M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 417
also appears as MIT EECS Master's Thesis February 1997

A Three-Dimensional Model of Human Lip Motion

by

Sumit Basu

S.B. in Electrical Engineering, Massachusetts Institute of Technology (1995)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1997

© Massachusetts Institute of Technology, MCMXCVII. All rights reserved.

The author hereby grants to MIT permission to reproduce and distribute publicly paper and electronic copies of this thesis document in whole or in part, and to grant others the right to do so.

Author .....
Department of Electrical Engineering and Computer Science
January 31, 1997

Certified by .....
Alex P. Pentland
Department Head/Associate Professor, Media Arts and Sciences
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Theses

A Three-Dimensional Model of Human Lip Motion

by

Sumit Basu

Submitted to the Department of Electrical Engineering and Computer Science
on January 31, 1997, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

I present a 3D model of human lips and develop a framework for training it from real data. The model starts off with generic physics specified with the finite element method and "learns" the correct physics through observations. The model's physics allow physically-based regularization between sparse observation points and the resulting set of deformations are used to derive the correct physical modes of the model. Preliminary results showing the model's ability to reconstruct lip shapes from sparse data are shown. The resulting model can be used for both analysis and synthesis.

Thesis Supervisor: Alex P. Pentland
Title: Department Head/Associate Professor, Media Arts and Sciences

Acknowledgments

I would first of all like to thank my advisor, Sandy Pentland, for his unfailing support and encouragement every step of the way. With his help, I could always see the big picture, even when lost in the most frustrating details.

Next of course is Irfan Essa, who got me started in all kinds of ways: he led me through some of my first steps in the research world, introduced me to finite elements, and always had time to discuss, explain, and just chat. When I had just finished my PWE's and was ready to drop a paper submission we had been working on, he pulled an all-nighter with me to help me get it done, leading to my first major paper.

Many thanks to the National Science Foundation for their support in the form of a Graduate Research Fellowship.

Thanks to my Academic Advisors, John Wyatt and Erich Ippen, for helping me to plan my courses and get the right requirements done at the right times.

Thanks (as always) to Anne Hunter, whose infinite help and reassurance on countless occasions has been critical in making everything run smoothly.

Thanks to all of the EECS and MAS faculty that have taught me the theory and tools I've needed to get to where I am, especially Bob Gallagher, Aaron Bobick, Victor Zue, Eric Grimson, John Wyatt, Hameed Nawab, Tom Quatieri, Bill Siebert, Erich Ippen, Cam Searle, Greg Papadopoulos, Lynn Stein, and Jesus del Alamo.

Thanks to Todd Cass, Warren Jackson, David Biegelsen, David Jared, and all the other great people at Xerox PARC who showed me how wide and wonderful the world of EE and CS could be.

Thanks to some of the super UROPs, especially Ken Russell for all of his help with Inventor and every other aspect of the SGI's and Cyrus Eyster for developing tools to deal with compressed video and audio data.

A special thanks to Deb Roy, Tony Jebara, and Chris Wren, both for their comraderie and for their willingness to discuss my latest ideas. Thanks to Deb for many years of guidance and discussions from way back in the UROP days, and most recently for kicking me into thesis gear when the time was right.

Thanks to Dave Becker for believing I could do something good for the world.

Thanks to Kris Popat, Martin Szummer, Tom Minka, and Fang Liu for many fascinating technical discussions on every possible topic.

Thanks to Erik "trims" Trimble for keeping all the machines running and improving our computing environment by the day.

Thanks also to Laurie and Karen for keeping all of Vismod running and improving our working environment by the day.

I will never forget the endless help, encouragement, and friendship of all the other Pentlandians: Trevor, Baback, Flavia, Wasi, Jeff, Ali, Hong, Ifung, and Nuria. Thanks also to all the other Vismod-ers past and present.

Thanks to all of my other friends for helping to relieve the pressure: Hanoz, Dan Gruhl, Djioh, Mike Decerbo, Lisa-Maria, Dhaya, Howard, Jeanie, Audrey, Connie, and of course Sejal.

Last but certainly not least, my thanks and love to my family. Mom, Dad, and Sharmila: without your help, support, and encouragement, I would never have gotten to where I am today. Thanks for everything - literally.

Contents

1 Introduction **7**
1.1 Background 7

2 The Model **10**
2.1 The initial shape 10
2.2 The finite element method 11
2.3 Model Specifics 12
2.4 Applying the method to the lips 15

3 The Observations **17**

4 Training the Model **22**
4.1 Reaching the Displacement Goals 22
4.2 Modeling the Observations 24

5 Results **26**

6 Conclusions and Future Directions **29**
6.1 Improving the Model 29

A Least Squares Solution with a Minimum Norm Constraint **30**

B Finding the Eigenvectors of a Sample Covariance Matrix **32**

List of Figures

2-1 Initial shape of lips 10
2-2 Example deformations of a planar mesh 15

3-1 Locations of marked points on the face 17
3-2 Data collection setup 18
3-3 Raw input from camera-mirror setup 18
3-4 Left (virtual) and right camera views 19
3-5 Point reconstruction geometry 19
3-6 An input frame and the reconstructed 3D points 20
3-7 Normalization of 3D point coordinates 21

4-1 Initial model and deformation to rest state 23
4-2 Example deformations of the model 23

5-1 The mean displacement and some characteristic modes 26
5-2 LLSE estimate of lip shape using point data from first eight points . . 27
5-3 LLSE estimate of lip shape using only y-z data 27
5-4 LLSE estimate of lip shape using only x-y data 27

List of Tables

5.1 Reconstruction error per DOF (in normalized coordinates) 28

Chapter 1

Introduction

In the course of ordinary conversation and expression, the human lips can deform in a variety of interesting and non-rigid ways. Tracking these motions accurately has proved to be quite difficult. There are a number of reasons behind this, but primary among them is that all of the information about the lips is not observable at a given point. For example, we can at best see only the inner and outer contours of the lips from a given view. In addition, the presence of the tongue can obscure the inner contour, and the flesh tone/lighting can obscure the outer contour. I propose that with an accurate spatio-temporal model of the lips, this task will be greatly simplified. In essence, the argument is that a model that can only vary in the ways that the lips actually move will not be "fooled" by erroneous data. Such a model could form the heart of a robust estimation framework for finding the best estimate of the lip pose given whatever data was available. In addition, this model could be used to synthesize lip motions for computer graphics. In this study, I present a model and a means for training it that I feel will be able to provide these benefits for analysis and synthesis.

1.1 Background

In looking at the prior work, there are two major groups of lip models. The first of these contains the models developed for analysis, usually intended for input into a combined audio-visual speech recognition system. The underlying assumption behind most of these models is that the head will be viewed from only one known pose. As a result, these models are often only two-dimensional. Many are based directly on image features: Coianiz *et al.* [6] and Kass *et al.* [9] model the lips with contours along the outer edge, while Duchnowski *et al.* [7] feed the raw pixel intensities into a neural net to classify lip shapes. Others use such low level features to form a parametrization of the lip shape: Petajan *et al.* use several image features to estimate an overal lip contour [13]; Adjoudani *et al.* relate a small set of observed features (such as lip opening width and height, etc.) to the controls of a polygonal lip model [1].

Still others have a trained model of lip variations and attempt to fit the observations to this model. Some of the most interesting work done in this area has been along these lines: Bregler and Omohundro's work, for example [5], models the non-

linear subspace of valid lip poses within the image space and can thus be used for both analysis and synthesis. Similarly, Luettin's system learns the subspace of variations for 2D contours surrounding the lips [11]. However, in order for these 2D models to be robust, they have to allow for at least small rotations of the head. The changes in the apparent lip shape due to rigid rotations, then, have to be modeled as changes in the actual lip pose. Our goal is thus to extend these ideas to 3D. By modeling the true three-dimensional nature of the lips, variations that look complex and nonlinear from a 2D perspective become simple and linear.

With a 3D model, we can simply rotate the model to match the observed pose, thus modeling only the actual variations in lip pose. Some researchers have argued that only two-dimensional information is observable and that it thus makes sense to base measurements and models on 2D features alone. However, while it is true that only the two-dimensional contours (at best) are visible in a given frame, the meaning of those two dimensions changes as the subject moves around - with changes is pose, the unobservables become observable. There has been some work done taking information from two known views [1], but this requires the head to remain fairly static. We feel that in order to capture interesting lip data during *natural* speech and gesture, it will be necessary to robustly track the lips from any pose. In addition, in order to fully train this model, it will be necessary to apply the observations from an arbitrary pose. Prior work has shown that the rigid position of the head can be robustly and accurately tracked [3], so it is feasible that we can apply the observations from any pose to the correct degrees of freedom of the model. As a result, our goal has been to create a model that can cover the full 3D variations of the lips.

The other category of lip models are those designed for synthesis and facial animation. These lip models are usually part of a larger facial animation system, and the lips themselves often have a limited repertoire of motions [10]. To their credit, these models are mostly in 3D. For many of the models, though, the control parameters are defined by hand. A few are based on the actual physics of the lips: they attempt to model the physical material and musculature in the mouth region [8],[16]. Unfortunately, the musculature of the lips is extremely complicated and has proved to be very difficult to model accurately. The basic physiology is comprised of an ellipsoidal muscle (the *Obicularis oris*) surrounding the mouth and several muscles which push and pull on this ring. This ellipsoidal muscle is exceedingly difficult to model in a computationally economic way. In addition, even if one were able to completely and correctly model the muscles and materials, the problem would not be solved: there would still be the very difficult issue of control. We do not have independent control of all of these facial muscles: the observed set of facial motions seem to be a slim subspace of the full range implied by the muscles. Some models, as in the work by Frisbie and Waters, have tried to approximate this subspace by modeling key lip positions (visemes) and then interpolating between them [16]. However, This limits the "correct" set of lip shapes to those fit by hand, without modeling how the lips really move between them.

I hope to fill the gap in these approaches with a 3D model that can be used for both analysis and synthesis. My approach is to start with a 3D shape model and generic physics. I then deform this initial model with real 3D data to learn the correct

physics and thus the correct modes of variation. I will demonstrate this technique using a simple finite element model and 3D point data, though the methodology could be applied to an arbitrary model and observation set. By training the modes of variation with real data, we can learn all of the deformation modes that occur in the observations. In this way, we not only solve the problem of parametrizing the model's motions, but also that of control. Because we learn only the modes that are observed, we end up with degrees of freedom that correspond only to plausible motions.

Chapter 2

The Model

The underlying representation of our initial model is a mesh in the shape of the lips constructed from a linear elastic material. In the following section, I give a brief description of the choice of model shape and the physics used.

2.1 The initial shape

To get an initial shape for the model, a region surrounding the lips was extracted from a Viewpoint Data Labs model of the human head. There were some simple changes that were necessary to place this shape into a finite element framework. Because this mesh was designed for graphics and not for physics, there was some asymmetry between the right and left halves of the lips. By forcing these to be symmetric, it was ensured that the resulting stress-strain relations of the initial model would also be symmetric. In addition, some faces placed at strange angles for graphics effects were smoothed out to further facilitate the physical modeling. The final model has 336 faces and 204 nodes, resulting in 612 degrees of freedom (three per node). The initial shape of the model is shown in figure 2-1.

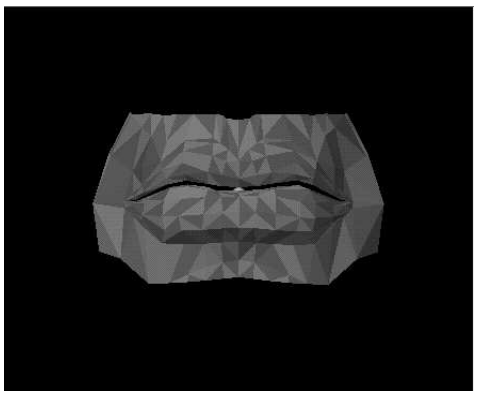

Figure 2-1: Initial shape of lips

2.2 The finite element method

The finite element method (FEM) is a numerical method for approximating the physics of an arbitrarily complex body. The central idea is to break the body into many small pieces whose individual physics are simple and then assembling the individual physics into a complete model. In contrast to the finite difference method, the finite element method also models the physics of the material *between* the nodes. This is possible because the method gives us interpolation functions that let us describe the entire modeled body in a piecewise analytic manner. Given this piecewise representation, the constitutive relations are *integrated* over the entire body to find the overall stress-strain relationship. These interpolation functions are written in vector form as:

$$\mathbf{u}(\mathbf{x}) = \mathbf{H}(\mathbf{x})\mathbf{U} \tag{2.1}$$

where $\mathbf{U}$ represents the values of the function to be interpolated at the nodes, $\mathbf{H}(\mathbf{x})$ is the interpolation matrix, and $\mathbf{u}(\mathbf{x})$ is the analytic representation of the function in the local coordinates of the element. In our case, the function we are in interested in is the strain ϵ (internal force) resulting from a given deformation. We can find this using the relation

$$\epsilon(\mathbf{x}) = \mathbf{B}(\mathbf{x})\mathbf{U} \tag{2.2}$$

where $\mathbf{U}$ now represents the displacements at each node and $\mathbf{B}$ is a combination of $\mathbf{H}(\mathbf{x})$ above and the stress-strain relationship of the material. It can obtained by appropriately differentiating and recombining the rows of $\mathbf{H}$ given the stress modes specified by ϵ. To find the stiffness matrix $\mathbf{K}$ for an entire element, we integrate this relationship over the volume of the element:

$$\mathbf{k_e} = \int \mathbf{B^T C B} dV \tag{2.3}$$

where $\mathbf{C}$ describes the stress-strain relationship between the different degrees of freedom in the element. For each such matrix, we have the relationship

$$\mathbf{k_e U} = \mathbf{F} \tag{2.4}$$

where $\mathbf{U}$ represents the nodal displacements and $\mathbf{F}$ the strain resulting from these displacements. Note that the stresses and the strains are both expressed in the local coordinate system at this point. Each of these element matrices can be transformed by a matrix λ, which transforms the global coordinate system to the local one:

$$\lambda = \begin{bmatrix} \longleftarrow & \hat{\imath} & \longrightarrow \\ \longleftarrow & \hat{\jmath} & \longrightarrow \\ \longleftarrow & \hat{k} & \longrightarrow \end{bmatrix} \tag{2.5}$$

where $\hat{\imath}$, $\hat{\jmath}$, and $\hat{k}$ are unit vectors in the local x, y, and z directions, respectively. Because these vectors are orthonormal, λ^{-1} is simply the transpose of the above matrix. λ^T thus transforms from the local cooordinate system to the global one.

Note that the matrix above transforms only three degrees of freedom: to apply it to the strain matrix for an entire element (which is nine-by-nine), we must repeat the same λ in the following block-diagonal form:

$$\mathbf{T} = \begin{bmatrix} \lambda & & \\ & \lambda & \\ & & \lambda \end{bmatrix} \tag{2.6}$$

The matrix $\mathbf{T}$ can then be applied to the element strain matrix to produce the strain matrix in the global coordinate system:

$$\mathbf{k'_e} = \mathbf{T^T k_e T} \tag{2.7}$$

In the expanded form on the right hand side, we can see how in a vector post-multiplication (by a global displacement) this $\mathbf{k'_e}$ first transforms the vector to the local coordinate system (with $\mathbf{T}$), applies the stress-strain relation (with $\mathbf{k_e}$), and transforms the resulting force back into the global coordinate system (with $\mathbf{T^T}$).

The resulting transformed strain matrices now have aligned degrees of freedom and can be assembled into a single, overall matrix such that

$$\mathbf{KU} = \mathbf{F} \tag{2.8}$$

where the displacements and forces are now in the global coordinate system.

2.3 Model Specifics

For this application, a thin-shell model was chosen. I constructed the model by beginning with a 2D plane-strain isotropic material formulation [17] and adding a strain relationship for the out-of-plane components. For each triangular element, then, the six in-plane degrees of freedom are related with a six-by-six matrix $\mathbf{k_{xy}}$, while the out-of-plane degrees of freedom are related by the three-by-three $\mathbf{k_z}$. In order to preserve the linearity of our model while maintaining the use of flat elements, I treat these two modes as being decoupled. They are thus assembled into the total $\mathbf{k_e}$ as shown in block-matrix form below:

$$\mathbf{k_e} = \begin{bmatrix} \mathbf{k_{xy}} & \\ & \mathbf{k_z} \end{bmatrix} \tag{2.9}$$

I built the 2D $\mathbf{k_{xy}}$ using the formulation as described by Zienkiewicz [17] and Bathe [4]. This formulation has the following stress modes:

$$\epsilon = \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{bmatrix} \tag{2.10}$$

where u and v correspond to displacements along the x and y dimensions of the local

coordinate system. Using the relation in equation 2.2 and the overall displacement vector

$$\mathbf{U_e} = \begin{bmatrix} u_1 & v_1 & u_2 & v_2 & u_3 & v_3 \end{bmatrix}^T \tag{2.11}$$

we can solve for $\mathbf{B}$. In addition, the material matrix $\mathbf{C}$ is

$$\mathbf{C} = \frac{E(1-v)}{(1+v)(1-2v)} \begin{bmatrix} 1 & \frac{v}{1-v} & 0 \\ \frac{v}{1-v} & 1 & 0 \\ 0 & 0 & \frac{1-2v}{2(1-v)} \end{bmatrix} \tag{2.12}$$

where E is the elastic modulus and v is Poisson's ratio. For the lip model, Poisson's ratio was chosen to be 0.01. Since the elastic modulus E is a constant multiplier of the entire material matrix, it can be used to vary the stiffness of the element is a whole. As a result, a default value of 1.0 was used for this parameter. Elements that were to be more or less stiff than the default material were then assigned larger and smaller values respectively. Some demonstrations of the effects of varying the relative stiffnesses of elements in a mesh will be shown later in this section.

The next step is to relate the out-of-plane degrees of freedom. It is important at this stage to consider the desired behavior of the material. If it were important for nodes to be able to move independently out of the plane without causing strain in the adjoined nodes, the $\mathbf{k_z}$ of equation 2.9 should be diagonal. In this case, however, it is desired that "pulling" on a given node has the effect of "pulling" its neighbors along with it. As a result, we construct the following $\mathbf{k_z}$:

$$\mathbf{k_z} = \frac{E(1-v)}{(1+v)(1-2v)} \begin{bmatrix} 1 & -0.5 & -0.5 \\ -0.5 & 1 & -0.5 \\ -0.5 & -0.5 & 1 \end{bmatrix} \tag{2.13}$$

Consider this matrix in terms of the basic relation $\mathbf{k_e u_e} = \mathbf{f_e}$. A positive displacement (out of the plane) in only one of these degrees of freedom produces negative forces (into the plane) in the other two. This means that stretching one node out of the plane without moving the other two *would require* forces pushing the other two down. When equal displacements are applied to all three nodes, there is zero strain on the element, resulting in a rigid motion mode out of the plane. Though the out-of-plane degrees of freedom are decoupled from the in-plane components, this mechanism acts to relate the strain energy to the deformation of the element due to out-of-plane displacements. The greater the disparity in out-of-plane displacements (i.e., the greater the stretching of the element due to such motions), the greater the strain energy is produced by this k_z.

Now that we have both $\mathbf{k_{xy}}$ and $\mathbf{k_z}$, we can from the complete strain matrix in equation 2.9. In order to keep the degrees of freedom in x-y-z order, we then rearrange

the rows and columns to produce the final $\mathbf{k_e}$:

$$\mathbf{k_e} = \begin{bmatrix} k_{x_1x_1} & k_{x_1y_1} & & k_{x_1x_2} & k_{x_1y_2} & & k_{x_1x_3} & k_{x_1y_3} & \\ k_{y_1x_1} & k_{y_1y_1} & & k_{y_1x_2} & k_{y_1y_2} & & k_{y_1x_3} & k_{y_1y_3} & \\ & & k_{z_1z_1} & & & k_{z_1z_2} & & & k_{z_1z_3} \\ k_{x_2x_1} & k_{x_2y_1} & & k_{x_2x_2} & k_{x_2y_2} & & k_{x_2x_3} & k_{x_2y_3} & \\ k_{y_2x_1} & k_{y_2y_3} & & k_{y_2x_2} & k_{y_2y_3} & & k_{y_2x_3} & k_{y_2y_3} & \\ & & k_{z_2z_1} & & & k_{z_2z_2} & & & k_{z_2z_3} \\ k_{x_3x_1} & k_{x_3y_1} & & k_{x_3x_2} & k_{x_3y_2} & & k_{x_3x_3} & k_{x_3y_3} & \\ k_{y_3x_1} & k_{y_3y_3} & & k_{y_3x_2} & k_{y_3y_3} & & k_{y_3x_3} & k_{y_3y_3} & \\ & & k_{z_3z_1} & & & k_{z_3z_2} & & & k_{z_3z_3} \end{bmatrix} \tag{2.14}$$

Once the degrees of freedom are properly aligned as described above, the resulting material has the approximate physics of many small plane-strain elements hooked together. The in-plane forces of one element can pull on both the in-plane and out-of-plane components of its neighbors, and the vice versa. Once the complete $\mathbf{K}$ matrix has been assembled, we have a linear approximation to the relationship between the displacement and the resulting strain. We can now invert this relationship to find the displacements produced by an applied force (external strain). However, the matrix cannot be inverted as is: it is necessarily singular, as there are several displacement vectors that produce no stress in the body (i.e., they exist in the nullspace of $\mathbf{K}$). These are the modes of rigid motion. Consider, for example, a uniform displacement applied to all of the nodes. This would clearly produce no strain on the body. As a result, a minimum set of nodes must be "grounded" (i.e., held fixed) to prevent these singularities. For a 3D body, two nodes (6 DOF) must be grounded. This amounts to removing the rows and columns corresponding to the degrees of freedom for these nodes. The remaining $\mathbf{K_s}$ has full rank and can be inverted to provide the desired strain-stress relation:

$$\mathbf{K_s^{-1}F_s} = \mathbf{U_s} \tag{2.15}$$

while $\mathbf{K}$ is easy to compute and band diagonal (due to the limited interconnections between nodes), finding its inverse is an expensive calculation. We thus want to take this inverse only once at a point where it is appropriate to linearize the physics.

To illustrate the stress-strain properties of this physical model, several 3D example deformations of a planar mesh are shown in figure 2-2. Two corners of the mesh are held fixed (this is the rigid constraint described above). An out-of-plane force is then applied to the central node of the mesh. The first figure (clockwise from upper left) shows the mesh in its initial state. The second figure shows the deformation when all of the elements have a uniform relative stiffness (i.e., $E = 1.0$). The third figure shows the effects of setting the stiffnesses of a ring of elements surrounding the center to a low value. Notice how these elements stretch much more than their stiffer neighbors. The last figure shows the effects of setting the stiffnesses of the elements inside the ring to a low value and the remaining stiffnesses to a much higher value. These stiff elements now prevent the effects of the applied force from significantly deforming the outer elements.

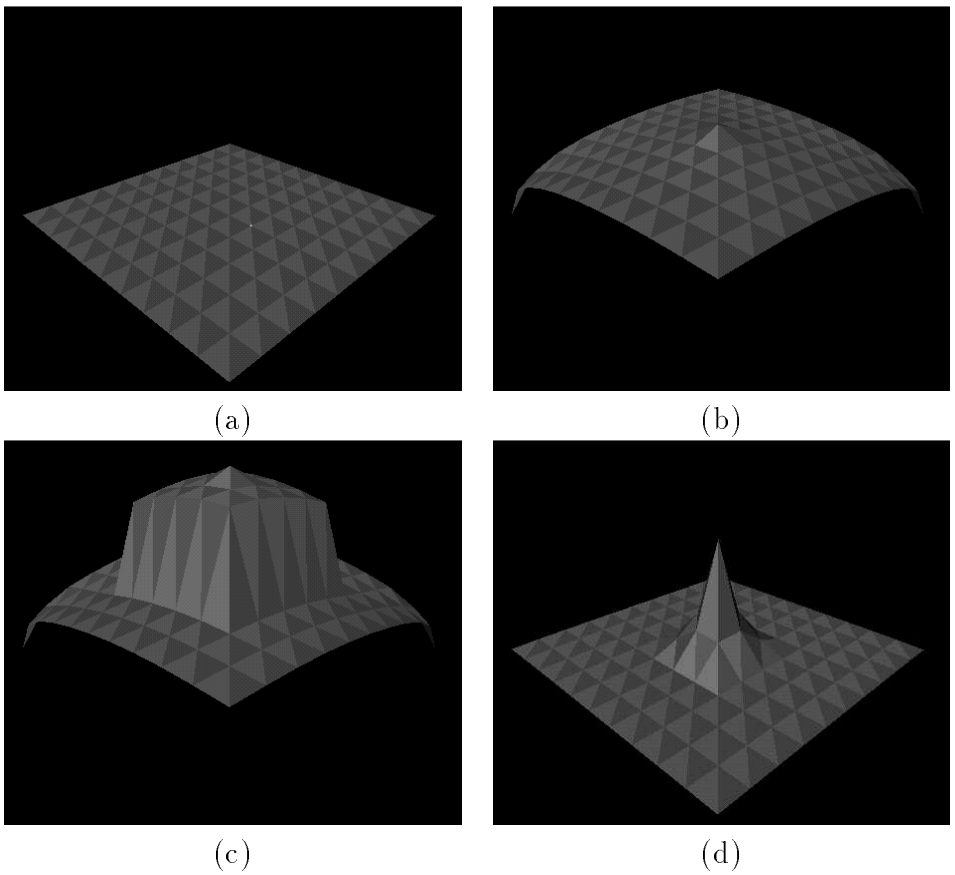

Figure 2-2: Example deformations of a planar mesh using the same force (applied out of the plane at the center of the mesh) with varying stiffness parameters: (a) initial state, (b) applied force with uniform stiffness, (c) applied force with a square ring of less stiff elements around the center, and (d) applied force with less stiff elements in the center and much stiffer elements in the rest of the mesh.

2.4 Applying the method to the lips

The method described above can be directly applied to the mesh in figure 2-1, resulting in a physical model in that shape made up of a uniform elastic material. However, in order to accentuate certain properties of the lips, some additional information was added to the model. First, in order to maintain smoothness in the inner contours of the lips, the faces along the inside ridges of the lips were made twice as stiff as the default material. In addition, to allow relatively free deformation of the lips while still maintaining the necessary rigid constraints, a thin strip of low-stiffness elements was added to the top of the lips stretching back into the oral cavity. The nodes at the far end of this strip were then fixed in 3D. Lastly, since the FEM linearizes the physics of a body around a given point, the initial $\mathbf{K}$ matrix was used to deform the model to the "rest state" of the lips (the method for this is described in a following

section). The final $\mathbf{K}$ and $\mathbf{K_s^{-1}}$ matrices were formed at this point, to allow for a greater effective range of the linearized physics. This $\mathbf{K_s^{-1}}$ was then used for all remaining calculations. The final state of the model after adding these constraints and deformations can be seen in figure 4-1.

Chapter 3

The Observations

To train this lip model to have the correct 3D variations of the lips, it was necessary to have accurate 3D data. Also, in order to observe natural motions, it was not acceptable to affix reflective markers or other easily trackable objects to the lips. To satisfy these criteria, seventeen points were marked on the face with ink: sixteen on the lips themselves and one on the nose. The placement of these points is shown in figure 3-1. Note that only a subset of the points drawn on the lips (those that are numbered in figure 3-1) were used for tracking. The points were chosen to obtain a maximally informative sampling of the 3D motions of the lips. The spacing of the points accounts for the relative nonlinearity of various regions: regions where more non-linearity was expected were sampled more heavily. In addition, with the results of this study, it should become clear where the greatest remaining non-linearities reside, thus guiding the next stage of training.

It is important to realize that the choice of points is not inherently limited to those shown. As the development below will show, any set of observations can be used to train the model. Of course, the higher the information content of the data, the less data will be required to train the model. In addition, because the points would be tracked from two views in this case, it was necessary to choose points that were visible over a reasonable range of head rotation from both perspectives.

Figure 3-1: Locations of marked points on the face

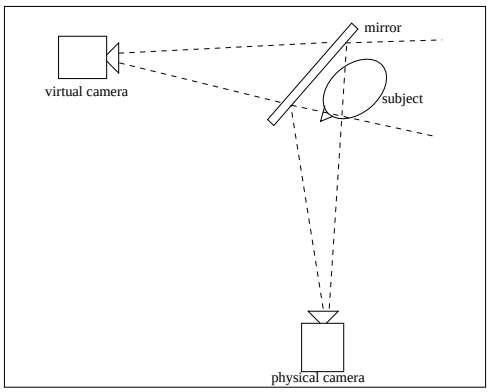

Figure 3-2: Data collection setup

Once the points were marked, two views of the points were taken by using a camera-mirror setup to ensure perfect synchronization between the two views (see figure 3-2). This resulted in two independent 2D views of the marked points. An example of the raw input from this setup is shown in figure 3-2.

Figure 3-3: Raw input from camera-mirror setup

As can be seen in figure 3-2, the left side of the raw camera view contains the flipped version of the virtual (left) camera view. The corrected, separated views are shown in figure 3-4 below. Note that the focal axis of the camera goes through the center of each displayed frame.

The points were tracked over 150 frames at a 30Hz frame rate using normalized correlation. Because of the large degree of motion of the lips between frames, normalized correlation did not always find the correct movement. As a result, a GUI was developed in which the user could move a tracker back to the correct point when normalized correlation failed.

It was attempted to have as great a variety of lip motions within this brief period as possible. To this end, several utterances using all of the English vowels and the major fricative positions were spoken during the tracking period. Clearly, 150 frames from one subject is still not enough to cover all possible lip motions, but it is enough to provide the model with the initial training necessary to cover a significant subset

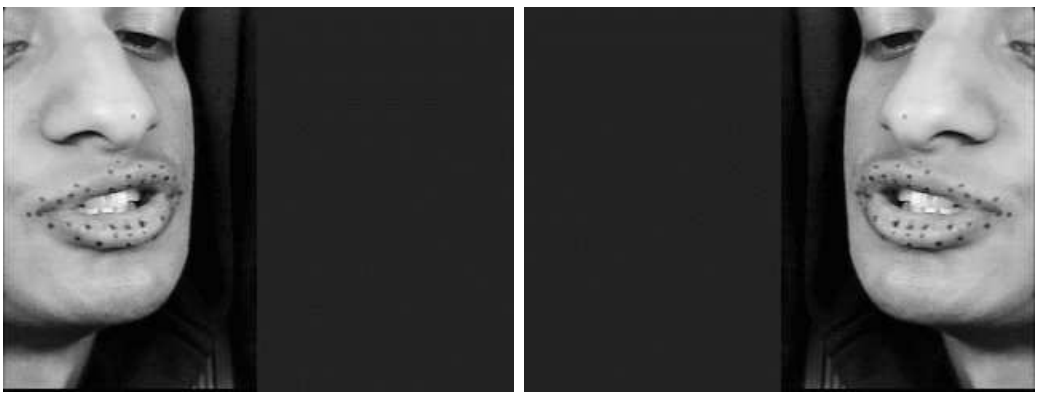

Figure 3-4: Left (virtual) and right camera views

of motions. Methods to continue the training using other forms of input data will be discussed in a later section. The 3D camera calibration algorithm of Azarbayejani and Pentland [2] was then used to calibrate the real and virtual cameras using pinhole camera models. Given this calibration, the 3D point location for a given point was estimated by computing the closest point between the projective rays from the camera COP's (centers of projection) corresponding to that point. A schematic showing the projective rays for a sample 3D point are shown in figure 3-5. Also, a sample frame showing the 3D reconstructed points and the original video frame is shown in figure 3-6. Note that the 3D points are shown at a different orientation than the camera views to demonstrate the reconstruction of depth information.

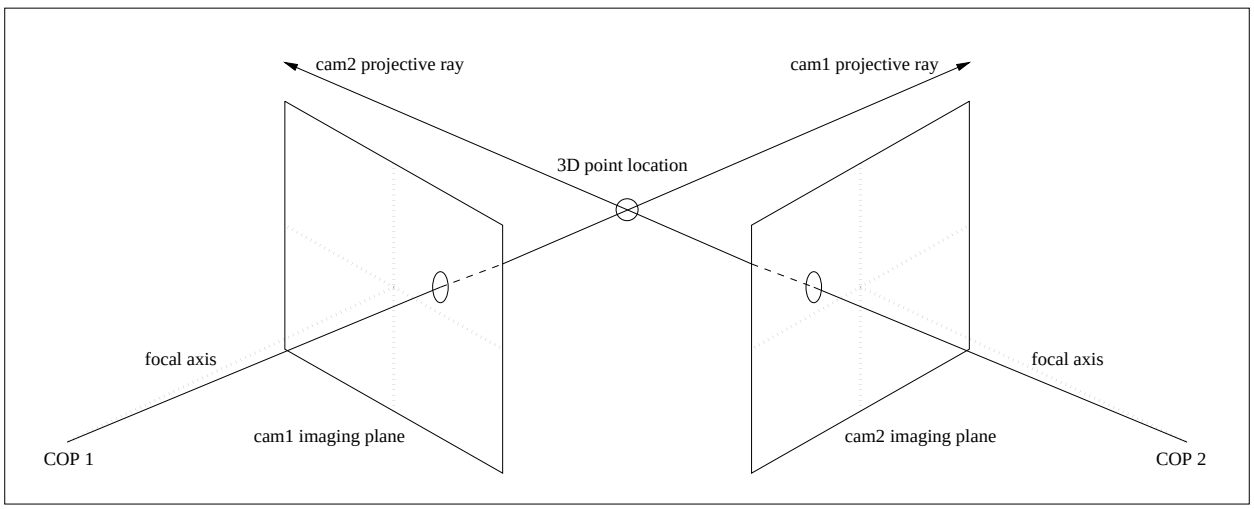

Figure 3-5: Point reconstruction geometry

After the reconstruction, several steps were taken to normalize the input data. The position of the nose was subtracted out from all points to help account for rigid translation. In addition, to normalize the effects of rigid rotation, planes were fit to the outer subsets of both the upper and lower lip point sets. The line forming the intersection of these planes was used to form the normalized coordinate: this line was

Figure 3-6: An input frame and the reconstructed 3D points

treated as the x axis, the y axis was formed by taking the global y and subtracting off the projection onto the new x, and the z axis was the cross product between the two. An example of fitting planes to the points and finding the resulting coordinate system is shown in figure 3-7 below.

In order to then transform the training data into the coordinate frame of the model, a graphical interface was used to find the best rigid rotation, translation, and scaling that would fit the default model to the first frame of this data set. This transform was then inverted to bring the data into the normalized coordinate system of the model. The philosophy behind this is that all of the training should be in the coordinate system of the model; the resulting model can then easily be scaled and moved (in 3D) to fit new incoming data. This final normalization step was done only once for the entire data set.

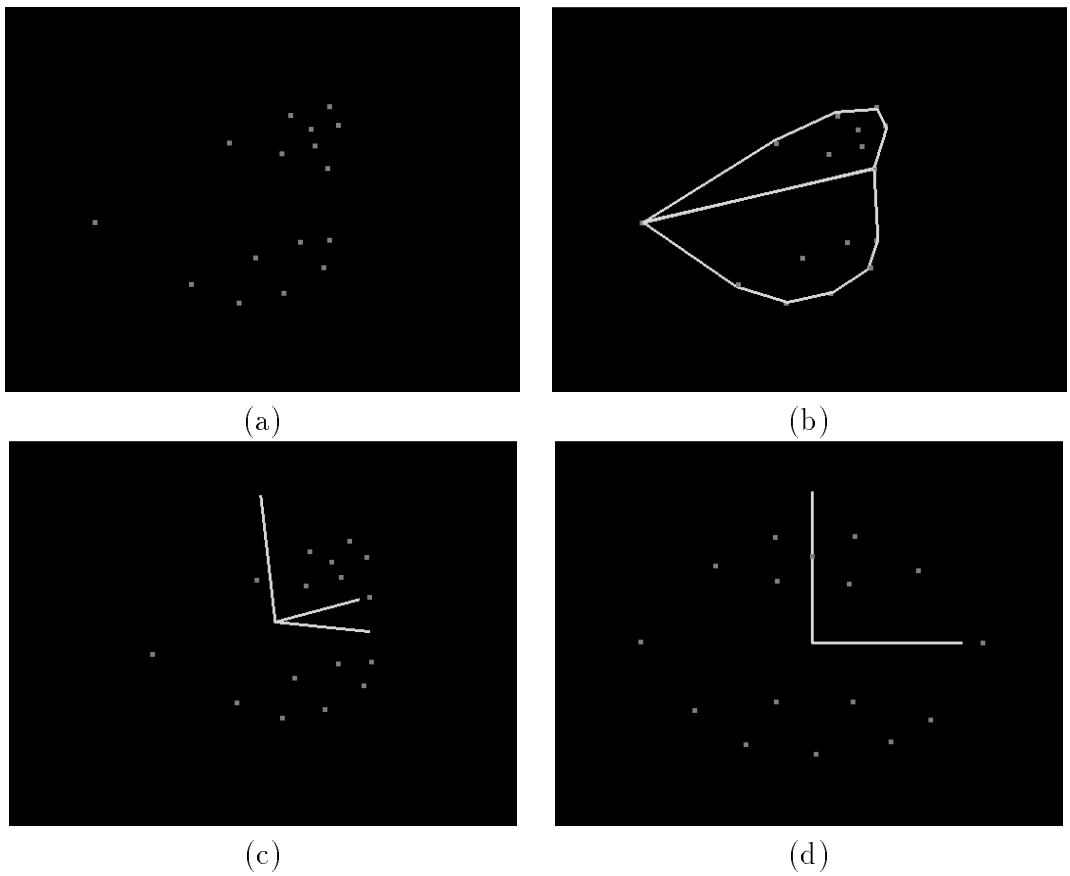

Figure 3-7: Normalization of 3D point coordinates: (a) original point locations, (b) planes fit to points and the line of their intersection, (c) normalized coordinate axes, and (d) the point locations in the normalized coordinate system.

Chapter 4

Training the Model

In order to relate the training data to the model, the correspondence between data points and model nodes had to be defined. This was a simple process of examining a video frame containing the marked points and finding the nodes on the lip model that best matched them in a *structural* sense. The difference between the goal locations of these points (i.e., the observed point locations) and their current location in the model is then the desired displacement goal, $\mathbf{U_g}$.

4.1 Reaching the Displacement Goals

The issue is now how to reach these displacement goals. The recorded data points have constrained 48 degrees of freedom (16 points on the lips with three degrees of freedom each). However, the other 564 degrees of freedom are left open. We thus have an underconstrained problem: there are an infinite number of ways to reach the desired displacement goals. However, we would not be satisfied with simply any one of these solutions - we want the one that minimizes the strain energy in the mesh. In other words, given the set of constrained point displacements, we want to find the set of displacements for the rest of the nodes that minimizes the strain felt throughout the structure. We denote the $\mathbf{K_s^{-1}}$ matrix with all 600 columns but only the rows pertaining to constrained degrees of freedom as $\mathbf{K_c^{-1}}$. We can then describe the problem in the following LaGrange multiplier formulation: we wish to minimize

$$\mathbf{F^T F} \tag{4.1}$$

with the constraint

$$\mathbf{K_c^{-1} F} = \mathbf{U_g} \tag{4.2}$$

which results in the solution

$$\hat{\mathbf{F}} = (\mathbf{K_c^{-1}})^{\mathbf{T}}(\mathbf{K_c^{-1}}(\mathbf{K_c^{-1}})^{\mathbf{T}})^{-1}\mathbf{U_g} \tag{4.3}$$

The detailed derivation of this result is given in Appendix A.

This solution is thus a physically based smoothing operation: we are using the physics of the model to smooth out the regions where we have no observation data by

minimizing the strain in the model. This method has thus given us displacements $\hat{\mathbf{U}}$ and forces $\hat{\mathbf{F}}$ for all 612 degrees of freedom for each frame. An example of using this method to deform the model is shown in figure 4-1 below. The deformed state shown was defined as the "rest state" of the lips as described above. The physics were thus re-linearized at this point, resulting in a final $\mathbf{K}_s^{-1}$ that was used for the rest of the deformations.

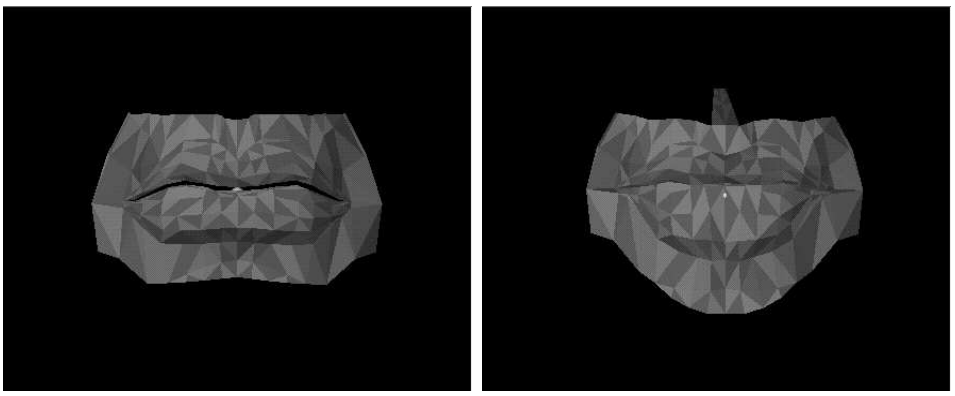

Figure 4-1: Initial model and deformation to rest state

Two frames of the original video along with the corresponding deformed model are shown in figure 4-2 below. Note that because the deformations of the model are shown in the coordinate system of the model, the deformations of the model will appear slightly different than the input data (applying the inverse of the rotation and independent scaling in x, y, and z that brought the data points to the model coordinate system would undo this effect).

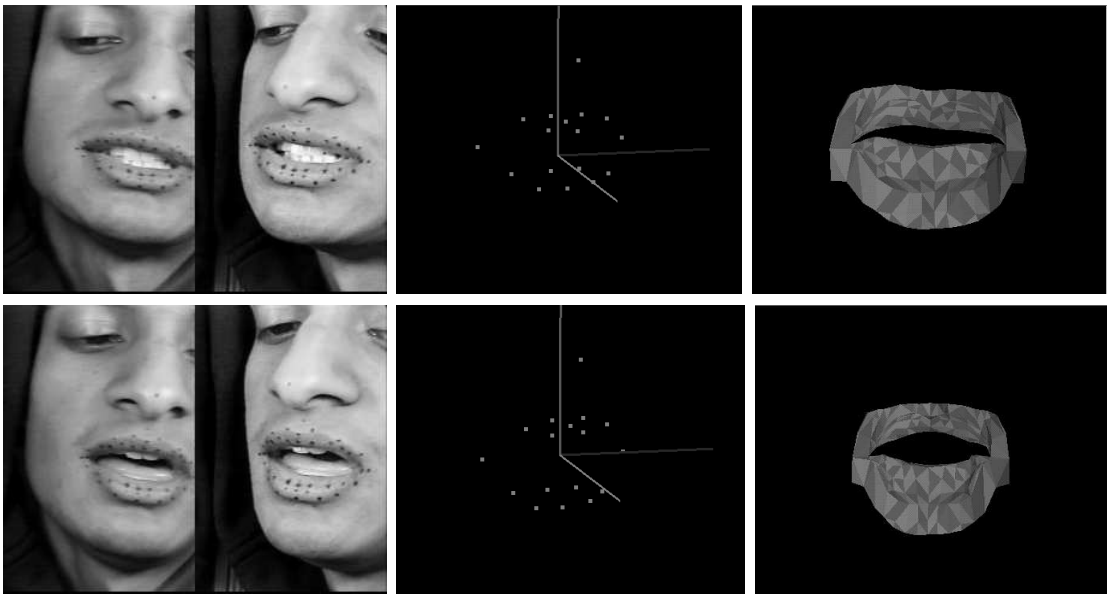

Figure 4-2: Example deformations of the model

4.2 Modeling the Observations

Once we have all the displacements for all of the frames, we can relate the observed deformations to a subset of the "correct" physics of the model. We began with the default physics (i.e., fairly uniform stiffness, only adjacent nodes connected) and have now observed how the model actually deforms. This new information can be used to form a new, "learned" $\mathbf{K}$ matrix. Martin et. al. [12] described the connection between the strain matrix and the covariance of the displacements. Since $\mathbf{K_s^{-1}}$ is a linear transform on $\mathbf{F}$, we can express the covariance of $\mathbf{U}$ in the following way:

$$\mathbf{K_U} = E[(\mathbf{K_s^{-1}F})(\mathbf{K_s^{-1}F})^\mathbf{T}] = \mathbf{K_s^{-1}}E[\mathbf{FF^T}](\mathbf{K_s^{-1}})^\mathbf{T} \tag{4.4}$$

If we now consider the components of the force to be IID with unit variance, the covariance matrix of $\mathbf{F}$ is the identity and we are left with

$$\mathbf{K_U} = \mathbf{K_s^{-1}}(\mathbf{K_s^{-1}})^\mathbf{T} = \mathbf{K^{-2}} \tag{4.5}$$

where the last step is possible because of the symmetry of $\mathbf{K_s}$ and thus of $\mathbf{K_s^{-1}}$. We can also take this mapping in the opposite direction: given the sample covariance matrix

$$\hat{\mathbf{K}}_\mathbf{U} = E[(\mathbf{U} - \bar{\mathbf{U}})(\mathbf{U} - \bar{\mathbf{U}})^\mathbf{T}] \tag{4.6}$$

We can find $\mathbf{K^{-1}}$ by taking its square root, i.e., diagonalizing the matrix into $\mathbf{S\Lambda S^T}$ (where each column of $\mathbf{S}$ is an eigenvector and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues) form and then reforming it with the square roots of the eigenvalues. We can then use the resulting "sample $\mathbf{K^{-1}}$" to represent the learned physics from the observations. Forces can now be applied to this matrix to calculate the most likely displacement given the observations.

However, because we only have a small number of observations (150) and a large number of degrees of freedom (612), we could at best observe 150 independent degrees of freedom. Furthermore, noise in the observations makes it unreasonable to estimate even this many modes. I thus take only the 10 observed modes that account for the greatest amount of variance in the input data. These modes are found by performing principal components analysis (PCA) on the sample covariance matrix [14], i.e., taking the eigenvectors and eigenvalues. Finding the eigenvalues and eigenvectors of the expected covariance matrix (which is 612 by 612) would take a great deal of computation. We can find the desired vectors by taking the eigenvectors of a much smaller matrix and then appropriately transforming the results. The details of this method are given in Appendix B. Note also that because the covariance matrix is symmetric, the eigenvectors will be orthogonal.

Once the eigenvectors were found, only the ten with the largest eigenvalues were kept. These were then normalized to unit length and used to reassemble the sample covariance matrix as $\mathbf{S\Lambda S^T}$.

We can also use the sample covariance matrix to find the linear least squares estimate (LLSE) of the overall displacements given any subset of the displacements as an observation. If we denote the observed displacements as $\mathbf{y}$ and the unknown

displacements as $\mathbf{x}$, we can estimate $\mathbf{x}$ with:

$$\hat{\mathbf{x}}_{\mathbf{LLSE}} = \bar{\mathbf{x}} + \mathbf{\Lambda}_{\mathbf{xy}}^{\mathbf{T}} \mathbf{\Lambda}_{\mathbf{y}}^{-\mathbf{1}} (\mathbf{y} - \bar{\mathbf{y}}) \tag{4.7}$$

We can easily pull the required covariance matrices out of our sample covariance matrix. Also note that we use the pseudoinverse of $\mathbf{\Lambda_y}$ in the case where $\mathbf{\Lambda_y}$ is not full rank.

A very interesting relationship exists between this solution and the least-squares solution of equation 4.3. Consider treating the quantity we wish to estimate as the combined vector of $\mathbf{x}$ and $\mathbf{y}$ (i.e., $\mathbf{U}$), and the covariance matrix of the data as that found in equation 4.5. The LLSE estimate of $\mathbf{U}$ is then *exactly* the solution we obtained in equation 4.7 above. In other words, minimizing the strain in the model for a given set of displacement constraints *is equivalent to* finding the LLSE estimate of the displacements using the covariances resulting from the physical model. This provides a full-circle link between the covariance of the input data and the strain matrix of the model.

Chapter 5

Results

The sample covariance was computed using only the first 140 frames so that the last ten could be used as a test set. After computing the covariance, the ten most expressive modes were found as described above. The mean displacement ($\bar{\mathbf{U}}$) and some of the first few modes are shown figure 5-1 below. It was found that the first ten modes cover 99.2 percent of the variance in the data - we should thus be able to reconstruct most shape variations from these modes alone.

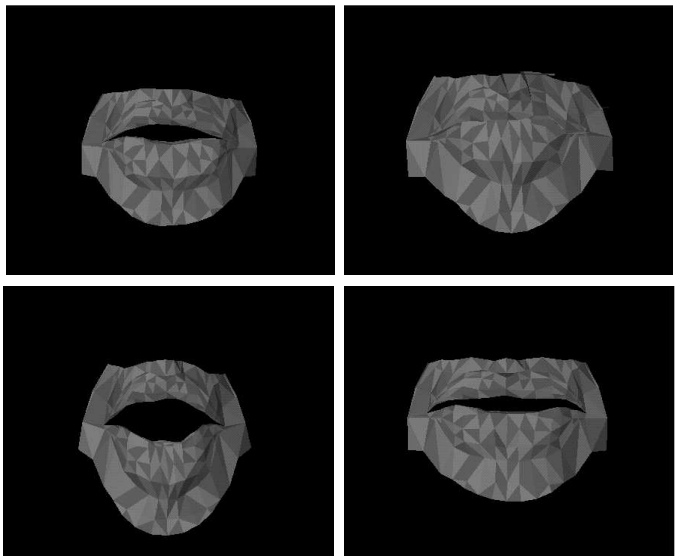

Figure 5-1: The mean displacement and some characteristic modes

To demonstrate how well this captures the variations in the data, the LLSE formulation of equation 4.7 was used with the complete input data for several frames to estimate the overall lip shape. The results for an example frame of the original video are shown along with this estimate in figure 5-2.

One of the major arguments behind the 3D representation was that we could use any number of observations from any viewpoint and find the best estimate of the model shape. To demonstrate this capability, I have again used the LLSE formulation to reconstruct the full shape, but this time using only y-z (figure 5-3) and only x-y

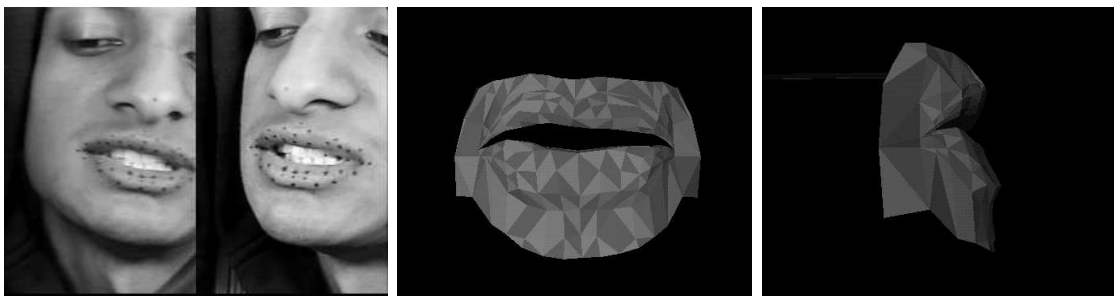

Figure 5-2: LLSE estimate of lip shape using point data from first eight points

(figure 5-4) data for the same frame as shown in figure 5-2.

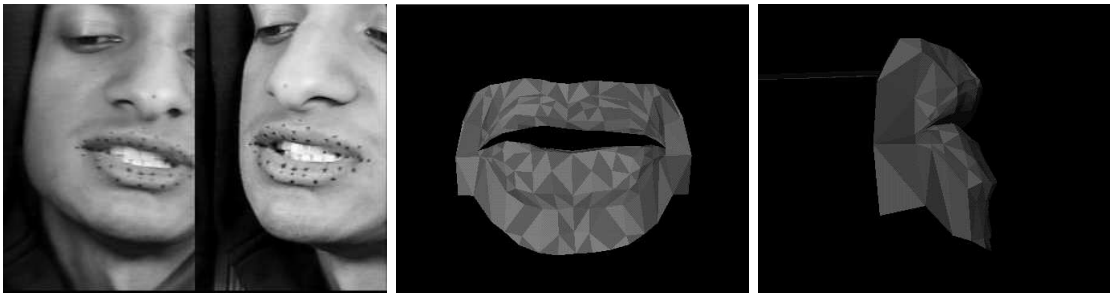

Figure 5-3: LLSE estimate of lip shape using only y-z data

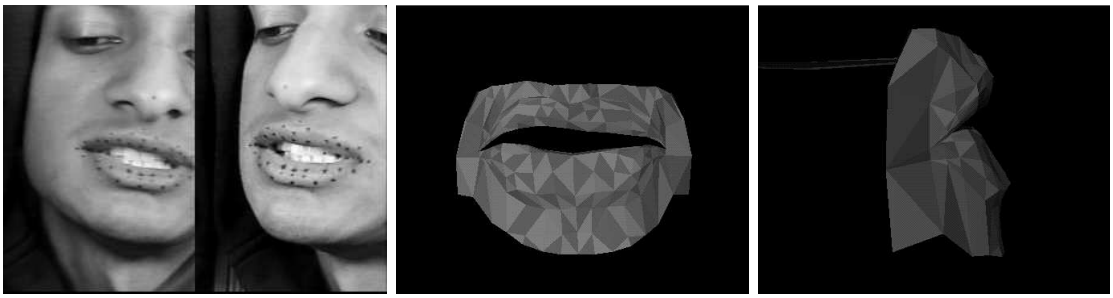

Figure 5-4: LLSE estimate of lip shape using only x-y data

In addition, for the ten data frames that were not included in the sample covariance, the mean-squared reconstruction errors per degree of freedom were found for several cases and are shown in the table below. The results are given in the coordinate system of the model, in which the model is 2.35 units across, 2.83 units wide, and 5.05 units deep. The first column of the table shows the reconstruction error using only the first ten modes; the second shows the error using the full covariance matrix. The fact that the performance is quite similar in both cases implies that the first ten modes have captured the important modes of variation in the data. The rows of the table correspond to what degrees of freedom were used to reconstruct the full 3D shape. In the first row, the first eight points (shown in figure 3-1) were used to reconstruct the remainder of the displacements. Note that the model performs quite well in this

case, again implying that it has learned to some degree the permissible subset of lip motions. The second row shows the results of using only the y and z components of the data. This corresponds to the data that would be available from a profile view. The last row contains the results using the x and y components (i.e., a frontal view). It is interesting to note that the y-z data provides much better peformance than the x-y case. This is understandable in that there was a significant amount of depth variation in the test frames. Because some x-y variations can occur with different degrees of z motion, the depth variation is not observable from the frontal plane. As a result, the y-z data provides more information about the lip shape in these cases. Since our model is a full 3D representation, it can take advantage of this disparity (or any other advantageous 3D pose) when these observations are available.

Table 5.1: Reconstruction error per DOF (in normalized coordinates)

Constraints Used	First 10 modes	Full Sample Covariance
xyz (8 points)	1.10e-3	7.80e-4
yz (all points)	7.13e-4	4.38e-4
xy (all points)	6.70e-3	7.50e-3

Chapter 6

Conclusions and Future Directions

With these preliminary experiments, the advantages of this full 3D analysis-synthesis model are clear. We can apply any number of observations for any points from any pose and robustly find the best lip shape estimate (although a minimum of points will be necessary for an accurate estimate). However, this is not the whole story. Clearly, it is not practical to mark and track points on every subject. To continue the training process, it will be necessary to use coarser features such as contours or color classification data. Though we will not have a direct correspondence with the model as in the case with point data, the relationship we presented between the covariance and the strain matrices allows us to use this information in a very different way. In essence, we can use the observed features to apply *forces* to the learned physics of the model. If a contour is not at the right position for the lip, it can apply a force in the right direction, and the learned physics will deform the rest of the model in an appropriate way.

6.1 Improving the Model

Because this model has only observed a subset of all possible lip motions, we cannot do the full deformation necessary for a new motion using only the learned physics. Clearly, as I have forced it to have a rank of only 10, it cannot cover all possible variations of the lips. It is thus necessary to retain the original full-rank physical model in order to reach the new deformations. It is critical to use both the learned and the original models together. The learned model acts as a prior for the gross deformations and can quickly get us to the neighborhood of the correct displacement. The original model can then make small perturbations to this to make the final fit. Using the learned model alone cannot reach all of the deformations, and using the original model alone can produce improbable lip displacements since it has no notion of the range of possible lip motions. With this combined approach, I can to continue to train this model to cover all possible lip motions.

Appendix A

Least Squares Solution with a Minimum Norm Constraint

In the general case, where we have a system

$$\mathbf{A}_{m\times n}\mathbf{x}_{n\times 1} = \mathbf{b}_{n\times 1} \tag{A.1}$$

in which A has more rows than columns ($m > n$), there are an infinite number of solutions (assuming the rows of $\mathbf{A}$ are linearly independent). However, if we also require that the norm

$$\mathbf{x^T x} \tag{A.2}$$

is minimized, there is a unique solution. To find this solution, we express the problem in a LaGrange multiplier formulation: we want to minimize the quantity in equation A.2 while satisfying the constraint in equation A.1. Using the method of LaGrange as described in Strang [15], we then have to minimize the following quantity:

$$\mathbf{f} = \frac{1}{2}\mathbf{x^T x} + \lambda^{\mathbf{T}}(\mathbf{Ax} - \mathbf{b}) \tag{A.3}$$

where λ is an $n \times 1$ vector of LaGrange multipliers for each of the rows of $\mathbf{Ax} - \mathbf{b}$. In other words, we are placing the constraint that each row of $\mathbf{Ax} - \mathbf{b}$ must be zero. We then take the vector partials of A.3 with respect to $\mathbf{x}$ and λ:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \mathbf{x} + \mathbf{A}^{\mathbf{T}}\lambda \tag{A.4}$$

$$\frac{\partial \mathbf{f}}{\partial \lambda} = \mathbf{Ax} - \mathbf{b} \tag{A.5}$$

When we set both of these partials to zero, we are left with the following system (shown in block-matrix form):

$$\begin{bmatrix} \mathbf{I} & \mathbf{A}^{\mathbf{T}} \\ \mathbf{A} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix} \tag{A.6}$$

We now solve this system directly to find

$$\begin{bmatrix} \mathbf{x} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{A^T}(\mathbf{AA^T})^{-1}\mathbf{b} \\ -(\mathbf{AA^T})^{-1}\mathbf{b} \end{bmatrix} \tag{A.7}$$

we have thus found the solution for x:

$$\mathbf{x} = \mathbf{A^T}(\mathbf{AA^T})^{-1}\mathbf{b} \tag{A.8}$$

Appendix B

Finding the Eigenvectors of a Sample Covariance Matrix

Consider the case where we have n observation vectors $\mathbf{O}$ in a high dimensional space $\Re^m$, where $(m > n)$. The unbiased estimate of the covariance matrix [14] is given by

$$\mathbf{K_s} = E[\mathbf{O}\mathbf{O}^\mathbf{T}] = \frac{1}{n-1}(\mathbf{A}\mathbf{A}^\mathbf{T}) \tag{B.1}$$

where $\mathbf{A}$ is an $m \times n$ matrix in which each column is an observation. Finding the eigenvectors of $\mathbf{K_s}$ (which is $m \times m$) can be computationally expensive when m is large. Fortunately, we can solve the much simpler problem of finding the eigenvectors and eigenvalues of $\mathbf{A}^\mathbf{T}\mathbf{A}$, which is only $n \times n$, and then transform the results. This can be easily seen by considering the definition of an eigenvector $\mathbf{x}$ of $\mathbf{A}^\mathbf{T}\mathbf{A}$ with eigenvalue λ:

$$\mathbf{A}^\mathbf{T}\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \tag{B.2}$$

we can then left-multiply both side by $\mathbf{A}$ and regroup terms:

$$(\mathbf{A}\mathbf{A}^\mathbf{T})(\mathbf{A}\mathbf{x}) = \lambda(\mathbf{A}\mathbf{x}) \tag{B.3}$$

The vector $\mathbf{A}\mathbf{x}$ is thus an eigenvector of $\mathbf{A}\mathbf{A}^\mathbf{T}$ with eigenvalue λ.

This only proves that the eigenvectors of $\mathbf{A}^\mathbf{T}\mathbf{A}$ transformed by $\mathbf{A}$ are *valid* eigenvectors of $\mathbf{A}\mathbf{A}^\mathbf{T}$. However, since $\mathbf{A}\mathbf{A}^\mathbf{T}$ is $m \times m$, the question remains as to which n of its m eigenvectors we are getting (since $\mathbf{A}^\mathbf{T}\mathbf{A}$ is only $n \times n$). The answer is simple: since $\mathbf{A}$ only had n independent columns to begin with, the rank of $\mathbf{A}\mathbf{A}^\mathbf{T}$ is only n. As a result, $\mathbf{A}\mathbf{A}^\mathbf{T}$ only has n eigenvectors with non-zero eigenvalues. Note that $\mathbf{A}^\mathbf{T}\mathbf{A}$ also is rank n and has n eigenvectors with *the same* non-zero eigenvalues as $\mathbf{A}\mathbf{A}^\mathbf{T}$. Therefore, the eigenvectors of $\mathbf{A}\mathbf{A}^\mathbf{T}$ corresponding to non-zero eigenvalues can be found by transforming the eigenvectors of $\mathbf{A}^\mathbf{T}\mathbf{A}$ by $\mathbf{A}$.

Lastly, note that though the eigenvectors of $\mathbf{A}\mathbf{A}^\mathbf{T}$ and $\mathbf{K_s}$ are the same, we must scale the eigenvalues by $1/(n-1)$ to find the eigenvalues of $\mathbf{K_s}$.

Bibliography

[1] A. Adjoudani and J. Benoit. "On the Integration of Auditory and Visual Parameters in an HMM-based ASR". In *NATO Advanced Study Institute: Speechreading by Man and Machine*, 1995.

[2] Ali Azarbayejani and Alex Pentland. Camera self-calibration from one point correspondence. Technical Report 341, MIT Media Lab Vision and Modeling Group, 1995. submitted IEEE Symposium on Computer Vision.

[3] Sumit Basu, Irfan Essa, and Alex Pentland. "Motion Regularization for Model-Based Head Tracking". In *To Appear in: Proceedings of 13th Int'l. Conf. on Pattern Recognition*, August 1996.

[4] Klaus-Jurgen Bathe. *Finite Element Procedure in Engineering Analysis.* Prentice-Hall, 1982.

[5] Christoph Bregler and Stephen M. Omohundro. "Nonlinear Image Interpolation using Manifold Learning". In *Advances in Neural Information Processing Systems 7*, 1995.

[6] Tarcisio Coianiz, Lorenzo Torresani, and Bruno Caprile. "2D Deformable Models for Visual Speech Analysis". In *NATO Advanced Study Institute: Speechreading by Man and Machine*, 1995.

[7] Paul Duchnowski, Uwe Meier, and Alex Waibel. "See Me, Hear Me: Integrating Automatic Speech Recognition and Lip-Reading". In *Int'l Conf. on Spoken Language Processing*, 1994.

[8] Irfan A. Essa. *"Analysis, Interpretation, and Synthesis of Facial Expressions"*. PhD thesis, MIT Department of Media Arts and Sciences, 1995.

[9] Michael Kass, Andrew Witkin, and Demetri Terzopoulous. "Snakes: Active Contour Models". *International Journal of Computer Vision*, pages 321–331, 1988.

[10] Y. Lee, D. Terzopoulos, and K. Waters. "Realistic Modeling for Facial Animation". In *Proceedings of SIGGRAPH*, pages 55–62, 1995.

[11] J. Luettin, N. Thacker, and S. Beet. Visual speech recognition using active shape models and hidden markov models. In *ICASSP96*, pages 817–820. IEEE Signal Processing Society, 1996.

[12] John Martin, Alex Pentland, and Ron Kikinis. Shape analysis of brain structures using physical and experimental modes. In *CVPR94*. IEEE Computer Society, 1994.

[13] E.D. Petajan. "Automatic Lipreading to Enhance Speech Recognition". In *Proc. IEEE Communications Society Global Telecom. Conf.*, November 1984.

[14] Henry Stark and John W. Woods. *Probability, Random Processes, and Estimation Theory for Engineers*. Prentice Hall, 1994.

[15] Gilbert Strang. *An Introduction to Applied Mathematics*. Cambridge Press, 1986.

[16] K. Waters and J. Frisbie. "A Coordinated Muscle Model for Speech Animation". In *Graphics Interface*, pages 163–170, 1995.

[17] O.C. Zienkiewicz. *The Finite Element Method in Structural and Continuum Mechanics*. McGraw-Hill, 1967.