^{N}(y_i - \mu - W' \cdot Pa(y_i))^2$, which is exactly the least-square solution for this set of N equations,

$$\mu + W' \cdot Pa(y_i) = y_i \quad (i = 1, 2, \cdots, N)$$

Solving this set of overconstrained linear equations, we will get μ and W in least-

square solutions. Letting the partial derivative of σ be zero, we have

$$\sigma^2 = \frac{\sum_{i=1}^{N} \left(y_i - \mu - W' \cdot Pa\left(y_i\right)\right)^2}{N}$$

3.6 Prediction

Given a test example, we know a subset of the nodes, and need to predict the remaining unknown nodes. A closed form solution to this prediction problem exists as follows. If we write the total probability as the product of conditional probability at each node and maximize it, we will have

$$\begin{aligned} P\left(BayesNet\right) &= \prod_{j=1}^{m} P\left(y_j \mid Pa\left(y_j\right)\right) \\ &= \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi}\sigma_j} exp\left(-\frac{\left(y_j - \mu_j - W_j' \cdot Pa\left(y_j\right)\right)^2}{2\sigma_j^2}\right) \\ &= \frac{1}{\left(\sqrt{2\pi}\right)^m \prod_{j=1}^{m} \sigma_j} exp\left(-\frac{1}{2}\sum_{j=1}^{m} \frac{\left(y_j - \mu_j - W_j' \cdot Pa\left(y_j\right)\right)^2}{\sigma_j^2}\right) \end{aligned}$$

where the total number of nodes is m.

In order to maximize the total probability, we need to minimize $\sum_{j=1}^{m} \frac{\left(y_j - \mu_j - W_j' \cdot Pa(y_j)\right)^2}{\sigma_j^2}$, which is exactly the linear least-square solution for this set of N equations,

$$\frac{\mu_j + W_j' \cdot Pa\left(y_j\right)}{\sigma_j} = \frac{y_j}{\sigma_j} \quad (j = 1, 2, \cdots, m)$$

Some of the nodes are known while some are unknown. The number of unknown nodes is also $m/2$. And the number of equations containing unknown nodes is generally greater than $m/2$. So generally it is an overconstrained problem and has a closed form least square solution. However, according to the ordering

we choose where nodes of pose1 all appear earlier than nodes of pose2, the number of equations containing unknown nodes is exactly $m/2$. The coefficients in the above equations will be in a lower triangle form. Solving this set of equations will give the closed-form exact solution for the unknown nodes as the prediction results. Each equation is exactly satisfied, with zero tolerance, and may be sensitive to errors. This means if any two unknown nodes are involved in the same equation, the prediction error may propagate and enlarge. Therefore, when using such ordering, we should try to avoid connections within the same pose, so that each unknown node has parents that are all known, and the prediction errors of all unknown node are independent.

3.6.1 Across Pose Connections v.s. Within Pose Connections

As we will explain in Section 3.6, with the ordering as "shape1 $\rightarrow$ appearance1 $\rightarrow$ shape2 $\rightarrow$ appearance2", all the known variables come before all unknown variables in the node ordering. So each known variable has all the parents being known variables, and such within-pose connections will not help the prediction at all. In the prediction, only the equations involving unknown variables will count. So the number of such equations will be exactly the same as the number of unknown variables, which leads to a square matrix equations, and has a unique solution. This means the solution will make each equation exactly satisfied, with zero error tolerance. If more than two unknown variables appear in one equation, the prediction error may propagate among the unknown variables and the results may be out of control. So we force all connections be across-pose connections, such that the prediction of unknown variables will not interfere each other.

3.7 Experimental Results

3.7.1 Datasets

We use data from two databases: Multi-PIE and Color-FERET, of frontal view (0°) and half profile (30°). We chose 1842 pairs of faces from 337 subjects in Multi-PIE database, and 1582 pairs of faces from 738 subjects in Color-FERET database, to form a training set of 3352 pairs of faces from 1075 subjects. We also chose 20 frontal images from 10 subjects Multi-PIE and 20 images from 15 subjects in Color-FERET, to form the test set. Our goal is to synthesize a half profile face from its frontal view.

We labeled 54 landmarks for each frontal view face and 47 landmarks for each half profile face. So the number of shape variables is 108+94=202. We chose the reference faces to be 100x100 size, so the number of appearance variables is 2x10000 = 20000. We also adjusted the pixel intensities of each warped image to be zero mean and unit variance.

3.7.2 Bayes Net Structure Results

In the Bayesian network structure that we obtained, each node has on average 15.5123 parents. The maximum number of parents is 152.

Figure 3.2 shows various learned parent-child relationships in the network. The corresponding examples of mutual information are shown in Figure 3.1. We see some interesting behavior.

- We see that the appearance of an eye in one pose is directly dependent on both eyes in the other pose.
- We see similar spatial and symmetric correspondence for the area just below the eye, for the cheek-bone and for the mustache area. However, in each of

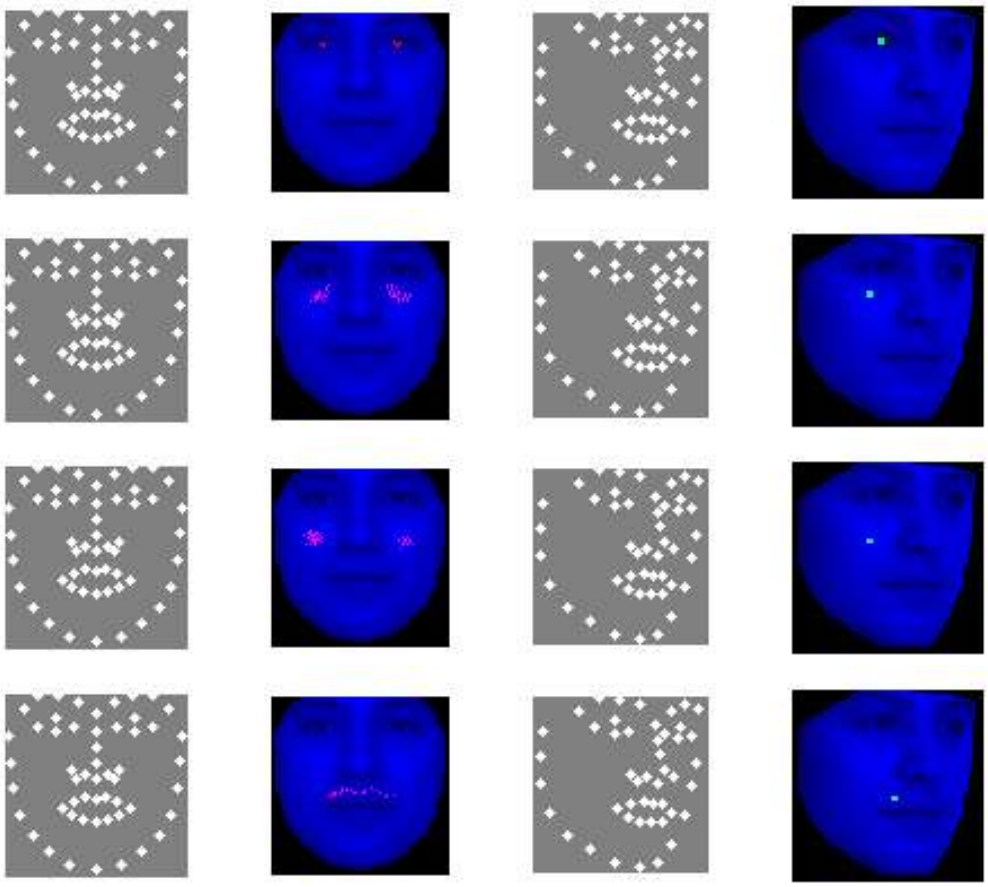

Figure 3.2: These examples show various parent-child relationships in the Bayesian network, in the areas of eyes, eyeglasses, cheekbones, mustache respectively. In each figure, the green dot is the variable that we are analyzing. The red dots are its parents.

these, the area of dependency is of larger spatial extent than for the eye.

- This Bayesian network models certain symmetry.

We also see that the shape variables often depend upon other shape variables and appearance variables. Figure 3.3 shows an example of the mutual information and Bayesian network connection between between shape and appearance. The

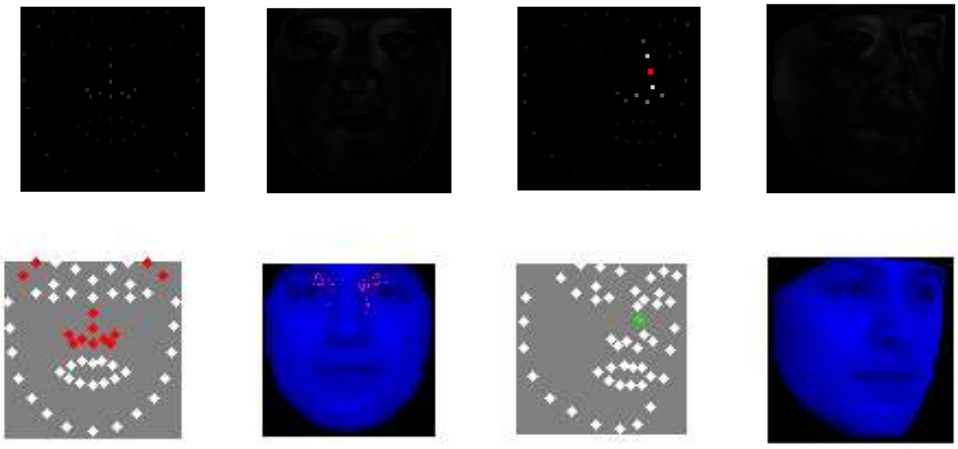

Figure 3.3: This example shows the dependency between shape and appearance. The four pictures in each row are shape 1, appearance 1, shape 2, appearance 2, respectively. The first row shows the mutual information between shape and appearance. The red rectangle denotes the shape variable we are analyzing, and the brightness of other pixels show the relative magnitude of mutual information with that shape variable visually. The second row shows the parent-child relationships in the Bayesian network between shape and appearance. The green dot is the shape variable that we are analyzing. The red dots are its parents.

shape variable we are analyzing in this example is a point on the nose in shape 2. Since it is in the half profile of the face, the location of this nose variable shows how protruding the nose is. That is why it has dependency with the intensity of the pixels beside the nose in appearance 1. The darker those pixels are, the more protruding the nose is.

For the occluded pixels, such as the side of the nose, there is no direct correspondence in the frontal view. Figure 3.4 shows an example of the mutual information and Bayesian network connection of this case. Since there is no direct correspondence, the overall mutual information is very low. The Bayesian network chooses a pixel that has relatively higher mutual information as the parent.

One interesting thought is whether we can use the mutual information to do

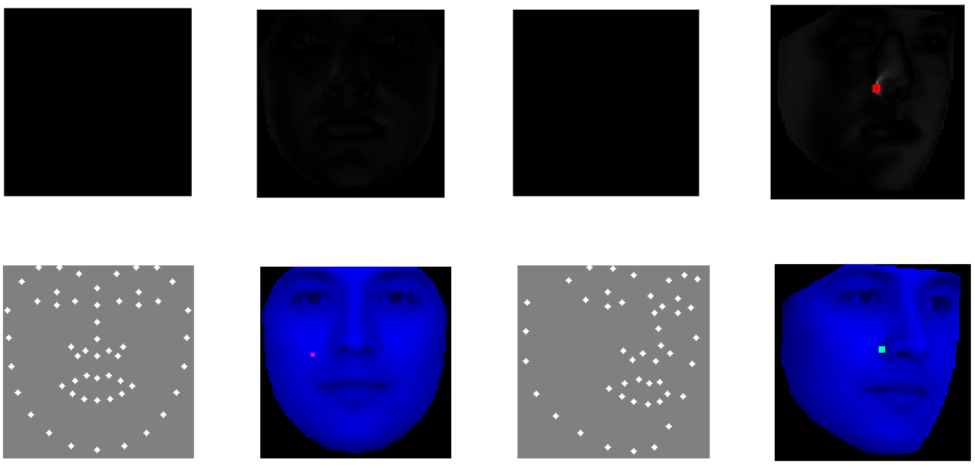

Figure 3.4: This example shows which pixel to predict the occluded nose pixel. The four pictures in each row are shape 1, appearance 1, shape 2, appearance 2, respectively. The side of the nose in the half profile is actually occluded in the frontal view. The first row shows this row shows the mutual information. The red rectangle denotes the nose pixel we are analyzing, and the brightness of other pixels show the relative magnitude of mutual information with that nose pixel visually. We can see the overall mutual information is very low in appearance 1, since there is no direct correspondence with the occluded nose pixel. The second row shows the parent-child relationships in the Bayesian network for the occluded nose pixel. The green dot is the nose pixel that we are analyzing. The red dot is its parent, which is not a direct correspondence, but a pixel that has relatively higher mutual information.

segmentation. We performed normalized cuts algorithm [38] with mutual information as the similarity matrix on each pose. Figure 3.5 and 3.6 show the segmentation results, with the number of segments varying from 2 through 10.

3.7.3 Prediction Results

Here are some results comparing the appearance prediction of the Bayesian network method with the linear method visually (figure 3.7, 3.8, 3.9). All results are

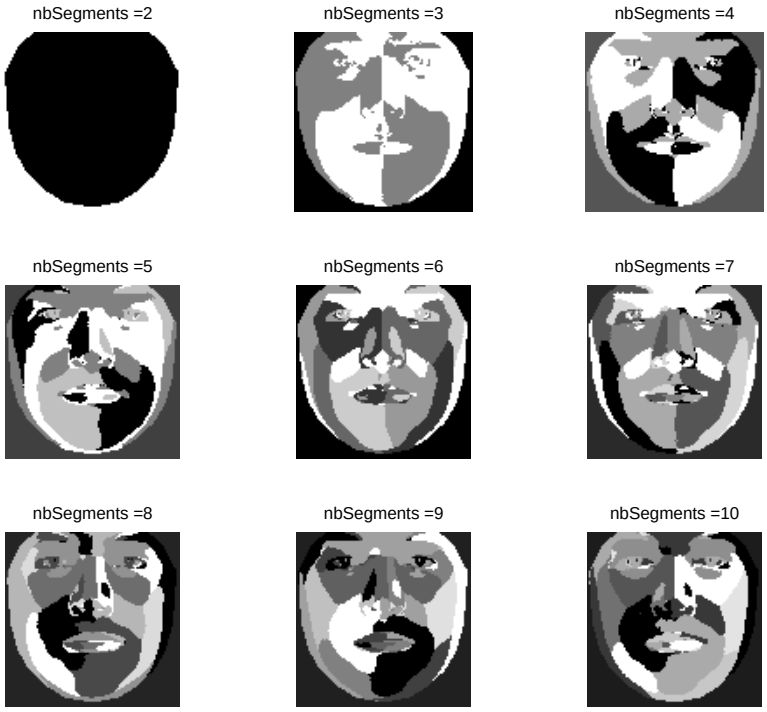

Figure 3.5: Segmentation using normalized cuts algorithm with mutual information as the similarity matrix, on frontal view of the face. nbSegments is the number of segments, which varies from 2 through 10. Different gray areas denote different segments.

warped to groundtruth shape so that the results of appearance prediction are easier to compare.

3.7.4 Numerical Results

The Bayesian network also gives better overall quantitative performance than the linear method (table 3.1). In comparing mean-squared error, the Bayesian network's average error in appearance prediction for the 40 test images is 167.8496,

Figure 3.6: Segmentation using normalized cuts algorithm with mutual information as the similarity matrix, on half profile view of the face. nbSegments is the number of segments, which varies from 2 through 10. Different gray areas denote different segments.

while in the linear method this average error is 178.5642. So the average improvement in appearance error among the 40 test images is 10.7147 ± 9.6887, with a margin of error for 90% confidence level.

3.7.5 Experiments of Predicting Frontal from Half Profile

We also performed the experiments of reverse predicting, i.e., predicting frontal from half profile. In figure 3.10, the first three rows show the linear method some-

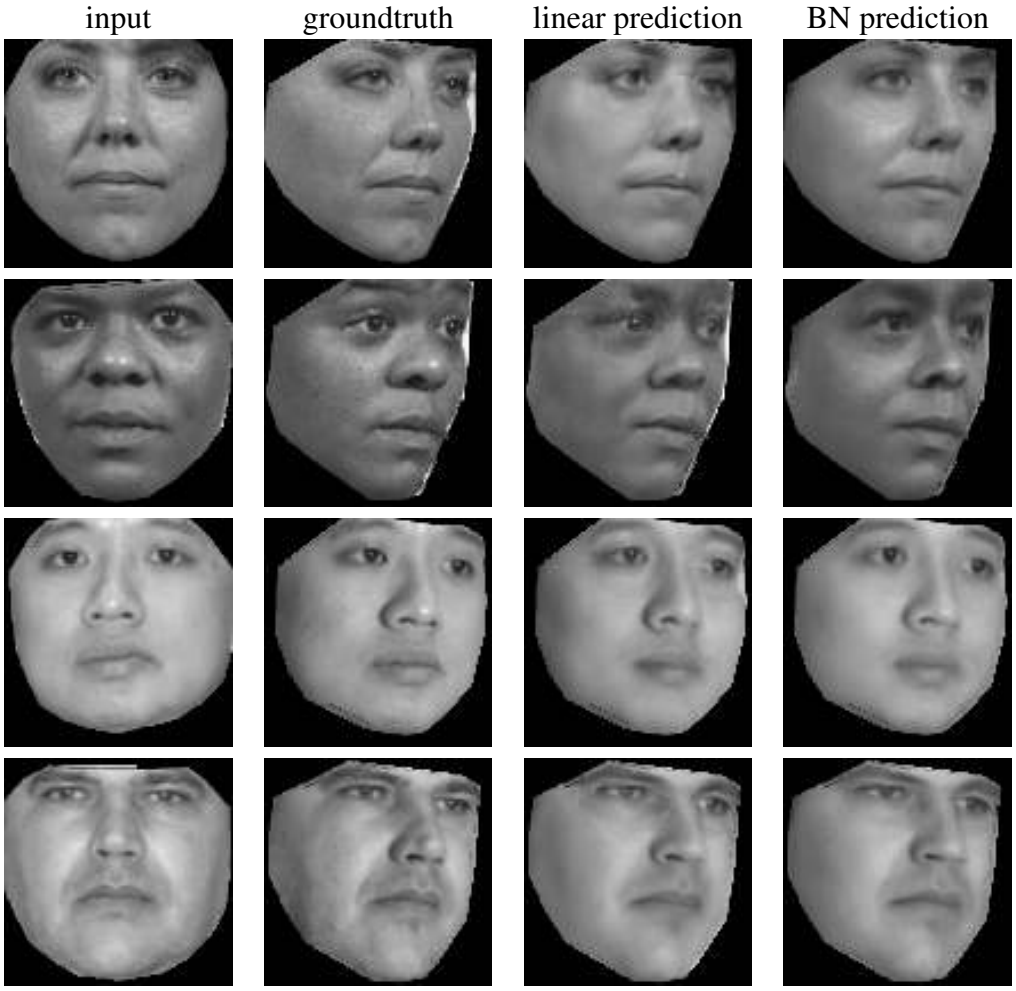

Figure 3.7: Results comparing the linear method and the Bayesian network method. The four examples show the linear method predicts a partially hallucinated pair of glasses around the subject's eyes while the Bayesian network approach does not have such artifacts.

	MSE in Appearance, predicting half profile
Linear Method	178.5642
BN method	167.8496

Table 3.1:
Comparison of mean-squared error over an independent test set, in predicting half profile from frontal.

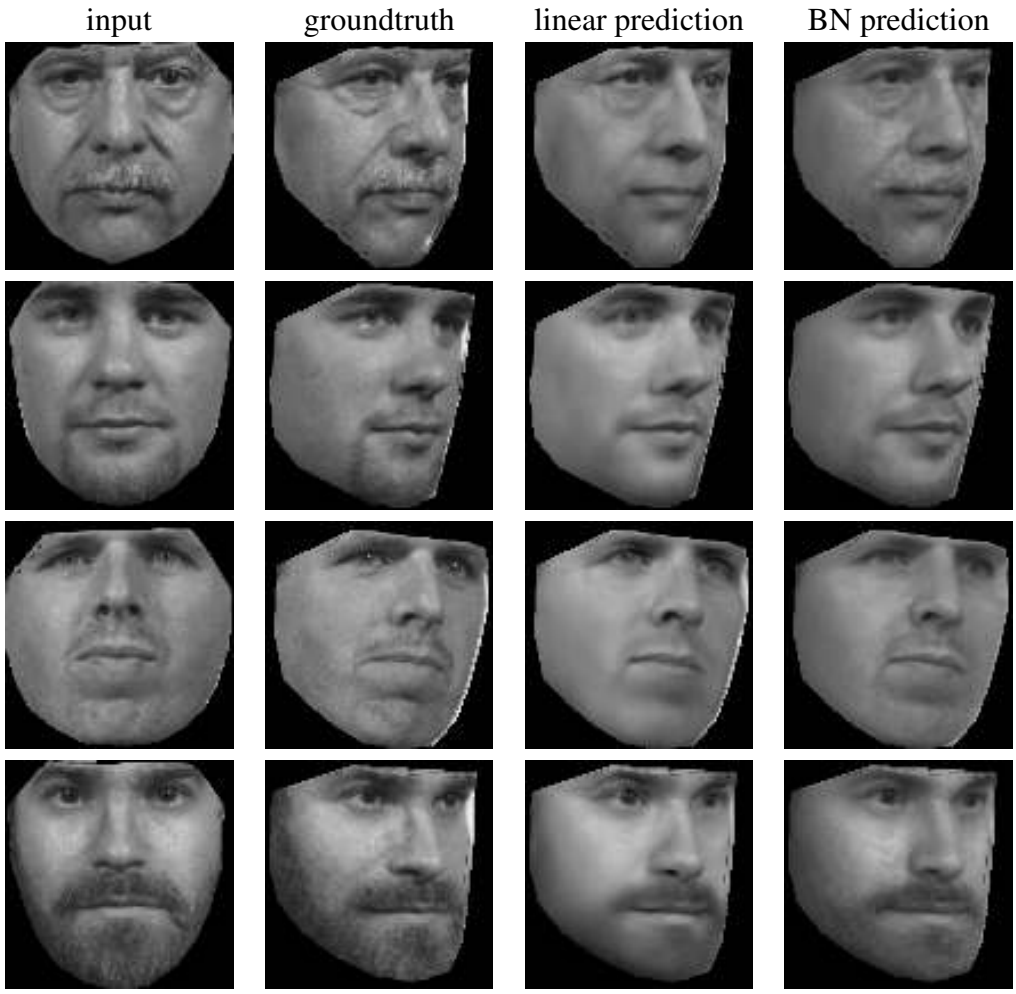

Figure 3.8: Results comparing the linear method and the Bayesian network method. The four examples show the linear method sometimes removes the mustache or loses the texture of the mustache while the Bayesian network approach predicts the mustache better.

Figure 3.9: Results comparing the linear method and the Bayesian network method. The four examples show the Bayesian network approach predicts the skin textures better than the linear method does. In the fourth row, the mole on the subject's face is totally lost in the synthesis of the linear method, but it is shown as a slightly darker area in the Bayesian network prediction.

	MSE in Appearance, predicting frontal
Linear Method	165.8925
BN method	151.3254

Table 3.2:
Comparison of mean-squared error over an independent test set, in predicting frontal from half profile.

times has artifacts around the eye area, while Bayesian network approach does not have this problem. In the fourth row, the mole on the subject's face is totally lost in the synthesis of the linear method, but it is shown as a slightly darker area in the Bayesian network prediction.

In terms of numerical results, the Bayesian network also gives better overall quantitative performance than the linear method (table 3.2). In comparing mean-squared error, the Bayesian network's average error in appearance prediction for the 40 test images is 151.3254, while in the linear method this average error is 165.8925. So the average improvement in appearance error among the 40 test images is 14.5672 ± 7.4785, with a margin of error for 90% confidence level.

3.8 Conclusions

In this chapter, we explore the Bayesian network approach that models the sparse statistical structure among the variables of the two poses. A Bayesian network uses a collection of lower dimensional models, with a conditional probability distribution at each node, so it leads to less overfitting. Such an approach can preserve localized features visually and numerically, in improving the performance of the appearance prediction.

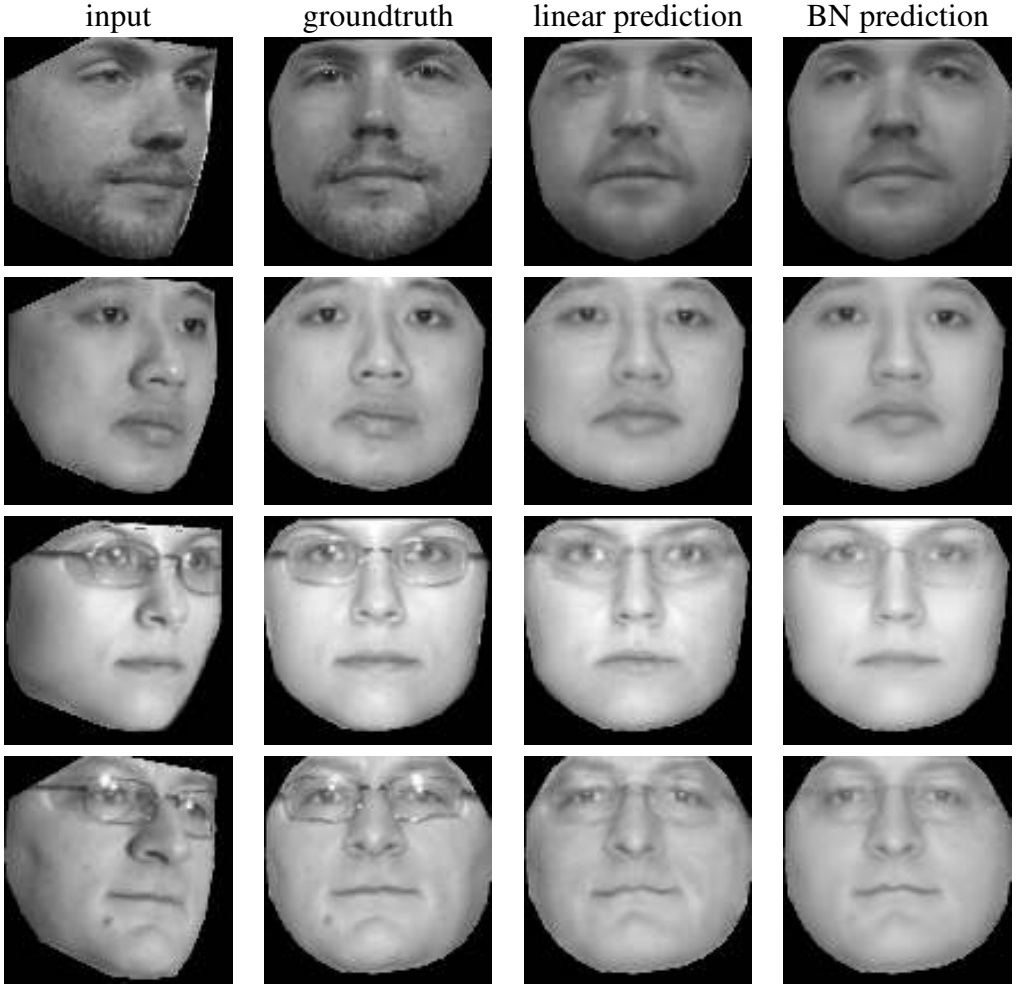

Figure 3.10: Results comparing the linear method and the Bayesian network method, in predicting frontal from half profile. The first three rows show the linear method sometimes has artifacts around the eye area, while Bayesian network approach does not have this problem. In the fourth row, the mole on the subject's face is totally lost in the synthesis of the linear method, but it is shown as a slightly darker area in the Bayesian network prediction.

Chapter 4

Discussion

In this chapter we will compare three methods: the linear method, the Bayesian network method and an established texture mapping method [42].

4.1 Texture Mapping

First we will describe our implementation of texture mapping. We use 100 images of 3D faces from the USF Human-ID 3D Face Database [41]. Each 3D face has 8955 vertices. We average these 100 3D faces with correspondence of vertices to get one 3D face model, then use this model to render 2D reference faces for frontal view and 30 degree view. The 3D face model provides the correspondence between the two 2D reference faces. Then we can directly copy the pixels of small meshes defined by the 8955 vertices from one view to the other. When a test image is given, we warp it to one reference face, copy the small patches to the other reference face, then unwarp it. The warping and unwarping procedures use positions of known landmarks on the face. The warping algorithm uses Delaunay triangulation which uses the Vonoroi diagram to find the triangles and then warp each triangle according to affine transform.

4.2 Comparison of the Three Methods in Theory

The linear method, the Bayesian network method, and the texture mapping method differ in the following assumptions:

- Dimensionality

The linear method and the Bayesian network method use different ways to group variables. The linear method considers all variables holistically, in a single high dimensional model. The graphical representation of a Bayesian network consists of a collection of models at each node in the graph. Each node models a conditional probability distribution over the variables flowing into the node. Each of these models is typically of much lower dimension than the total number of variables. On the other hand, the texture mapping does not group variables at all. It directly copies pixels to the corresponding locations.

- Learning

The linear method and the Bayesian network method both use information from the training data and the input test image, because both are statistical models with Gaussian representation. The texture mapping copies pixels from input test image deterministically, and does not use a prediction model based on training data. With the correspondence known, all it needs is the input image.

- Independence of Shape and Appearance

The linear method assumes the shape and the appearance are independent, and analyzes them separately. The Bayesian network method allows dependency between the shape and the appearance. The texture mapping can only predict appearance on normalized shapes. The predicted appearance by texture mapping has to be warped using the shape predicted by another method, such as the linear method.

- Lambertian Assumption

The texture mapping method assumes Lambertian surfaces. So the texture mapping method can not handle shadow or highlights, and partial or full specularities. The linear method and the Bayesian network methods do not have such an assumption.

- Occlusion Modeling

Also, the texture mapping method requires pixel-to-pixel correspondence between the two views. So it cannot handle occlusion. The linear method and the Bayesian network method can handle occlusion.

4.3 Comparison of the Three Methods in Performance

The linear method is prone to make localized errors. For example, it creates artifacts over the eye areas. For people with no glasses, the prediction results show the eyes are surrounded by partially hallucinated pairs of glasses. The linear method often fails to predict distinctive local features such as mustaches or moles (Figure 2.10). One hypothesis is that, because of the holistic representation and high dimension, global control dominates over local control, compromising local feature appearance. The Bayesian network method and the texture mapping method preserve these distinctive local features.

The texture mapping method fails under occlusion or near occlusion. The nosewings disappear in the prediction results, because they are near occlusion in the frontal view. Also the side of the faces are not predicted because it is also near occlusion in the frontal view. The linear method and the Bayesian network method do not have this problem, because they can learn these occluded areas from the training data. (figure 4.1)

Figure 4.1: Texture mapping fails to predict nose appearance near occlusion. The nosewing disappears in the prediction.

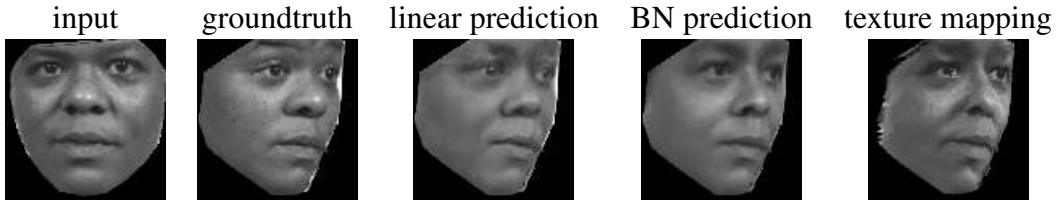

Figure 4.2: Results showing specularities or near specularities. With oily skin, the shines on the face are directly copied to the predicted face in the texture mapping result, which is not correct.

Texture mapping also fails to predict specularities or near specularities. Shiny areas on the face are re-mapped in a way that looks artificial and wrong. The linear method and the Bayesian network method do not create spurious or unnatural looking specularities. (figure 4.2 and 4.3) Also, in figure 4.3, the shape of the glasses is also predicted incorrectly due to the difference in the 3D structures between this subject and the general face model.

Figure 4.3: Results showing specularities on glasses. The texture mapping prediction gives fake reflections on glasses. And the shape of the glasses is also predicted incorrectly due to the difference in the 3D structures between this subject and the general face model.

Chapter 5

Conclusions and Contribution

In this thesis, we have addressed the problem of face view synthesis using a single image, and we focus on 2D approaches instead of building 3D models. Although a single 2D image does not contain depth information, our research show machine learning technique can learn a lot of prior information between poses, and can predict novel views with good performance. In particular, we formulate a regularized holistic linear model, to model the variables holistically and combine the "distance-in-feature-space" and "distance-from-feature-space" using a properly determined weight for regularization in order to get reliable prediction results. Such approach performs well in shape prediction and outperforms the linear model with no regularization, however, it is not always reliable in predicting localized features in appearance. To better capture localized relationship we use a Bayesian network. The de-centralized structure of a Bayesian network forms a collection of localized models, where each such model represents a group of statistically dependent variables. By having such a collection of lower dimensional models, with a conditional probability distribution at each node, a Bayesian network leads to less overfitting. Such an approach can preserve localized features visually and numerically, in improving the performance of the appearance prediction.

Bibliography

[1] Belhumeur, P.N., Kriegman, D.J., and Yuille, A.L., "The Bas-Relief Ambiguity", CVPR, 1997.

[2] Beymer, D. and Poggio, T., "Face recognition from one example view", ICCV, 1995.

[3] Black, M. and Jepson, A., "Eigen-tracking: Robust matching and tracking of articulated objects using a view-based representation," International Journal of Computer Vision, Vol.36, No.2, pp.101-130, 1998.

[4] Blanz, V., Mehl, A., Vetter, T., and Seidel, H.-P., "A Statistical Method for Robust 3D Surface Reconstruction from Sparse Data", International Symposium on 3D Data Processing, Visualization and Transmission, Thessaloniki, Greece 2004.

[5] Blanz, V. and Vetter, T., "A Morphable Model for the Synthesis of 3D Faces," ACM Siggraph, 1999.

[6] Blanz, V. and Vetter, T., "Face Recognition Based on Fitting a 3D Morphable Model," IEEE TPAMI, Vol.25, No.9, pp.1063-1074, 2003.

[7] Blanz, V., Grother, P., Phillips, P.J., and Vetter, T., "Face Recognition Based on Frontal Views generated from Non-Frontal Images," CVPR, 2005.

[8] Cheng, J., Greiner, R., Kelly, J., Bell, DA and Liu, W., "Learning Bayesian Networks from Data: an Information-Theory Based Approach", The Artificial Intelligence Journal, Volume 137, Pages 43-90, 2002.

[9] Cootes, T., Walker, K., and Taylor, C., "View-based active appearance models", International Conference on Automatic Face and Gesture Recognition, 2000.

[10] Cootes, T.F., Edwards G.J., and Taylor, C.J., "Active Appearance Models," IEEE TPAMI, Vol.23, No.6, pp.681-685, 2001.

[11] Cover, T.M. and Thomas, J.A., "Elements of Information Theory," New York: Wiley, 1991.

[12] Criminisi, A., Reid, I., and Zisserman, A., "Single view metrology", IJCV, pp434-442, September 1999.

[13] Debevec, P., Taylor, C., and Malik, J., "Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach," Computer Graphics (SIGGRAPH), pages 11-20, 1996.

[14] Forsyth, D.A. and Ponce J., "Computer vision: a modern approach", Prentice Hall, 2003.

[15] Gersho, A. and Gray, R.M., "Vector Quantization and Signal Compression", Boston, MA: Kluwer, 1992.

[16] Goldenberg, A. and Moore, A., "Tractable Learning of Large Bayes Net Structures from Sparse Data", ICML, 2004.

[17] Gross R., Matthews, I., and Baker, S., "Appearance-Based Face Recognition and Light-Fields," IEEE TPAMI, Vol.26, No.4, pp.449-465, 2004.

[18] Gross R., Matthews I., Cohn J., Kanade T., and Baker S., "The CMU multi-pose, illumination, and expression (Multi-PIE) face database," Technical Report, Carnegie Mellon University Robotics Institute. TR-07-08, 2007.

[19] Gu, L. and Kanade, T. 3D Alignment of Face in a Single Image Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York, 2006.

[20] Gu, L., Xing, E. and Kanade, T. "Learning GMRF Structures for Spatial Priors", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, 2007.

[21] Hoiem, D., Efros, A.A., and Hebert, M., "Geometric context from a single image", CVPR, 2005.

[22] Hong W., Yang A.Y., Huang K., and Ma Y., "On symmetry and multiple view geometry: Structure, pose and calibration from a single image," International Journal on Computer Vision (IJCV), 60(3):241-C265, December 2004.

[23] Horry, Y., Anjyo, K., and Arai, K., "Tour into the picture: using a spidery mesh interface to make animation from a single image", ACM Siggraph, 1997.

[24] Hwang, B-W. and Lee S-W., "Reconstruction of Partially Damaged Face Images Based on a Morphable Face Model," IEEE TPAMI, Vol.25, No.3, pp.365-372, 2003.

[25] Koller, D. and Friedman N., "Bayesian Networks and Beyond", book draft, 2005.

[26] Leonardis, A. and Bischof, H., "Robust recognition using eigenimages," Computer Vision and Image Understanding," Vol.78, No.1, pp.99-118, 2000.

[27] MacKay, D.J.C., "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, 2003.

[28] Moghaddam, B. and Pentland, A., "Probabilistic Visual Learning for Object Representation," IEEE TPAMI, Vol.19, No.7, pp.696-710, 1997.

[29] Neapolitan, R.E., "Learning Bayesian Networks," Pearson Prentice Hall, Upper Saddle River, NJ, 2004.

[30] Ni, J. and Schneiderman, H., "Face View Synthesis Across Large Angles," in Proceedings of the Second International Workshop on Analysis and Modeling of Faces and Gestures (AMFG 2005) held in conjunction with ICCV 2005, October, 2005.

[31] Pentland, A., Moghaddam, B., and Starner, T., "View-based and modular eigenspaces for face recognition", CVPR, 1994.

[32] P.J. Phillips, H. Wechsler, J. Huang, P. Rauss, "The FERET database and evaluation procedure for face recognition algorithms," Image and Vision Computing J, Vol.16, No.5, pp. 295-306, 1998,.

[33] P.J. Phillips, H. Moon, S.A. Rizvi, P. J. Rauss, "The FERET Evaluation Methodology for Face Recognition Algorithms," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 22, pp. 1090-1104, 2000.

[34] Poggio, T. and Vetter, T., "Recognition and structure from one 2D model view: Observations on prototypes, object classes, and symmetries", A.I. Memo No. 1347, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1992.

[35] Sali, E. and Ullman S., "Recognizing novel 3-D objects under new illumination and viewing position using a small number of example views or even a single view", ICCV, 1998.

[36] Schneiderman, H., "Learning Statistical Structure for Object Detection," Computer Analysis of Images and Patterns (CAIP), Springer-Verlag, August, 2003.

[37] Schneiderman, H., "Learning a Restricted Bayesian Network for Object Detection," CVPR, 2004.

[38] Shi, J. and Malik, J., "Normalized Cuts and Image Segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.22, No.8, pp.888-905, 2000

[39] Sim, T., Baker, S., and Bsat, M., "The CMU Pose, Illumination, and Expression Database," IEEE TPAMI, Vol.25, No.12, pp.1615-1618, 2003.

[40] Spirtes, P., Glymour, C., and Scheines, R., "Causation, Prediction, and Search", The MIT Press, Cambridge, Massachusetts, 2000.

[41] USF DARPA Human ID 3D face database. Courtesy of Prof. Sudeep. Sarkar, University of South Florida, Tampa, FL.

[42] Vetter, T., "Synthesis of novel views from a single face image", IJCV, Vol.28, No.2, pp.103-116, 1998.

[43] Vetter, T. and Poggio, T., "Linear object classes and Image Synthesis From a Single Example Image," IEEE TPAMI, Vol.19, No.7, pp.733-742, 1997.

[44] Wei, H., "Face pose change based on a 3D model", Tsinghua University Master thesis, 2000.

[45] Wiskott, L., Fellous, J.-M., and von der Malsburg, C., "Face recognition by elastic bunch graph matching", IEEE TPAMI, Vol.19, pp.775-779, 1997

[46] Xiao, J., "Reconstruction, Registration, and Modeling of Deformable Object Shapes," CMU RI PhD thesis, 2005.

[47] Yehezkel, R. and Lerner B., "Bayesian Network Structure Learning by Recursive Autonomy Identification", SSPR/SPR, ser. Lecture Notes in Computer Science, vol. 4109, pp. 154-162, 2006.

[48] Zhang, L., Dugas-Phocioin, G., Samson, J.-S., and Seitz, S.M., "Single view modeling of free-form scenes", Journal of Visualization and Computer Animation, vol.13, No.4, pp.225-235, 2002.

[49] Zhao, W. and Chellappa, R., “SFS based view synthesis for robust face recognition”, International Conference on Automatic Face and Gesture Recognition, 2000.