January 1998

Generation, Estimation and Tracking of Faces

Douglas M. DeCarlo
University of Pennsylvania

Generation, Estimation and Tracking of Faces

Abstract

This thesis describes techniques for the construction of face models for both computer graphics and computer vision applications. It also details model-based computer vision methods for extracting and combining data with the model. Our face models respect the measurements of populations described by face anthropometry studies. In computer graphics, the anthropometric measurements permit the automatic generation of varied geometric models of human faces. This is accomplished by producing a random set of face measurements generated according to anthropometric statistics. A face fitting these measurements is realized using variational modeling. In computer vision, anthropometric data biases face shape estimation towards more plausible individuals. Having such a detailed model encourages the use of model-based techniques—we use a physics-based deformable model framework. We derive and solve a dynamic system which accounts for edges in the image and incorporates optical flow as a motion constraint on the model. Our solution ensures this constraint remains satisfied when edge information is used, which helps prevent tracking drift. This method is extended using the residuals from the optical flow solution. The extracted structure of the model can be improved by determining small changes in the model that reduce this error residual. We present experiments in extracting the shape and motion of a face from image sequences which exhibit the generality of our technique, as well as provide validation.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-98-36.

GENERATION, ESTIMATION AND TRACKING OF FACES

Douglas M. DeCarlo

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

1998

[illegible]

Dimitris Metaxas
Supervisor of Dissertation

[illegible]

Jean Gallier
Graduate Group Chair

Acknowledgements

Looking back, it’s quite difficult for me to say why this work proceeded in the manner in which it did. But I must say that I feel quite strongly about some of the ideas presented here—this must have come from somewhere. I suppose I can start with my advisor, Dimitris Metaxas, whose unswerving devotion to model-based approaches helped me appreciate how valuable these techniques are. Discussions with Ruzena Bajcsy enlightened me to the true difficulty of the computational vision problem, and also gave me important appreciation for the need for robustness in vision systems.

My undergraduate experience at Carnegie Mellon provided many influences as well, but there are two that stand out for me. Andy Witkin was the first person to get me obsessed about work. Andy has a knack for getting people interested in very tough problems, without letting them become intimidated. Finally, my first experience with research was with Eric Krotkov—it was here that I learned to think problems through, and to be willing to learn the math to do them.

In thinking *this* problem through (and learning this math), discussions I had with Jean Gallier, Eero Simoncelli, Will Welch, Max Mintz, Norm Badler, and Matthew Turk helped me understand what I was actually doing, helped give this work direction, and finally helped make it understandable (hopefully!). There were also many great discussions I had with fellow graduate students at Penn—Catherine Pelachaud, Jeffry Nimeroff, Ken Shoemake, to name a few. The others know who they are. And of course, late nights in the lab would not have been the same without Nick Foster and Jonathan Kaye. It’s quite clear that insanity would have prevailed if not for their antics and distractions.

I would also like to thank the members of my family who showed confidence in me throughout this entire process, and who had to endure weeks without hearing from me during those times when my work habits shifted towards the nocturnal. And finally, to Matthew Stone, who not only provided me with the all-important emotional support, but also helped make myself, and my research, well-tempered.

I’m just glad it’s all over.

Abstract

GENERATION, ESTIMATION AND TRACKING OF FACES

Douglas M. DeCarlo

Supervisor: Dimitris Metaxas

This thesis describes techniques for the construction of face models for both computer graphics and computer vision applications. It also details model-based computer vision methods for extracting and combining data with the model. Our face models respect the measurements of populations described by face anthropometry studies. In computer graphics, the anthropometric measurements permit the automatic generation of varied geometric models of human faces. This is accomplished by producing a random set of face measurements generated according to anthropometric statistics. A face fitting these measurements is realized using variational modeling. In computer vision, anthropometric data biases face shape estimation towards more plausible individuals. Having such a detailed model encourages the use of model-based techniques—we use a physics-based deformable model framework. We derive and solve a dynamic system which accounts for edges in the image and incorporates optical flow as a motion constraint on the model. Our solution ensures this constraint remains satisfied when edge information is used, which helps prevent tracking drift. This method is extended using the residuals from the optical flow solution. The extracted structure of the model can be improved by determining small changes in the model that reduce this error residual. We present experiments in extracting the shape and motion of a face from image sequences which exhibit the generality of our technique, as well as provide validation.

Contents

Chapter 1

Introduction

Appropriate combinations of statistical and constraint-based geometric methods permit modeling and estimation of complex objects. This thesis investigates such combinations in the modeling and estimation of human faces. By concentrating on such a particular class of objects, we show how model-based techniques can exploit existing data and knowledge about facial shape and motion.

Across the human population, the faces of individuals exhibit a great deal of variation in their appearance, but they all still have a good deal of structure in common. A similar statement can be made about facial motion—while it is complex and non-rigid, the motions are still fairly constrained. The work presented here takes advantage of this commonality—information concerning the appearance of faces is either used by or encoded directly into the model. This results in more successful and robust systems.

Aside from being a good testbed for our model-based techniques, faces are interesting on their own, playing an important role in a wide range of applications. Graphical modeling of faces has obvious applications in the entertainment industry for character animation and in simulations involving people. Vision research on face tracking contributes toward the ability to monitor a user's attention and reactions automatically and without intrusion, and thus would have obvious benefits in human-machine interaction. Other applications range from interactive entertainment, to security, to ergonomic studies.

Model specificity

Modeling commonality in computer graphics and computer vision is notoriously difficult. It requires making a decision concerning the trade-off between model specificity and generality. As a result, models capable of representing the shape of an arbitrary face can be categorized based on the restrictiveness of their coverage:

- **Unconstrained coverage.** The face model has enough flexibility (in terms of degrees of freedom) to represent any face, although the model can also represent many other objects that are not faces (a table or a banana). Representation schemes that fall into this category are typically free-form meshes.

- **Constrained coverage.** In addition to having the flexibility to represent any face, the model has a bias towards representing actual faces, but can still represent objects that are not faces. A good example of a representation in this category is a model based on principal component analysis (PCA), which is a technique based on statistical analysis of examples. The manually constructed face model used in this thesis for shape estimation (described in Chapter 4) also falls in this category.

- **Generative.** For this most restrictive category, models can represent any face, but include a probability distribution on the faces represented (as to the likelihood of each represented face). This dissertation describes the first generative model for faces in Chapter 3.

The choice of which class is appropriate is largely application dependent. The state-of-the-art models used for face tracking tend to be models with constrained coverage models, although unconstrained models are sometimes used as well.

Model-based techniques

Using detailed models raises distinctive challenges and opportunities for computer vision. In the absence of knowledge about the objects being observed, vision techniques often will extract information for each image pixel. Improvements to this individual pixel view simply exploit the spatial or temporal coherency expected in most situations. Of course, such assumptions carry with them the added complexity of segmentation—finding the boundaries of coherent regions in space and time [NH87]. Meanwhile, in the presence of knowledge concerning the observed objects, such as membership in a particular class of objects, *model-based* vision techniques extract information for each degree of freedom of a particular model.

The use of model-based techniques introduce a new set of problems, however. The most significant of these is the problem of maintaining alignment of the model with the image. Even though an accurate observation model might be available, should its current state not correspond with the current state of the observed object, the advantage of using a model is lost (and its use may even be detrimental). Another difficulty arises from potential interdependencies between parameters, and possible ambiguity between two model configurations (which have similar appearances, but different parameter values). This becomes more important to address as model complexity increases. As a result, information extracted in a model-based framework must be used carefully, and combined in a way that respects these interdependencies.

The model-based techniques described in this dissertation are applied to faces (but can be applied more generally). Face tracking is a particularly natural testbed for our research for two reasons. The actual shape and motion of faces makes edge and optical flow information easy to use and advantageous to combine; and the abundance of data describing human face shape [Far94] facilitates the development of three-dimensional models of faces with separable shape and motion parameterizations.

Constraints and Modeling

The main techniques in this thesis each use some form of constraints on the model to achieve the desired goal. The following is a brief description of the different uses of constraints used in this thesis.

Variational modeling techniques use *geometric constraints*: these constraints restrict the coverage of a model (either shape or motion) by limiting the space of acceptable parameter combinations. Typically, solving for parameters which satisfy geometric constraints involves some form of constrained optimization. Existing uses of these techniques range from surface design to the modeling and estimation of articulated rigid motion. In this document, geometric constraints are used to construct a generative face model—where the coverage of the model is limited by geometric constraints derived from anthropometric data.

The model-based tracking technique described in this thesis uses a *data-based constraint* on the estimated parameters as a means of combining information. In other words, the optimization problem that is solved to determine the current estimate is converted into a constrained optimization problem, where the constraint is used to disambiguate the solution. Using one source of data to constrain the solutions from another source can help when the optimization problem involved in the unconstrained solution is difficult. For example, we use optical flow information to constrain a template alignment problem (based on edges). By limiting the choices available to the alignment, many of the local minima of this problem are avoided.

Finally, we can use a *model constraint*, which is the implicit constraint a model places by restricting possible configurations to those given by the model. The method of regularization in computer vision is probably the best known example of using a model constraint. For the work here, we use a model constraint to assess and improve the accuracy of the current estimate. Later, we present how we can improve the shape estimate by making small adaptations which force the motion observations to agree with the data.

1.1 Contributions

This dissertation describes face model construction processes, as well as techniques for the use and combination of different sources of model-based information for estimation and tracking. For model construction, knowledge of facial geometry comes in the form of measurements from *face anthropometry* [Far94, KS96], the science dedicated to the measurement of the human face. Anthropometric studies such as [Far87, Far94] provide data on the shape of faces which help characterize the distinctive features of faces from a particular population. This represents the first significant use of data from face anthropometry studies in both graphics and vision.

In computer graphics, we present a system which is capable of generating distinct and plausible face geometries automatically. The system generates a set of random facial measurements from statistics gathered from face anthropometry studies [Far87, Far94]. Armed with a set of measurements, variational modeling techniques are used to construct a face geometry that realizes the measurements. Variational modeling is a surface modeling tool that employs constrained optimization methods to find the fairest surface that satisfies a set of geometric constraints. In this case, a fair surface is one that minimizes bending away from a prototypical face, and is subjected to a set of geometric constraints that are an abstraction of the measurements performed by anthropometrists.

Face model construction in computer vision is a considerably different task, where simplicity in parameterization takes precedence over appearance. The generative model described above is not appropriate for use by a vision system. Instead, the parameterization of our model is constructed by hand—a series of localized deformations are specified that allow for shape variations observed in anthropometry studies [Far94]. During shape estimation, the data from these studies is used to bias the model towards more likely individuals, by minimizing deviation from expected values of anthropometric measurements. Since motion tracking is also a goal of this vision system, a motion parameterization is also constructed (by hand) for a small set of facial expressions.

The shape and motion estimation of model parameters is realized using a deformable model framework [Met96]. This framework uses a parameterized face model, which has parameters for both the shape of the face (the unchanging appearance of an individual) as well as its motion (facial expressions and displays). Shape estimation is performed using edges found in the image, guided by knowledge of where edges are likely to occur.

The apparent motion of brightness patterns in an image—the optical flow—provides a constraint on the motion of a deformable model. We derive and solve a system that incorporates this constraint, which is then used for model tracking. The solution contains a familiar term found in other model-based optical flow tracking work. However, it contains an additional term which maintains the optical flow constraint in the presence of other data (in this case, edges). The use of this constraint enforcement term greatly improves the robustness of the tracking results by producing a motion estimate that is consistent with all observed data. This fruitful combination of optical flow information with edges combats error accumulation in model tracking.

The model-based optical flow solution also provides an estimate of how much error is present (a residual). We present a model-based technique which improves the shape and motion estimate by minimizing this residual. The use of the model gives meaning to the error estimate, which describes how the observed motion deviates from what the model can represent.

1.2 Overview

This document proceeds as follows. Chapter 2 provides background information on the use of a deformable model framework for computer vision, and a review of model-based techniques using such a framework. Also included is a summary of Farkas's system of anthropometric face measurements, and a description of variational modeling.

The next two chapters describe the construction of face models for graphics and vision, and how data from anthropometry studies can be used in their formation. Chapter 3 describes our generative face model. It describes our method for generating sets of measurements consistent with population groups. This leads to a discussion of how variational modeling can be used to produce face geometries that realize a set of generated measurements. In Chapter 4, a model suitable for shape and motion estimation is described. In contrast to the generative face model, this face is geared more towards vision. While its parameterization is hand constructed, it still relies on face anthropometry data to maintain a consistent geometry.

The face model developed in Chapter 4 is used in a deformable model framework for the shape and motion estimation of human subjects, which is the subject of Chapter 5. The optical flow constraint equation [Hor86] is reformulated as a constraint on the motion of the deformable model. This constrained system is solved to produce a model-based optical flow solution which allows for the addition of other data sources (in this case, edge data), so that the optical flow constraint is maintained. Kalman filtering is then used to allow small violations in this optical flow constraint, given that the optical flow measurements are noisy. Chapter 6 then describes a technique which can be used to improve the shape and motion estimates by reducing the error residuals from the optical flow constraint solution. A series of vision experiments using this framework are then presented, which exhibit the generality, as well as validate the accuracy of these techniques. Finally, Chapter 8 summarizes the contributions of this work, along with a discussion of future work possibilities.

Chapter 2

Background

This chapter reviews a variety of topics that are touched upon elsewhere in this document. The focus of this dissertation is on modeling in graphics and vision. Central to both of these areas is the issue of representation: how to specify shape and motion for a particular class of objects. More specific to computer vision is the process of estimation, whose goal is to minimize the deviation between the model and data. For the applications here, we use physical simulation as an analogy for optimization, which permits a powerful set of existing techniques to be borrowed from physics. It also couples these with existing geometric techniques from vision. This is often called a *physics-based* framework, or a *deformable model* framework [MT93, PS91, TWK88], and is the subject of Section 2.1. This is followed by descriptions of model-based shape estimation using edges and model-based motion estimation using optical flow information in Section 2.2.1. Later in Chapter 5, these techniques will be combined together in a deformable model framework.

These modeling techniques say little about the actual construction of a model. This process can become quite difficult, especially for face models. Part of this difficulty comes from the complex variability in measurements seen in human faces and face models. This variability has been systematically studied in the field of *anthropometry*.

Section 2.3 provides a brief review of anthropometry—the biological science of human body measurement. The procedures for measurement in anthropometry are precisely specified, allowing data between individuals to be successfully compared, and for useful statistics of population groups to be derived. Later in this document, our approaches rely on this large body of existing data that describes the shapes of people's faces. In graphics, it will allow for the automatic generation of varied face geometries. In vision, it provides information about the shapes of faces which is used to bias the estimation process towards more likely occurring individuals.

The use of anthropometric data for graphics model generation is described in Chapter 3. This data is supplied as input to modeling techniques which allow the low-level parameters of a shape to be determined indirectly. In particular, *variational modeling* techniques are used, which is the subject of Section 2.4. We use variational modeling to obtain a surface that conforms to a set of anthropometric measurements while retaining characteristics that all faces share.

2.1 Deformable models for computer vision

Deformable models [MT93, PS91, TWK88] are parameterized shapes that deform due to forces according to physical laws. For vision applications, physics provides a useful analogy for treating shape estimation [MT93], where forces are determined from visual cues such as edges in an image. The deformations that result produce a shape that agrees with the data.

The use of physics also makes available additional mathematical tools. For example, smooth surfaces that interpolate a set of sparse data can be determined by associating an energy with the surface (which is minimized) [TWK87], and produces a method of regularization useful as a data fitting technique. Constraint techniques from physics have been used to form articulated rigid models [MT93], and will be used in Chapter 5 to incorporate optical flow information. The next section describes how a model is specified and represented in a deformable model framework.

2.1.1 Model formulation

The shape of the deformable model $\mathbf{x}$ is parameterized by a vector of values $\mathbf{q}$ (sometimes called generalized coordinates) and is defined over a domain Ω which can be used to identify specific points on the model; a particular point on the model is written as $\mathbf{x}(\mathbf{q};\mathbf{u})$ with $\mathbf{u} \in \Omega$, although the dependency of $\mathbf{x}$ on $\mathbf{q}$ is often omitted.

The model $\mathbf{x}$ is formed by applying a deformation function to an underlying shape $\mathbf{s}$ (which has parameters $\mathbf{q_s}$). An example shape primitive is the ellipsoid:

$$\mathbf{s}_{\text{ellip}}\big((a_x, a_y, a_z); (u,v)\big) = \begin{pmatrix} a_x \cos u \cos v \\ a_y \cos u \sin v \\ a_z \sin u \end{pmatrix} \qquad (2.1)$$

$$\text{where } \Omega = \left\{ (u,v) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times [0, 2\pi) \right\}$$

which has parameters a_x, a_y and a_z for scaling in the x, y and z directions, respectively. While the shape of this model is limited to ellipsoids, its coverage can be extended by applying deformations. The deformation function $\mathbf{T} : \mathbb{R}^q \times \mathbb{R}^3 \mapsto \mathbb{R}^3$ deforms the underlying shape based on the q deformation parameters in $\mathbf{q_T}$, so that:

$$\mathbf{x}(\mathbf{q};\mathbf{u}) = \mathbf{T}\left(\mathbf{q_T};\ \mathbf{s}(\mathbf{q_s};\mathbf{u})\right) \qquad (2.2)$$

Here, $\mathbf{T}$ is defined as a composed sequence of deformation functions (such as bending or scaling deformations) [MT93]. To allow for a more streamlined discussion here, it can also contain rigid motions (translations and rotations), for example:

$$\mathbf{T}_{\text{rigid}}\left((\mathbf{c}^\top, \theta^\top)^\top; \mathbf{x}\right) = \mathbf{c} + \mathbf{R}\mathbf{x} \qquad (2.3)$$

where $\mathbf{c}$ specifies the translation vector, and $\mathbf{R}$ is a rotation matrix given by the quaternion θ.

The kinematics of the model can be determined in terms of the parameter velocities $\dot{\mathbf{q}}$. As the shape changes, the velocity at a point $\mathbf{u}$ on the model is given by:

$$\dot{\mathbf{x}}(\mathbf{u}) = \mathbf{L}(\mathbf{q};\mathbf{u})\dot{\mathbf{q}} \qquad (2.4)$$

where $\mathbf{L} = \partial\mathbf{x}/\partial\mathbf{q}$ is the model Jacobian [Met96]. Note that the dependency of $\mathbf{L}$ on $\mathbf{q}$ is not always written, for reasons of conciseness. For cases where $\mathbf{x}$ is defined using a sequence of deformation functions, the Jacobian can be computed using the chain rule as in Appendix A.

A good geometric intuition for $\mathbf{L}(\mathbf{u})$ is obtained by noting that each column of $\mathbf{L}$ corresponds to a particular parameter in $\mathbf{q}$, and is a three-dimensional vector which "points" in the direction that $\mathbf{x}(\mathbf{u})$ moves as that parameter is increased—of course, it is only a linear approximation to the actual motion.

There really isn't anything special about the models used here. Basically, any explicitly parameterized model will work. For model-based applications, the construction of the parameterization often captures the geometric structure for the class of objects being modeled, whether constructed automatically or by hand, so it is the *choice* of what model to use that is important. How to choose an appropriate model is largely an engineering decision.

2.1.2 Perspective projection of the model

When modeling an object viewed in images, $\mathbf{x}$ needs to include a camera projection, resulting in a two-dimensional model (called $\mathbf{x}_p$), which is projected flat from the original three-dimensional model. Under perspective projection (with a camera having focal length f), the point $\mathbf{x}(\mathbf{u}) = (x,y,z)^\top$ projects to the image point $\mathbf{x}_p(\mathbf{u}) = \frac{f}{z}(x,y)^\top$.

The velocities of model points projected onto the image plane, $\dot{\mathbf{x}}_p$, can be found in terms of $\dot{\mathbf{x}}$. The Jacobian $\mathbf{L}_p = \partial\mathbf{x}_p/\partial\mathbf{q}$ is given by:

$$\dot{\mathbf{x}}_p(\mathbf{u}) = \frac{\partial\mathbf{x}_p}{\partial\mathbf{x}}\dot{\mathbf{x}}(\mathbf{u}) = \left(\frac{\partial\mathbf{x}_p}{\partial\mathbf{x}}\mathbf{L}(\mathbf{q};\mathbf{u})\right)\dot{\mathbf{q}} = \mathbf{L}_p(\mathbf{q};\mathbf{u})\dot{\mathbf{q}} \tag{2.5}$$

where

$$\frac{\partial\mathbf{x}_p}{\partial\mathbf{x}} = \begin{bmatrix} f/z & 0 & -fx/z^2 \\ 0 & f/z & -fy/z^2 \end{bmatrix} \tag{2.6}$$

The matrix in (2.6) projects the columns of $\mathbf{L}$ (which are three-dimensional vectors) onto the image plane.

2.1.3 Estimation using dynamics

The models defined earlier become useful for applications such as shape and motion estimation when used in a physics-based framework [Met96]. These techniques are a form of optimization whereby the deviation between the model and the data is minimized. The optimization is performed by integrating differential equations derived from the Euler-Lagrange equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{f}_{\mathbf{q}} \tag{2.7}$$

where $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ are mass, damping and stiffness matrices, respectively, and $\mathbf{f}_{\mathbf{q}}$ are generalized forces, derived from data, and applied to the model.

When used as an optimization tool, the full generality of these equations is not needed. Simplification can provide a more efficient and stable solution to the optimization—the mass matrix $\mathbf{M}$ is often zeroed in estimation applications, since model inertia can produce oscillations around the desired minimum. This simplification also has the desirable property that the model state no longer changes once all forces vanish or equilibrate. The damping matrix $\mathbf{D}$ specifies how energy is dissipated, and is typically simplified to be diagonal (or the identity), to allow for fast solution. However, for situations where there is a fairly significant interdependency between the parameters in $\mathbf{q}$, the damping matrix (as $\mathbf{D} = \int \mathbf{L}^\top \mathbf{L}$) [Met96] can alleviate this problem. Although inverting $\mathbf{D}$ then becomes necessary for the solution of the dynamic system. The use of the stiffness matrix $\mathbf{K}$ is associated with the quadratic strain energy $\frac{1}{2}\mathbf{q}^\top \mathbf{K}\mathbf{q}$, and provides a measure of "fairness" of the model (preferred surfaces minimize bending energy), allowing for reasonable solutions in situations where the data is sparse (relative to the number of model parameters) or particularly noisy.

For the applications here, the mass term is omitted, and the damping is set to be the identity. Additionally, no stiffness term is used since the models used here have a fairly small set of parameters. This results in the following simplified dynamic equations of motion:

$$\dot{\mathbf{q}} = \mathbf{f}_{\mathbf{q}} \tag{2.8}$$

where the applied forces $\mathbf{f}_{\mathbf{q}}$ are computed from three-dimensional forces $\mathbf{f}_{\text{3D}}$ and two-dimensional image forces $\mathbf{f}_{\text{image}}$ as:

$$\begin{aligned}\mathbf{f}_{\mathbf{q}} &= \int \Big(\mathbf{L}(\mathbf{u})^{\top}\mathbf{f}_{\text{3D}}(\mathbf{u}) + \mathbf{L}_p(\mathbf{u})^{\top}\mathbf{f}_{\text{image}}(\mathbf{u})\Big)\, d\mathbf{u} \\ &\cong \sum_j \Big(\mathbf{L}(\mathbf{u}_j)^{\top}\mathbf{f}_{\text{3D}}(\mathbf{u}_j) + \mathbf{L}_p(\mathbf{u}_j)^{\top}\mathbf{f}_{\text{image}}(\mathbf{u}_j)\Big)\end{aligned} \tag{2.9}$$

The distribution of forces on the model is based in part on forces computed from the edges of an input image [Met96]. Using $\mathbf{L}$ and $\mathbf{L}_p$, the applied forces are converted to forces which act on $\mathbf{q}$ and are integrated over the model to find the total parameter force $\mathbf{f}_{\mathbf{q}}$. The dynamic system in (2.8) is solved by integrating over time, using standard (explicit) differential equation integration techniques. Euler integration is used in [Met96].

2.2 Model-based estimation

The use of parameterized models, such as those introduced in Section 2.1, suggest a model-based approach to estimation. Instead of extracting information per image pixel, or per node (in a mesh), a model-based approach extracts information for each degree of freedom in the model parameterization—for each model parameter (typically there are far fewer parameters in the model than are needed to represent arbitrary shapes). This section summarizes previous approaches for model-based shape estimation using edge information and model-based optical flow computation.

2.2.1 Model-based shape estimation

The model-based extraction of shape using image edge information can be accomplished using the physics-based framework described in Section 2.1. All that is needed is a method for determining forces from image data. With that, and given an adequate model initialization, these techniques will align features on the model with image features, determining object pose and shape parameters. The remainder of this section describes current approaches for data-force computation.

There are two basic approaches to this problem in this framework—both assign forces derived from image features that are applied to particular model locations. The two issues to address here are force determination (given the data, determine an appropriate force), and force assignment (what model locations should be affected by this force). Upon each solution iteration, these forces are determined again, and re-assigned. A successful fit has the model-data alignment improving (and converging) over a series of iterations.

The first approach involves determining a force distribution designed to "attract" the model towards regions with significant image intensity gradient. Given the image I, the resulting two-dimensional potential field at image location (x,y) is given by [TWK88]:

$$P(x,y) = -\,\|\nabla\,(G_\sigma * \mathrm{I})\,(x,y)\| \tag{2.10}$$

where the image I is blurred by convolution $(*)$ with the Gaussian G_σ of radius σ, and as a result produces a larger area of influence in the image, permitting a greater deviation between the actual model position from its initialized position. The image processing steps are shown in the first three frames of Figure 2.1. This potential results in the following force distribution on the image, with weighting factor β:

$$\mathbf{f}(\mathbf{u}_{\text{contour}}) = -\beta \cdot \nabla P(x,y) \tag{2.11}$$

where $\mathbf{u}_{\text{contour}}$ is the point on the model that projects to the image position (x,y), and is nearby (measured along the surface) an occluding contour of the observed object [Met96] (if there is no nearby occluding contour, this force is not applied). This is a simple three-dimensional analog of using this image potential for two-dimensional "snakes" [TWK88]. The resulting force distribution from this potential for the example in Figure 2.1 is shown in the rightmost frame, where it can be seen how the field would tend to draw the model into alignment with nearby image features.

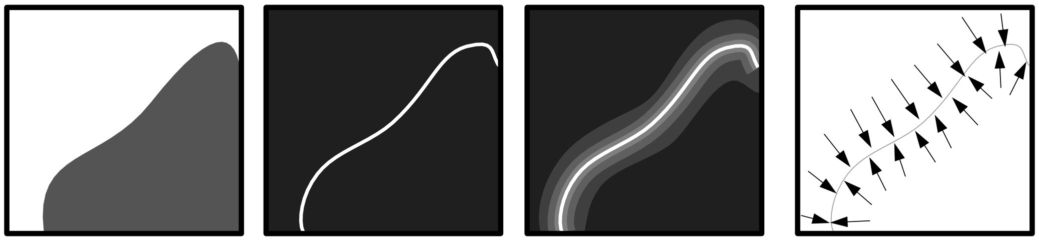


Figure 2.1: Force distribution derived from image features

An alternative method of force determination is possible when a more detailed model is used. With more detail, comes knowledge of where image edges are likely to occur—edges are caused by the presence of occlusion boundaries and highly curved regions on the model surface. In this case, instead of producing an image gradient force field, forces are applied directly to the closest edge-producing point on the model from each edge pixel. This is the reverse matching problem: finding a mapping from model features to image edges (as opposed to a mapping from image edges to features).

Chan and Metaxas [CMD94, Met96] determine the image edges using a qualitative shape recovery process [DPR92], which extracts sets of pixel coordinates E_i for each identified edge segment i. Then, the set of edge-producing model locations $M_i \subset \Omega$ (corresponding to the edge set E_i) is determined using occluding contours and surface creases. Correspondences between pixels in E_i and model points in M_i are determined by a proximity based assignment between the model locations and each image pixel $\mathbf{e} \in E_i$: $\mathbf{u}_m \in M_i$ is the point on the model that is closest to $\mathbf{e}$ in the image. This choice results in the following long-range forces (weighted by β):

$$\mathbf{f}(\mathbf{u}_m) = \beta \cdot (\mathbf{e} - \mathbf{x}_p(\mathbf{u}_m)) \tag{2.12}$$

As seen in Figure 2.2, forces are assigned to feature-producing model locations to the nearest edge feature.

While the first of the techniques listed here can be used for any model (even a free-form mesh), the second method requires a fairly detailed model, so that the predicted edge locations are fairly accurate. However, by using long range forces, the model can be much

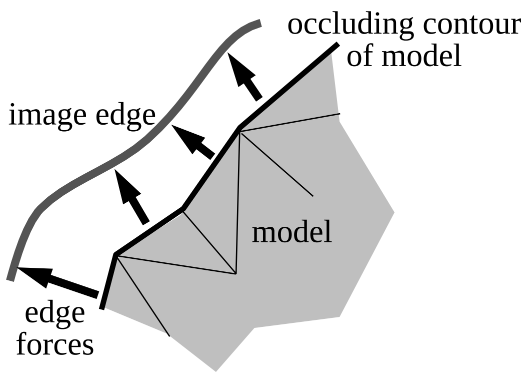


Figure 2.2: Force assignments derived from image features and predicted model locations

farther from initialization position using the second technique, showing the advantage of a having a detailed model.

2.2.2 Model-based optical flow

Optical flow information, which describes the apparent motion of brightness patterns in an image, is often used for object tracking in vision. Direct use of this information often requires assumptions about the objects being viewed. Most common, is the assumption that particular locations on viewed objects do not change brightness. This *brightness constancy* assumption allows the formulation of the well-known optical flow constraint equation [Hor86] for the image I (the assumption manifests itself as the zero on the right-hand-side):

$$\nabla \mathrm{I} \begin{bmatrix} u \\ v \end{bmatrix} + \mathrm{I}_t = 0 \tag{2.13}$$

where $\nabla \mathrm{I} = [\mathrm{I}_x \;\; \mathrm{I}_y]$ are the spatial derivatives and I_t is the temporal derivative of the image intensity. u and v are the components of the image velocities. The following is a brief discussion of how a model-based approach reformulates (2.13) in terms of the model parameters $\mathbf{q}$, which replace the image velocities. For consistency, this discussion will use the notation described in Section 2.1, which is different (and more compact) from that used in previous model-based optical flow work.

For a model under perspective projection, there exists a unique point **u** on the model that corresponds to each pixel (provided it is not on an occluding boundary). The crucial observation is that in a model-based approach, u and v are *identified* with the components of the projected model velocities $\dot{\mathbf{x}}_p(\mathbf{u})$:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \dot{\mathbf{x}}_p(\mathbf{u}) = \mathbf{L}_p(\mathbf{u})\dot{\mathbf{q}} \tag{2.14}$$

The model-based optical flow constraint equation in the image can be found by rewriting (2.13) using (2.14):

$$\nabla \mathrm{I}\mathbf{L}_p(\mathbf{u})\dot{\mathbf{q}} + \mathrm{I}_t = 0 \tag{2.15}$$

Formulations which are basically identical to (2.15) (although are often confined to rigid motion) can be found in [Adi85, BAHH92, CAHT94, HW88, LRF93, NH87, NS85]. Negahdaripour and Horn [NH87] refers to a formulation such as this as a *direct method* for motion estimation. The discussion of (2.15) in [BAHH92, NH87, NS85] is specialized for rigid motion, and while still general, requires a lengthy derivation by hand. Using the modular shape formulation described in Section 2.1 allows for more simple derivations of (2.15), and is more similar to the description in [CAHT94, LRF93]. (Another difference between these techniques is noted by their use of either Euler angles or quaternions as the representation for the rotations).

There are a number of techniques available for solving (2.15). The most common is the iterative minimization of the quadratic error measure, summed over a set of pixels:

$$\min_{\dot{\mathbf{q}}} \sum_i (\nabla \mathrm{I}_i \mathbf{L}_p(\mathbf{u}_i)\dot{\mathbf{q}} + \mathrm{I}_{t_i})^2 \tag{2.16}$$

which is the approach taken in [BAHH92, HW88, NH87, NS85]. An alternative method solves the least-squares problem using the pseudo-inverse of the matrix formed by stacking a set of equations like (2.15) for a set of pixels [CAHT94, NS85, LRF93]:

$$\mathbf{B}\dot{\mathbf{q}} + \mathbf{I}_\mathrm{t} = 0 \tag{2.17}$$

which is solved as:

$$\dot{\mathbf{q}} = -\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}} \tag{2.18}$$

where $\mathbf{B}^{+}$ is the pseudo-inverse of $\mathbf{B}$ [Str88]. This approach is basically a one-step version of the iterative approach above. This version linearizes by assuming $\mathbf{L}_p$ is constant for the entire time step (instead, with the above iterative solution, $\mathbf{L}_p$ is re-evaluated at each iteration step per time step).

Once a solution is obtained, image warping techniques can be used to improve the solution [BAHH92], which can to some degree correct for the linearization performed (such as in the formation of $\mathbf{L}_p$), or for the determination of large motions (using a coarse-to-fine strategy). Given the solution, the current image is warped to “undo” the current flow estimate, allowing a more detailed estimate to be obtained as increments to the original solution, using the warped image.

There are a number of benefits obtained when using a model-based optical flow formulation in place of an image-based method (should the application permit their use). By restricting the extracted motion to a particular motion parameterization, the problem of flow field determination is no longer underconstrained[1]. Image-based techniques require the presence of smoothness conditions to determine a solution, and even worse, may require a motion segmentation to determine the boundaries of where the smoothing is performed. Model-based techniques are able to extract flow information, even when the useful information is sparse, and do not need to impose any smoothness constraints to determine a solution (since they are implicit in the model).

Without the use of position information, however, the tracking solution will drift; solving (2.15) over a sequence of frames involves integrating a velocity. Chapter 5 will describe a technique for combining model-based optical flow solutions with edge information, and therefore prevents this tracking error accumulation.

[1]This assumes, of course, that the motion parameterization of the model does not have an extremely large set of motion parameters. This would preclude the use of model-based techniques over image-based techniques.

Aside from tracking, it is also possible to use a model-based optical flow formulation to estimate the model structure. In particular, Koch [Koc93] describes a model-based framework which uses optical flow information to estimate the rigid translation and rotation of a moving face, and adapts the shape of the face to account for the motion discrepancy. Chapter 6 presents an alternative method of structure estimation from optical flow information.

2.3 Face anthropometry

Anthropometry is the biological science of human body measurement. Anthropometric data informs a range of enterprises that depend on knowledge of the distribution of measurements across human populations. For example, in human-factors analysis, a known range for human measurements can help guide the design of products to fit most people [Doo82]; in medicine, quantitative comparison of anthropometric data with patients' measurements before and after surgery furthers planning and assessment of plastic and reconstructive surgery [Far94]; in forensic anthropology, conjectures about likely measurements, derived from anthropometry, figure in the determination of individuals' appearance from their remains [Rog84, Far94]; and in the recovery of missing children, by aging their appearance taken from photographs [Far94]. The use of anthropometry data in this dissertation describes a similar use of anthropometry in the construction of face models for computer graphics and computer vision applications.

In order to develop useful statistics from anthropometric measurements, the measurements are made in a strictly defined way [Hrd72]. The rest of this section outlines one popular regime of such measurements and the information available from analyses of the resulting data.

Anthropometric evaluation begins with the identification of particular locations on a subject, called *landmark* points, defined in terms of visible or palpable features (skin or bone) on the subject. A series of measurements between these landmarks is then taken using carefully specified procedures and measuring instruments (such as calipers, levels and measuring tape). As a result, repeated measurements of the same individual (taken a few days apart) are very reliable, and measurements of different individuals can be successfully compared.

Farkas [Far94] describes a widely used set of measurements for describing the human face. A large amount of anthropometric data using this system is available [Far87, Far94]. The system uses a total of 47 landmark points to describe the face; Figure 2.3 illustrates many of them. The landmarks are typically identified by abbreviations of corresponding anatomical terms. For example, the inner corner of the eye is *en* for *endocanthion*, while the top of the flap of cartilage (the tragus) in front of the ear is *t* for *tragion*.

Five of the landmarks determine a canonical coordinate system for the head. The horizontal plane is determined by the two lines (on either side of the head) connecting the landmark *t* to the landmark *or* (for *orbitale*), the lowest point of the eye socket on the skull. The vertical mid-line axis is defined by the landmarks *n* (for *nasion*), a skull feature roughly between the eyebrows; *sn* (for *subnasale*) the center point where the nose meets the upper lip; and *gn* (for *gnathion*), the lowest point on the chin. In measurement, anthropometrists actually align the head to this horizontal and vertical, in what is known as Frankfurt horizontal (FH) position [Far94, KS96], so that measurements can be made easily and accurately with respect to this coordinate system.

Farkas's inventory includes the five types of facial measurements described below and illustrated in Figure 2.4:

- the *shortest distance* between two landmarks. An example is *en-ex*, the distance between the landmarks at the corners of the eye
- the *axial distance* between two landmarks—the distance measured along one of the axes of the canonical coordinate system, with the head in FH position. An example is *v-tr*, the vertical distance (height difference) between the top of the head (*v* for *vertex*) and hairline (*tr* for *trichion*).

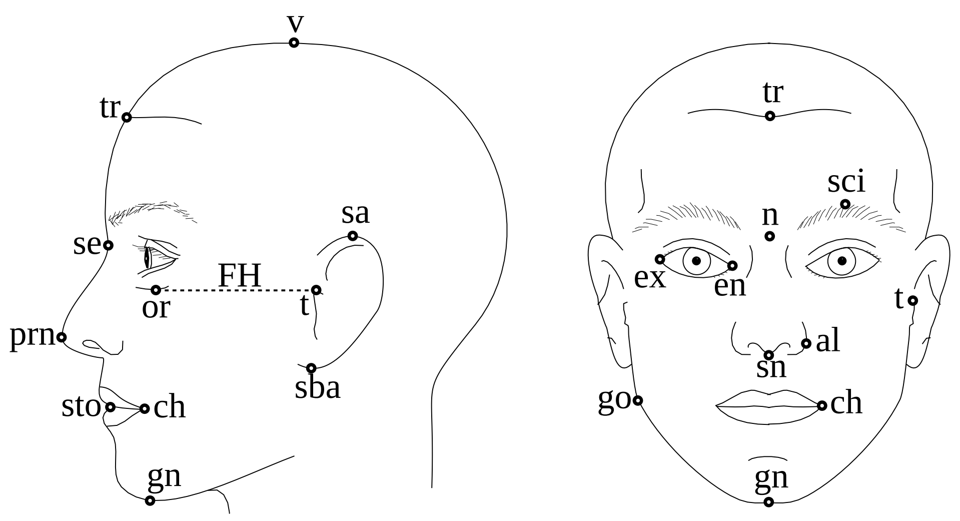


Figure 2.3: Anthropometric landmarks on the face [Far94]

- the *tangential distance* between two landmarks—the distance measured along a prescribed path on the surface of the face. An example is *ch*-*t*, the surface distance from the corner of the mouth (*ch* for *cheilion*) to the tragus.
- the *angle of inclination* between two landmarks with respect to one of the canonical axes. An example is the inclination of the ear axis with respect to the vertical.
- the *angle between locations*, such as the mentocervical angle (the angle at the chin).

Farkas describes a total of 132 measurements on the face and head. Some of the measurements are *paired*, when there is a corresponding measurement on the left and right side of the face. Until recently, the measurement process could only be carried out by experienced anthropometrists by hand. However, recent work has investigated 3-D range scanners as an alternative to manual measurement [BA96, Far94, KS96].

Systematic collection of anthropometric measurements has made possible a variety of statistical investigations of groups of subjects. Subjects have been grouped on the basis of gender, race, age, "attractiveness" or the presence of a physical syndrome. Means and variances for the measurements within a group, tabulated in [Far94, Gor89], effectively provide a set of measurements which captures virtually all of the variation that can occur in the group.

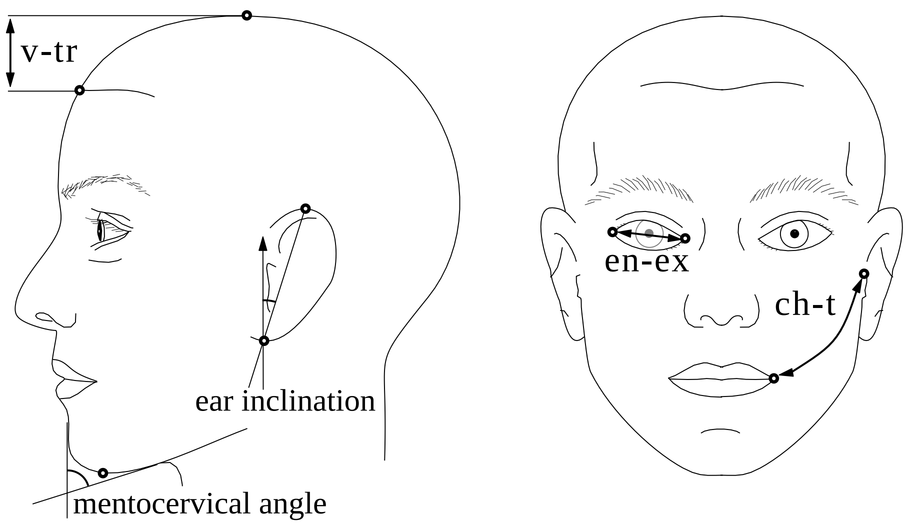


Figure 2.4: Example anthropometric measurements [Far94]

In addition to statistics on measurements, statistics on the *proportions* between measurements have also been derived. The description of the human form by proportions goes back to Dürer and da Vinci; anthropometrists have found that proportions give useful information about the correlations between features, and can serve as more reliable indicators of group membership than can simple measurements [Far87]. Many facial proportions have been found to show statistically significant differences across population groups [Hrd72]. These proportions are averaged over a particular population group, and means and variances are provided in [Far87]. An example proportion is shown in Figure 2.5, which states that the width of the mouth *ch-ch* is roughly three-halves the size of the width of the nose (at the base) *al-al*.

Later, in Chapter 3, Farkas's anthropometry is applied to the generation of distinct, plausible face geometries. Face anthropometry data is also used to bias a face model towards more likely individuals during shape estimation in Chapter 4. Both of these applications involve the representation of anthropometric measurements, in order to apply Farkas's anthropometry. Additional techniques are developed in Chapter 3 to deal with the fact that only limited information is provided by these sources (means and variances for measurements and proportions). Of course, with more detailed information, such as

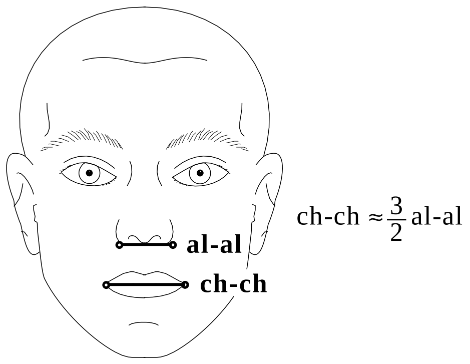


Figure 2.5: Example anthropometric proportion

measurement and proportion *co*variance data, as well as fitted distributions (instead of assuming Gaussian), different methods would be used.

2.4 Variational modeling

Traditional work on surface modeling provides the designer with "handles" for modifying the shape, which are directly related to the underlying representation. In an attempt to move away from this paradigm, work on *variational shape design* attempts to provide a more abstract level of control over the shape to the designer, such as "construct a smooth surface which passes through these points and contains this curve" [CG91, GC95, HKD93, MS92, TQ94, WW92, WW94].

This section provides a brief overview of variational modeling. Later in this document (in Chapter 3), the underlying method used for the automatic generation of varied face geometries will draw on the techniques presented here, using anthropometric measurements (described in Section 2.3) as a means of abstract specification of shape.

Variational modeling allows the specification of shape to be separated from the representation of shape. This abstraction is realized using standard optimization techniques, where the desired shape is the solution to a problem rooted in the calculus of variations.

Instead of directly manipulating the underlying representation, the designer supplies *constraints* on the desired shape, which are taken into account during the optimization. Typically these constraints only determine a small number of the degrees of freedom necessary to describe the entire shape—the remaining degrees of freedom are determined by minimizing an *objective function* which specifies the *fairness* of the shape. While fair is often interpreted as “visually pleasing,” its use here can be application dependent. For a general surface, these optimization problems have no closed-form solutions. To make the problem tractable, a shape representation method is chosen from the Computer Aided Geometric Design (CAGD) literature which confines the resulting surface to be of a certain class. Furthermore, the objective function is often approximated and discretized to allow its efficient solution.

Use of a wide variety of constraint types and objective functions can be found in the variational modeling literature. The presentation here, however, will focus on the framework in [WW92]. The next few sections contain discussions of the surface representations used in variational modeling frameworks, how constraints are specified for these surfaces, and how these surfaces are faired in the presence of these constraints.

2.4.1 Linear surface representations

As stated earlier, the surface representation schemes used in variational modeling do not allow for arbitrary surfaces to be specified. Instead, the representation methods define a surface in terms of a finite number of degrees of freedom (the control points)–this is known as the finite-element method [Zie77]. If the representation scheme is flexible enough, the resulting surfaces can closely approximate the true solution to the variational problem.

Parametric surface modeling schemes map $\mathbb{R}^2$ to surfaces in $\mathbb{R}^3$. The parametric surface $\mathbf{s}$ is defined over a two-dimensional domain with parameters u and v using the differentiable functions $x(u,v)$, $y(u,v)$ and $z(u,v)$:

$$\mathbf{s}(u,v) = \Big(x(u,v), y(u,v), z(u,v)\Big) \quad u,v \in [a,b] \tag{2.19}$$

where $\|\mathbf{s}_u(u,v) \times \mathbf{s}_v(u,v)\| \neq 0$ for all (u,v). $\mathbf{s}_u$ and $\mathbf{s}_v$ denote parametric derivatives in the u and v directions, respectively.

Variational modeling frameworks typically use a surface representation scheme for $\mathbf{s}(u,v)$ which is a *linear* combination of control points $\mathbf{p}$ weighted by a set of basis functions b:

$$\mathbf{s}(u,v) = \sum_{i \in I} b_i(u,v)\mathbf{p}_i \tag{2.20}$$

Since any point $\mathbf{s}(u,v)$ on the surface $\mathbf{s}$ depends *linearly* on its degrees of freedom (the control points $\mathbf{p}$), finding solution of variational problems can be made quite efficient. In this document, the particular surface representation scheme used will be B-splines [Far93], which have a rectangular domain (such as $[0,1] \times [0,1]$).

2.4.2 Fairing

Measures of surface fairness are formulated as a local measure, and are integrated over the entire shape. The integral of these objective functions are a single positive value, evaluating to zero for the fairest shape possible. A fairing process will minimize this functional, in the presence of constraints. Most variational modeling systems use quadratic objective functions (for efficiency reasons), which require approximation and linearization.

The most prevalent surface objective function is the *thin plate* functional, which approximates the bending energy in a thin elastic sheet—it is defined in terms of surface curvatures. Surface curvature (measured at a particular surface point, in a particular tangent direction) measures the rate of change of tangent inclination. A basic result from differential geometry tells us that curvature is a smoothly varying function which takes on maximum and minimum values κ_1 and κ_2 (principal curvatures) in orthogonal directions $\hat{\mathbf{e}}_1$ and $\hat{\mathbf{e}}_2$ (principal directions). These curvatures are computed using the first and second fundamental forms from differential geometry [dC76].

The thin-plate function is the sum of the principal curvatures squared, and integrated over the entire surface:

$$E(\mathbf{s}) = \int \left(\kappa_1^2 + \kappa_2^2\right) du\,dv \tag{2.21}$$

However, due to its highly non-quadratic character (that is, non-quadratic in **p**), this function is often replaced by the quadratic approximation:

$$E_{\text{approx}}(\mathbf{s}) = \int (\mathbf{s}_{uu} \cdot \mathbf{s}_{uu} + 2\mathbf{s}_{uv} \cdot \mathbf{s}_{uv} + \mathbf{s}_{vv} \cdot \mathbf{s}_{vv})\, du\,dv \tag{2.22}$$

(double subscripts on **s** denote second parametric derivatives). However, this function tends to poorly approximate the thin-plate functional when the parameterization area does not scale uniformly onto the surface. A first order membrane term (scaled by α) is added to penalize non-uniform parameterizations:

$$\begin{aligned} E_{\text{approx}}(\mathbf{s}) = \int \big[&(\mathbf{s}_{uu} \cdot \mathbf{s}_{uu} + 2\mathbf{s}_{uv} \cdot \mathbf{s}_{uv} + \mathbf{s}_{vv} \cdot \mathbf{s}_{vv}) + \\ &\alpha(\mathbf{s}_u \cdot \mathbf{s}_u + 2\mathbf{s}_u \cdot \mathbf{s}_v + \mathbf{s}_v \cdot \mathbf{s}_v)\big]\, du\,dv \end{aligned} \tag{2.23}$$

Typically, α is just large enough to prevent approximation error. This is the surface functional used in most variational frameworks including [GC95, HKD93, WW92, WW94].

Clearly, $E(\mathbf{s})$ is a function of **p** (the model degrees of freedom). $E_{\text{approx}}(\mathbf{s})$ is quadratic in **p** for linear surface schemes [GC95, HKD93, WW94], so E can be written as:

$$E_{\text{quadratic}}(\mathbf{s}) = \frac{1}{2}\mathbf{p}^\top \mathbf{K} \mathbf{p} \tag{2.24}$$

where **K** is a matrix derived from the shape representation. For linear surface representations, **K** can usually be found explicitly (given a particular mesh topology). A derivation of **K** for B-splines is provided in [WW92]. Due to the local refinement properties of most shape representations used in these applications, **K** will be sparse (mesh nodes have only a few neighbors), containing only $O(\dim \mathbf{p})$ non-zero entries, allowing for more efficient optimization techniques.

The objective function can also be measured with respect to the difference from a prototype shape $\mathbf{s}'$ [WW92] (with control points $\mathbf{p}'$), so that the minimization is performed with respect to $(\mathbf{s}-\mathbf{s}')$, resulting in:

$$E_{\text{quadratic}}(\mathbf{s}-\mathbf{s}') = \frac{1}{2}(\mathbf{p}-\mathbf{p}')^{\top}\mathbf{K}(\mathbf{p}-\mathbf{p}') \tag{2.25}$$

When minimized, this produces shapes that retain characteristics of the prototype, since $\mathbf{s}'$ is now the fairest possible surface. Effectively, this objective function measures bending away from the prototype surface.

2.4.3 Constraints

The constraints give the user control over the geometry of the surface. They provide a layer of abstraction between the underlying shape representation and parameterization so that the user can make statements like "the curve must pass through this point" and "the surface should contain this curve." A survey by Nowacki, Liu and Lu [NLL90] reviews types of constraints used with polynomial curve and surface schemes.

Of course, given a shape with a fixed number of degrees of freedom, it might not always be possible to satisfy all the constraints. This is solved by simply refining or subdividing the shape to add the degrees of freedom necessary [GC95, WW92, WW94]. Another possibility is that the constraints may be dependent, or even worse, might conflict. Automatic solutions to these issues are left as open problems (or are ignored) in the above mentioned work.

The user specifies a set of constraints, each taking the form:

$$A(\mathbf{p}) = 0 \tag{2.26}$$

where A is a function, reaching only zero when the constraint is satisfied. For example, to constrain a particular location on a surface (u_0, v_0) to pass through a point $\mathbf{x}$, the constraint is:

$$\mathbf{s}(u_0, v_0) - \mathbf{x} = 0 \tag{2.27}$$

When using linear surface representations, point constraints are linear in $\mathbf{p}$. If all of the constraints are linear, they can be accumulated into a matrix equation, with each row of the matrix corresponding to a single constraint:

$$\mathbf{A}\mathbf{p} = \mathbf{b} \tag{2.28}$$

where the matrix **A** and vector **b** depend on the particular constraints being imposed.

Note that the above point constraint does not mean that "some location" on the surface pass through **x**, but rather that a specific location, given by (u_0, v_0), passes through **x**. While it might seem desirable for the constrained points to slide in parameter space, it is prohibitively expensive to do so [WW94].

2.4.4 Fairing with constraints

The minimization of the objective function subject to the constraints is a constrained optimization problem. Most work [GC95, HKD93, WW94] uses a quadratic objective with linear constraints, which can be solved with a single linear system.

Solving a quadratic objective with linear constraints amounts to solving the following constrained minimization problem:

$$\min_{\mathbf{p}} \left\| \frac{1}{2}(\mathbf{p}-\mathbf{p}')^\top \mathbf{K}(\mathbf{p}-\mathbf{p}') \right\| \quad \text{subject to } \mathbf{A}\mathbf{p} = \mathbf{b} \tag{2.29}$$

There are a number of approaches to solving such a system including the use of Lagrange multipliers [GC95, WW92, WW94] and null-space projection [HKD93].

The Lagrange multiplier technique [Str88] adds additional degrees of freedom (one for each degree of constraint), to solve a larger, unconstrained system. The Lagrange multiplier **y** yields the unconstrained minimization:

$$\min_{\mathbf{p},\mathbf{y}} \left\| \frac{1}{2}(\mathbf{p}-\mathbf{p}')^\top \mathbf{K}(\mathbf{p}-\mathbf{p}') + (\mathbf{A}\mathbf{p}-\mathbf{b})^\top \mathbf{y} \right\| \tag{2.30}$$

At the minimum, the partial derivatives of the bracketed terms vanish (since this system is symmetric and positive definite). Differentiation leads to the linear system:

$$\begin{vmatrix} \mathbf{K} & \mathbf{A}^\top \\ \mathbf{A} & 0 \end{vmatrix} \begin{vmatrix} \mathbf{p} \\ \mathbf{y} \end{vmatrix} = \begin{vmatrix} \mathbf{K}\mathbf{p}' \\ \mathbf{b} \end{vmatrix} \tag{2.31}$$

Provided there are no problems with the constraint matrix (such as dependent rows), the above system can be solved with techniques such as LU decomposition [Str88]. Since this matrix is sparse (due to the local refinement property of the resulting surface, it has $O(\dim \mathbf{p})$ non-zero entries), sparse matrix techniques can be employed to solve the system in $O((\dim \mathbf{p})^2)$ time.

In [WW94], instead of solving this system in a single step, the conjugate gradient method [Str88] (an iterative linear equation solution technique) is used. Given the sparsity of the matrix, each iteration takes $O(\dim \mathbf{p})$, with convergence typically occurring in $O(\dim \mathbf{p})$ iterations (resulting in a quadratic time solution). Since this is an interactive system, with the user working directly with the surface, there is no need to show the user the final solution if they are still interacting—it is considered more important to show the user feedback [GW93]. As a result, the solver is only run a few steps before being redisplayed. During this time, the constraints may drift slightly, and the surface may become somewhat unfair. Once the user releases the surface, the solver can "catch up," and display the iterations toward the final surface over the next few seconds. This iterative technique requires a reasonable initial guess at the solution, to be efficient and ensure convergence. In this framework, since surfaces are built up from scratch, using the answer from the previous iteration is always sufficient.

The use of wavelets in [GC95] was intended for reducing the number of iterations required for the convergence of the conjugate gradient method. Other techniques for solving the main system are also available. Null-space projection (also called constraint reduction) transforms the constrained system into a smaller unconstrained system with the constraints built in [HKD93].

These variational techniques provide a valuable abstraction that allows the user to be ignorant of the underlying surface representation scheme. While some simple applications of these methods have appeared in recent modeling software, they are by no means in widespread use at this time. Chapter 3 discusses a new use of variational modeling techniques—for the generation of face models. This is a departure in using these techniques for interactive modeling or data fitting.

Chapter 3

Face model generation

A hallmark of the diversity and individuality of the people we encounter in daily life is the range of variation in the shape of their faces. A simulation or animation that fails to reproduce this diversity—whether by design or circumstance—deprives its characters of independent identities. To animate a bustling scene realistically or to play out an extended virtual interaction believably requires hundreds of different facial geometries, maybe even a distinct one for each person, as in real life.

It is a monumental challenge to achieve such breadth with existing modeling techniques. One possibility might be to use range scanning technology. This involves all the complexities of casting extras for a film: with scanning, each new face must be found on a living subject. And although scanning permits detailed geometries to be extracted quickly, scanned data frequently includes artifacts that must be touched up by hand. Another alternative is manual construction of face models, by deforming an existing model or having an artist design one from scratch; this tends to be slow and expensive.

This chapter describes an alternative, which was first presented in [DMS98]: a system capable of automatically generating distinct, plausible face geometries. This system constructs a face in two steps. The first step is the generation of a random set of *measurements* that characterize the face. The form and values of these measurements are computed according to *face anthropometry*, the science dedicated to the measurement of the human face. Anthropometric studies like [Far87, Far94] report statistics on reliable differences in shape across faces within and across populations. Random measurements generated according to the anthropometric profile of a population characterize the distinctive features of a likely face in that population.

In the second step, our system constructs the best surface that satisfies the geometric constraints that a set of measurements imposes, using *variational modeling* [GC95, TQ94, WW92], which was reviewed in Section 2.4. Variational modeling is a framework for building surfaces by constrained optimization; the output surface minimizes a measure of *fairness*, which in our case formalizes how much the surface bends and stretches away from the kind of shape that faces normally have. Having a fairness measure is necessary, since the anthropometric measurements leave the resulting surface underdetermined. Bookstein [Boo89] uses this same fairness measure as a method of data interpolation for sparse biometric data, supporting its utility for determining the geometry of an underdetermined biological shape. Variational modeling provides a powerful and elegant tool for capturing the commonalities in shape among faces along with the differences. Its use reduces the problem of generating face geometries into the problem of generating sets of anthropometric measurements.

The remainder of this chapter describes our techniques in more detail. We begin in Section 3.1 by introducing the problem of representing and specifying face geometry. In Section 2.3, we summarize the research from face anthropometry that we draw on; Section 3.2 describes how random measurements are generated from these results. In Section 3.3, we describe our use of variational techniques to derive natural face geometries that satisfy anthropometric measurements. We finish in Section 3.4 with illustrations of the output of our system.

3.1 Face modeling background

Human face animation is a complex task requiring modeling and rendering not only of face geometry, but also of distinctive facial features (such as skin, hair, and tongue) and their motions. Most research in face modeling in computer graphics has addressed these latter problems [LTK95, MTMdAT89, Par82, PW96].

Research on human geometry itself falls into two camps, both crucially dependent (in different ways) on human participation. The first approach is to extract geometry automatically from the measurement of a live subject. Lee, et al. [LTK95] use a range scan of a subject, and produce a physics-based model capable of animation. Akimoto, et al. [ASR93] use front and profile images of a subject to produce a model.

The second approach is to facilitate manual specification of new face geometry by a user. A certain facility is offered already by commercial modelers (though of course their use demands considerable artistic skill); several researchers have sought to provide higher levels of control. Parke [Par82] provides parameters which can control the face shape; and Magnenat-Thalmann, et al. [MTMdAT89] describe a more comprehensive set of localized deformation parameters. Patel [PW91] offers an alternative set of parameters similar in scope to [MTMdAT89] but more closely tied to the structure of the head. DiPaola [DiP91] uses a set of localized volumetric deformations, with a similar feel to [MTMdAT89] in their effects. Lewis [Lew89] discusses the use of stochastic noise functions as a means of deforming natural objects (including faces). In this case, the control maintained by the user is limited to noise generation parameters.

In contrast, we adopt a different approach: generating new face geometries automatically. More so than interactive methods, this approach depends on a precise mathematical description of possible face geometries. Many conventional representations of face shape seem inadequate for this purpose.

For example, the simple scaling parameters used by manual modeling techniques can perform useful effects like changing the width of the mouth or the height of the head; but they are unlikely to provide sufficient generality to describe a wide sampling of face geometries.

Meanwhile, for models based on principal components analysis (PCA)—an alternative representation derived from work in face recognition [VP97]—the opposite problem is likely. PCA describes a face shape as a weighted sum of an orthogonal basis of 3-D shapes (called principal components). This basis is constructed from a large bank of examples that have been placed in mutual correspondence. (This correspondence is very much like that required for image morphing [BN92]; establishing it is a considerable task, but not one that has evaded automation [VP97].)

PCA typically allows faces nearly identical to those in the bank to be accurately represented by weighting a truncated basis that only includes a few hundred of the most significant components. However, because components are individually complex and combined simply by addition, alternative weightings could easily encode implausible face shapes. Identifying which basis weights are reasonable is just the original problem (of characterizing possible faces) in a different guise. Bookstein [Boo91] describes this problem in terms of "latent variables," and notes that principal components often bear little resemblance to the underlying interdependent structure of biological forms. (In other words, it is quite difficult to extract non-linear dependencies between different shape aspects using a linear model like PCA.) At the same time, there is no guarantee that faces considerably outside the example set will be approximated well at all.

We therefore adopt a representation of face shape based on constrained optimization. The constraints—generated as described in Section 3.2—are based on the anthropometric studies of the face of [Far87, Far94, KS96] described in Section 2.3; we avoid the difficulty of learning possible geometries since these studies identify the range of variation in real faces. The constraint optimization, as described in Section 3.3, is accomplished by variational surface modeling.

3.2 Generating measurements

The rich descriptions of human geometry developed in anthropometry provide an invaluable resource for human modeling in computer graphics. This goes for artists as well as automatic systems: Parke and Waters [PW96] describe the importance of having a set of "conformation guidelines" for facial shape, which draw from artistic rules of face design. These guidelines provide qualitative information about the shape and proportion of faces, respecting the quantitative information found in anthropometric measurements.

In using such descriptions, automatic systems immediately confront the problem of bringing a model into correspondence with a desired set of measurements. A widely-used approach is to design a model whose degrees of freedom can be directly specified by anthropometric measurements. For example, in the early visualization frameworks for human factors engineering surveyed in [Doo82]—where anthropometric data first figured in graphics—articulated humans were made to exhibit specified body measurements by rigidly scaling each component of the articulation. Grosso, et al. [GQB89] describe a similar model, but scale physical characteristics (such as mass) as well, to produce a model suitable for dynamic simulation and animation. Azuola [Azu96] builds on Grosso's work, and generates random sets of (axis-aligned distance) measurements using covariance information (but not proportions). The purpose of this generation is to produce a fairly small sampling of differently sized people for human factors analysis.

Our work represents a departure in that we use anthropometric data to constrain the degrees of freedom of the model indirectly (as described in Section 3.3). This is a must for the diverse, abstract and interrelated measurements of face anthropometry. The flexibility of generating measurements as constraints offers additional benefits. In particular, it allows statistics about proportions to be taken into account as precisely as possible.

This section describes how our system uses published facial measurement and proportion statistics [Far87, Far94] to generate random sets of measurements. The generated measurements both respect a given population distribution, and—thanks to the use of proportions—produce a believable face.

3.2.1 The need for proportions

Start with a given population, whose anthropometric measurements are tabulated for mean and standard deviation (we later use the measurements from [Far94]). We can assume that the measurements are given by a Gaussian normal distribution (due to the lack of data concerning the shape of these the distributions), as corroborated by statistical tests on the raw the data [Far94]. This gives a naive algorithm for deriving a set of measurements—generate each measurement independently as if sampled from the normal distribution with its (estimated) mean and variance. Such random values are easily computed [PTVF92]; then, given the constraint-based framework we use, a shape can be generated to fit the resulting suite of measurements as long as the measurements are geometrically consistent.

Mere geometric consistency of measurements is no guarantee of the reasonable appearance of the resulting face shape, however. Anthropometric measurements are not independent. On the face, one striking illustration comes from the inclinations of the profile, which are highly intercorrelated. In the population described in [Far87], the inclinations to the front of the chin from under the nose (*sn-pg*) and from the lower lip (*li-pg*) take a wide range of values, but, despite the many curves in this part of the face, tend to agree very closely.

Published proportions provide the best available resource to model correlations between measurements such as these. (Covariance information more naturally applies here, but it is simply not available). For example, [Far87] tabulates the mean and variance for statistically significant ratios between anthropometric measurements for a population of young North American Caucasian men and women. Given a calculated value for one measurement, the proportion allows the other measurement to be determined using a random value from the estimated distribution of the proportion. Since the proportion reflects a correlation between these values, the resulting pair of measurements is more representative of the population than the two measurements would be if generated independently.

With many measurements come many useful proportions, but each value will be calculated only once. We must find the proportions that provide the most evidence about the distribution. The next section describes the algorithm we use to do that. It assumes that proportions can be applied in either direction (by approximating the distribution for the inverse proportion) and that we are generating a set of measurements all of which are related by proportions. (We can split the measurements into groups before applying this algorithm.) The algorithm also assumes that we are given a fixed initial measurement (or measurements) in this set from which other measurements could be generated. If we are generating a random face, the choice of which initial measurement to use is up in the air. We therefore find the best calculation scheme for each possible initial measurement, and then use the best of those. Random values for this initial measurement are generated by sampling its distribution. Thereafter, randomly generated proportions are used to generate the remaining dependent measurements.

The same algorithm could also be used to fill in measurements specified by a user (as a rough guide of the kind of face needed) or selected to be representative of an extreme in the population (for use in human-factors analysis). In this case, the algorithm gives a way of generating a plausible, random variation on this given information.

3.2.2 An algorithm for proportions

Given base measurements, our goal is to find the best way to use an inventory of proportions to calculate dependent measurements. We can describe this problem more precisely by viewing measurements as vertices and proportions as edges in a graph. Figure 3.1(a) shows a portion of this graph, given the measurements and proportions from [Far87, Far94] (some edge labels are omitted for the sake of readability). The presence of cycles in this graph exhibits the need to select proportions. A particular method for calculating measurements using proportions can be represented as a *branching* in this graph—an acyclic directed graph in which each vertex has at most one incident edge. The edge e from s to d in this branching indicates that d is calculated by proportion e from s. By assumption, we will require this branching to span the graph (this means adding dummy edges connecting multiple base measurements). An example branching is illustrated in Figure 3.1(b), and contains a single base measurement (the vertex marked with a double circle).

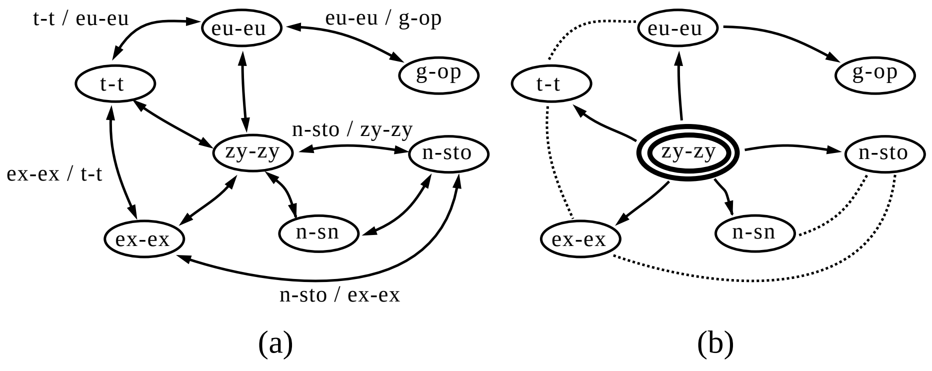


Figure 3.1: Interpreting measurements and proportions as a graph (a); Example branching used to compute measurements (b)

The algorithm associates each vertex v in the branching with a mean μ_v and variance σ_v^2. The variance is an indication of the precision of the statistical information applied in generating the measurement at v from given information. The smaller σ_v/μ_v, the more constrained the measurement. We take σ_v/μ_v as the weight of d.

For base measurements, σ_v is simply the standard deviation of the measurement. Thereafter, if an edge connects s to d with a proportion with mean μ_e and standard deviation σ_e, and s has mean μ_s and standard deviation σ_s, then the induced distribution at d is characterized by:

$$\mu_d = \mu_s \mu_e$$
$$\sigma_d^2 = \mu_s^2 \sigma_e^2 + \mu_e^2 \sigma_s^2 + \sigma_e^2 \sigma_s^2$$

(This assumes proportions and measurements are independent and Gaussian.) Note that the weight of d is always larger than the weight of s—this means the precision of the information concerning the distribution decreases as we go deeper into the branching.

Our goal in selecting proportions is to derive a branching T_M which assigns a minimum total weight to its vertices. This allows the most constrained features to determine the remaining features via proportionality relationships. We can modify Prim's algorithm for minimum spanning tree to solve this problem. Our algorithm maintains a subtree T of some optimal branching. Initially, the subtree contains just the root for the initial measurement. At subsequent stages, each vertex is associated with the least weight induced by any edge running from the branching to it. The algorithm incorporates the vertex v whose weight is the least into the tree, by the appropriate new edge e.

As with Prim's algorithm (c.f. [Gib85]), the argument that this algorithm works ensures inductively that if T is a subtree of some optimal branching T_M, then so is $T+e$. If e is not an edge in T_M, then T_M contains some other directed path to v, ending with a different edge e'. This path starts at the root of T, so it must at some point leave T. Because e was chosen with minimum weight and weights increase along paths, in fact the path must leave T at e'; since the algorithm chose e, e and e' induce the same weight for v. The inductive property is now established, since $(T_M - e') + e$ is an optimal branching of which T is a subtree.

3.3 Variational Modeling

Using the method outlined in Section 3.2, we generate complete sets of anthropometric measurements in Farkas's system. These constraints describe the geometry of the face in great detail, but they by no means specify a unique geometry for the face surface. For example, Farkas's measurements are relatively silent about the distribution of curvature over the face—the particular measurement that specifies the angle formed at the tip of the chin (the mentocervical angle; as in Figure 2.4), does not actually specify how sharply curved the chin is. What is needed then, intuitively, is a mechanism for generating a shape that shares the important properties of a typical face, as far as possible, but still respects a given set of anthropometric measurements. This intuition allows the problem of building an anthropometric face model to be cast as a constrained optimization problem—anthropometric measurements are treated as constraints, and the remainder of the face is determined by optimizing a surface objective function. This characterization allows us to apply variational modeling techniques, as described in Section 2.4.

This section describes how we adapt existing variational modeling techniques to develop the anthropometric face model. Our approach to variational modeling greatly resembles the framework in [WW92]; a key difference is that we perform most of the variational computation in advance and share results across different face generation runs. This amortization of computational cost makes it feasible to construct larger models subject to many constraints. However, it requires careful formulation of constraints and algorithms to exploit the constancy of the face model and its inventory of constraints.

As described in Section 3.3.1, we begin by specifying a space of possible face geometries using a parametric surface $\mathbf{s}(u,v)$, and locating the landmark points on the surface. We use a B-spline surface [Far93] to represent $\mathbf{s}$. This surface is specified by a control mesh, where the mesh degrees of freedom are collected into a vector $\mathbf{p}$. A particular instantiation $\mathbf{p}'$ of $\mathbf{p}$ provides a *prototype shape*, a reference geometry that epitomizes the kind of shape faces have. Both $\mathbf{s}(u,v)$ and $\mathbf{p}'$ are designed by hand, but the same parameterized surface and prototype shape are used to model any set of anthropometric measurements.

Given this shape representation, the task of the face modeling system is to allow a given set of anthropometric measurements $\mathbf{m}$ to be used as degrees of freedom for $\mathbf{s}$, *in place of* $\mathbf{p}$. It does so in two logical steps: (1), expressing $\mathbf{m}$ as constraints on $\mathbf{p}$ in terms of the landmark points as described in Section 3.3.2; and (2), using variational techniques as described in Section 2.4.2 through Section 3.3.5 to find a surface that satisfies the constraints and which minimizes bending and stretching away from the prototype face shape.

3.3.1 Surface representation

We choose a B-spline surface as a shape representation because of the demands both of anthropometric modeling and variational techniques. Our shape must be smooth, must permit evaluation of our constraints, and must have surface points and tangent vectors that are defined as linear combinations of its control mesh points. This scheme meets all of these requirements.

The specification of $\mathbf{s}(u,v)$ involved the manual construction of a B-spline control mesh for the face, shown in Figure 3.2. The mesh is a tube with openings at the mouth and neck; the geometry follows an available polygonal face model and (as required for accurate variational modeling) is parameterized to avoid excessive distortion of (u,v) patches.

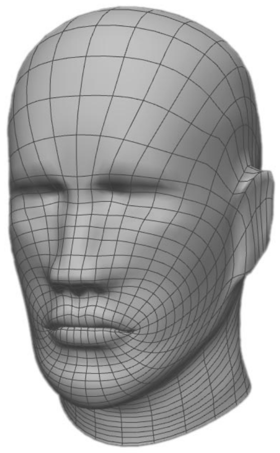


Figure 3.2: The prototype face model

Anthropometric landmarks are assigned fixed locations on the surface in (u,v) parameter space; some are also associated with constraints that enforce their fixed geometric interpretations. For example, in the case of the v landmark, which represents the top of the head, we ensure that the tangent to the surface at the point representing the landmark is in fact horizontal. We likewise add constraints to keep the model in FH position, so that the horizontal axis of the model is consistent with the axis by which landmarks are identified (and measurements taken). These constraints together constitute a set of *base* constraints which must be satisfied to apply any anthropometric measurement. Further constraints are then added to the model—one for each measurement.

3.3.2 Surface constraints

Our framework derives a shape by applying both linear and non-linear constraints. The linear constraints are derived from axial distance anthropometric measurements and the base constraints on the model; both can be represented as a linear function of the degrees of freedom of the model, $\mathbf{p}$. A matrix $\mathbf{A}$ describes how the values of all linear constraints are calculated, while a vector $\mathbf{b}$ encodes the intended values for those measurements. Thus solutions to these constraints satisfy:

$$\mathbf{A}\mathbf{p} = \mathbf{b} \tag{3.1}$$

Because $\mathbf{A}$ depends only on the *types* of constraint measurements, $\mathbf{A}$ can be solved in advance; then values of $\mathbf{p}$ can be computed directly from $\mathbf{b}$ given particular measurements $\mathbf{m}$.

Many of the constraints are non-linear, however. Each non-linear constraint is associated with a positive function measuring how far the surface is from the correct measurement. These functions are summed to give an overall penalty function P so that non-linear constraints impose the equation:

$$P\left(\mathbf{p}\right) = 0 \tag{3.2}$$

($P\left(\mathbf{p}\right) \geq 0$ for all $\mathbf{p}$). The remainder of this section describes the penalty functions associated with each type of measurement constraint.

The shortest distance measurement constrains the points $\mathbf{x}_i$ and $\mathbf{x}_j$ at a distance r apart using the penalty:

$$P_{\text{dist}}(\mathbf{x}_i, \mathbf{x}_j) = \left(\left\|\mathbf{x}_i - \mathbf{x}_j\right\| - r\right)^2 \tag{3.3}$$

The tangential distance constraint, which specifies the length of a surface curve to be r, is approximated using the chord-length approximation of a curve [Far93] using the points $\mathbf{x}_1 \ldots \mathbf{x}_n$:

$$P_{\text{arc}-\text{len}}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \left(\sum_{i=1}^{n-1} \|\mathbf{x}_i - \mathbf{x}_{i+1}\| - r\right)^2 \tag{3.4}$$

The points $\mathbf{x}_i$ all lie on a predetermined curve specified in (u, v)-space (using a B-spline), and are adaptively sampled as to achieve a good estimate of the arc length using the chord-length approximation.

The inclination measurement constraint fixes a vector $\mathbf{v}$ at an angle θ to a fixed axis $\mathbf{a}$:

$$P_{\text{incl}}(\mathbf{v}) = (\hat{\mathbf{v}} - Rot(\mathbf{a}, \theta))^2 \tag{3.5}$$

Using the rotation *Rot*, the axis $\mathbf{a}$ is aligned with the "goal" direction. $\mathbf{v}$ can be the direction between two points on the surface, as well as a surface tangent vector.

The angle measurement constraint positions the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ to be separated by the angle θ. It is treated as two independent inclination constraints:

$$\begin{aligned} P_{\text{angle}_1}(\mathbf{v}_1) &= (\hat{\mathbf{v}}_1 - Rot(\hat{\mathbf{v}}_2, \theta))^2 \\ P_{\text{angle}_2}(\mathbf{v}_2) &= (\hat{\mathbf{v}}_2 - Rot(\hat{\mathbf{v}}_1, -\theta))^2 \end{aligned} \tag{3.6}$$

3.3.3 Fairing

A fair surface can be constructed by minimizing an objective function $E(\mathbf{s})$. We will be using the linearized *thin-plate* functional (2.23) which measures the bending of the surface $\mathbf{s}$, with respect to the prototype shape (2.25). The use of a prototype shape instructs the fairing process to ignore expected regions of sharp curvature, such as the ears and nose on the face.

As described in Section 2.4.2, for linear surface representation schemes (including B-splines), the objective function in (2.23) can be evaluated exactly as a quadratic form $\frac{1}{2}\mathbf{p}^\top\mathbf{K}\mathbf{p}$, where $\mathbf{K}$ is determined based on the surface representation scheme; the construction for B-splines is given in [WW92]. Due to the local refinement property of B-splines, $\mathbf{K}$ is sparse.

3.3.4 Fairing with constraints

Given $\mathbf{K}$, the problem of fairing given purely linear constraints as in (3.1) is reduced to the a linearly constrained quadratic optimization problem (see Section 3.3.4), solved using the following linear system:

$$\begin{vmatrix} \mathbf{K} & \mathbf{A}^\top \\ \mathbf{A} & 0 \end{vmatrix} \begin{vmatrix} \mathbf{p} \\ \mathbf{y} \end{vmatrix} = \begin{vmatrix} \mathbf{K}\mathbf{p}' \\ \mathbf{b} \end{vmatrix} \tag{3.7}$$

Solving such a system requires selecting a technique that is mathematically sound and computationally feasible. For example, interactive modeling, with varying constraints and response time demands, requires the use of iterative solution methods, such as the conjugate gradient technique [GC95, WW94]. However, we can solve this system without iteration, using a sparse LU decomposition technique [GL89]; producing the decomposition takes $O(n^2)$ time given a $O(n)$ sparse $n \times n$ system. This technique is applicable because the set of constraints is hand-constructed, so we can guarantee that the constraint matrix $\mathbf{A}$ contains no dependent rows, and hence that the LU decomposition is well defined. It is feasible because the control mesh topology and the constraint matrix are unchanging, so that only one decomposition ever needs to be generated. Finding solutions is then quite efficient. In general, solving a system given an LU decomposition takes $O(n^2)$ time. However, we have found that the LU decomposition is roughly $O(n)$ sparse given our constraints. (This is not too surprising given that the each constraint involves only a few points on the surface; note that an LU decomposition can be sparse even if the actual inverse is dense.) This means that, in practice, solution steps require roughly linear time.

3.3.5 Non-linear constraints

As described in Section 3.3.2, the non-linear constraints are specified using the penalty function $P(\mathbf{p})$. Since this function is positive, it is simply added into the minimization (2.30) [PB88, WW92]. The extended linear system (3.7) has $\mathbf{Kp}' - \partial P(\mathbf{p})/\partial \mathbf{p}$ in place of $\mathbf{Kp}'$. Due to the non-linearity of P, this system must be solved iteratively. (By contrast, Section 3.3.4 described a non-iterative method for solving the linear constraints.)

At iteration i, we determine C_i to be used in place of $-\partial P(\mathbf{p})/\partial \mathbf{p}$ as:

$$C_i = C_{i-1} - \mu_i \frac{\partial P(\mathbf{p}_{i-1})}{\partial \mathbf{p}} \tag{3.8}$$

with $C_0 = \mathbf{0}$. The scalar value μ is a positive weight (analogous to a time-step in ODE integration), determined using an adaptive method such as step-doubling (for ODE solution) [PTVF92]. This results in the iterative linear system:

$$\begin{vmatrix} \mathbf{K} & \mathbf{A}^\top \\ \mathbf{A} & 0 \end{vmatrix} \begin{vmatrix} \mathbf{p}_i \\ \mathbf{y} \end{vmatrix} = \begin{vmatrix} \mathbf{K}\mathbf{p}' + C_i \\ \mathbf{b} \end{vmatrix} \tag{3.9}$$

where $\mathbf{p}_0$ is the solution corresponding to (2.31). Note that we still exploit the LU decomposition to allow steps to be solved quickly and exactly; this technique is stabler and faster to converge than the combination of a conjugate gradient technique with the penalty method. We experimented with linearizations of some of the non-linear constraints (and added them into **A**), but found little gain in efficiency, and decreased stability in solving.

In practice, the simultaneous use of all anthropometric constraints will lead to conflict. For example, some measurements lead to linearly dependent constraints; they are easily identified by inspection, and culled to keep **A** invertible. Similarly, when multiple measurements place non-linear constraints on similar features of nearby points on the model (without providing additional variation in shape), including all can introduce a source of geometric inconsistency and prevent the convergence of C. Our constraint set was selected by following a strategy of including only those constraints with the most locally confining definitions (i.e. constraints which affected fewer facial locations or more proximate facial locations were favored).

3.4 Results and discussion

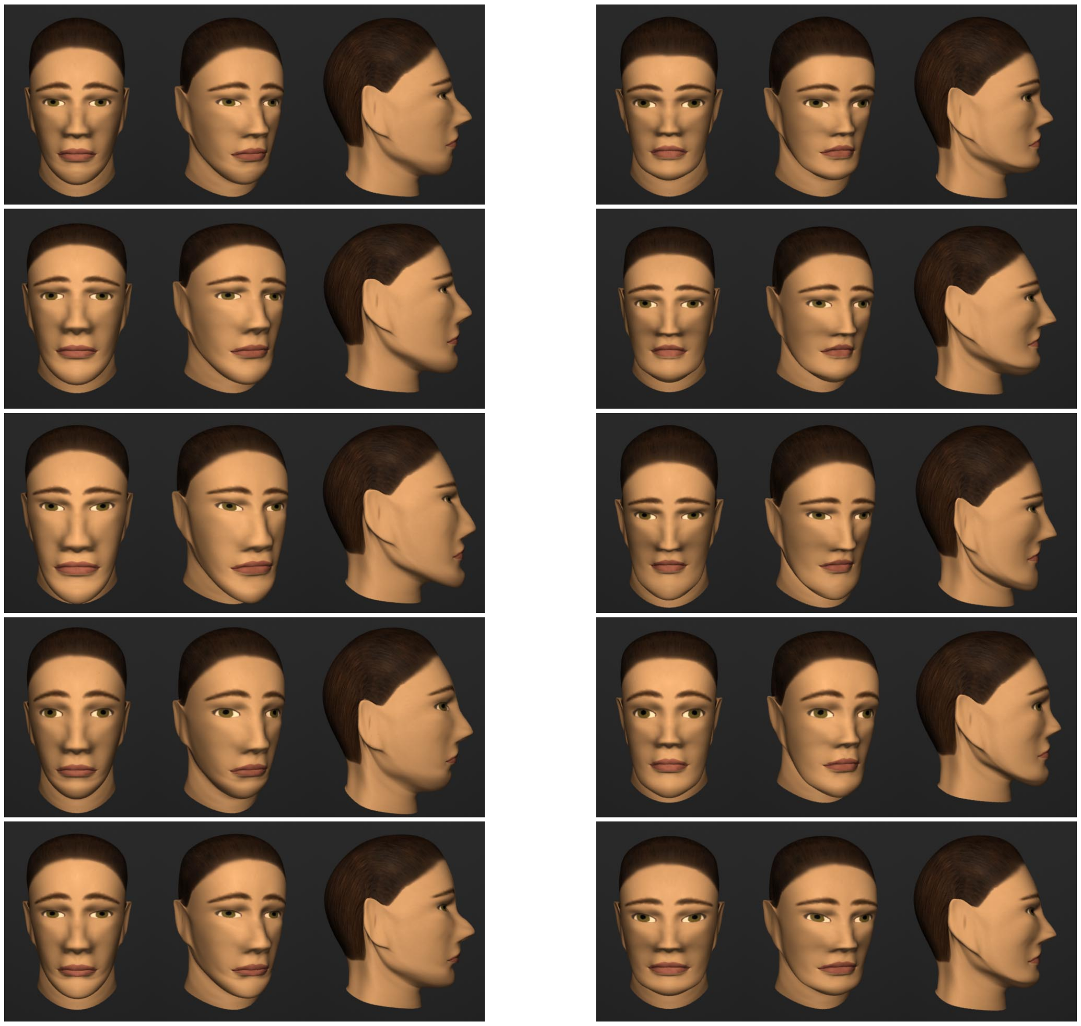
Sample face models derived using this technique are shown in Figure 3.3. To produce the measurements for these models, we ran the generation algorithm described in Section 3.2 on the measurements from [Far94] and the proportions from [Far87] for North American Caucasian young adult men and women. Faces for the random measurements were realized by applying the variational framework to a B-spline mesh (a grid 32 by 32) so as to satisfy the base constraints (a total of 15) and 65 measurements that give good coverage both of the shape of the face and of the kinds of measurements used in Farkas's system. There were a total of 120 proportions used as input to the algorithm in Section 3.2.2.

Producing the LU decomposition used for all these examples involved a one-time cost of roughly 3 minutes on an SGI 175 MHz R10000. Faces typically found their rough shape within 50 iterations; our illustrations were allowed to run for up to 200 iterations to ensure convergence to millimeter accuracy, resulting in runs that took about 1 minute for each face. Models were rendered using RenderMan.

Individual variation across the example males and females in Figure 3.3 encompass a range of features; for example, clear differences are found in the length and width of the nose and mouth, the inclinations of forehead and nose, as well as the overall shape of the face. At the same time, traits that distinguish men and women—such as the angle at the chin, the slope of the eyes and the height of the lower face (particularly at the jaw)—vary systematically and correctly (based on qualitative comparisons with the anthropometric data). Examining the variation within a population group, the thirty generated males in Figure 3.4 exhibit the expected range of geometric variation.

In order to quantify this comparison, the proportion-based measurement generation algorithm from Section 3.2.2 was validated by generating a large number of measurement sets, and comparing the resulting measurement distributions to the published figures from the corresponding population groups. On average, the means differed by about 1% (with a maximum deviation of 4.5%)—well below the differences in means between population groups. The standard deviations agreed comparably, where the generated measurements had standard deviations that range from being 5% lower to 20% higher than the published values. While this validation guarantees the plausibility of measurements on the generated face models, data is not available for comparing the entire geometry (this would require having, for example, a set of measurements of an individual along with a corresponding range scan). One would not expect such a comparison to agree anyway, as the prototype shape has a measurable effect on the resulting geometry. However, this effect decreases with the use of additional measurements, which suggests the need to search out additional data on face geometry (morphometrics [Boo91] seems to be a good starting point).

Despite the many changes, a single prototype shape was used for all examples. This



Males Females

Figure 3.3: Automatically generated face models (3 views of each)

gives the models commonalities in shape where anthropometric data is silent. Further, all the faces in Figures 3.3 and 3.4 use the same texture so as not to exaggerate their differences. The ears remain coarsely modeled (partly as a result of scarcity of measurements within the ear). Figure 3.5 shows the results when the skin and eye color is varied (manually), and hair is added, and additional detail (such as in the ears) is painted onto the texture. Note that the same texture is still used on all faces (with color conversions), and that the locations of hair is the same on all generated subjects–only the hair styles are manually specified.

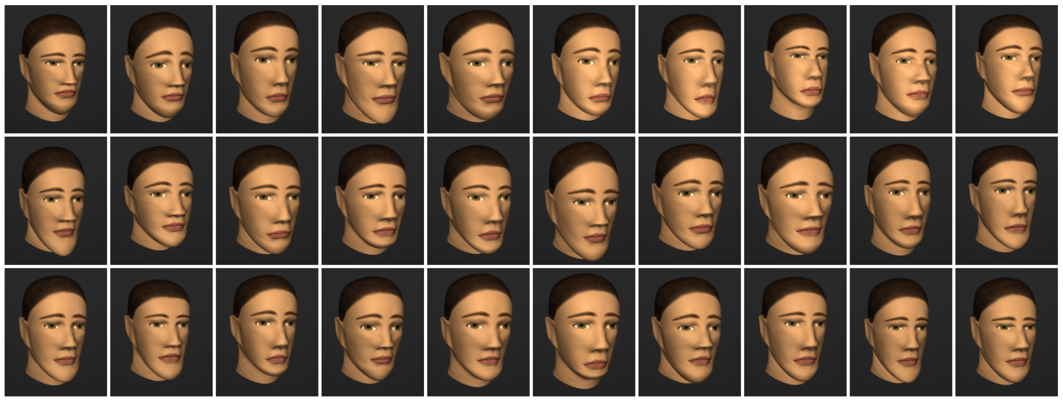
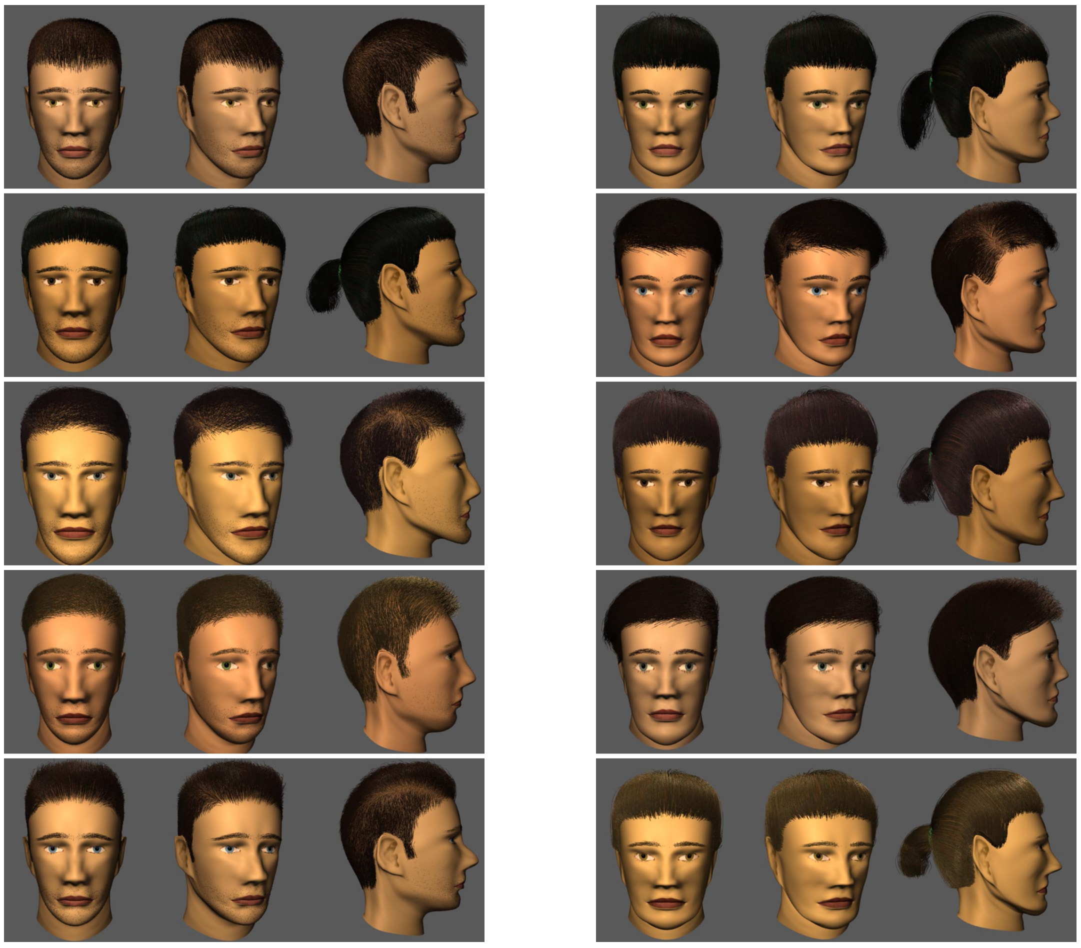


Figure 3.4: A male a minute



Males

Females

Figure 3.5: All gussied up

3.5 Summary

We have described a two step procedure for generating novel face geometries. The first step produces a plausible set of constraints on the geometry using anthropometric statistics; the second derives a surface that satisfies the constraints using variational modeling. This fruitful combination of techniques offers broader lessons for modeling: in particular, ways to scale up variational modeling—a technique previously restricted to modeling frameworks that have seen limited use to surface fitting tasks—for constrained classes of shapes, and ways to apply anthropometric proportions—long valued by artists and scientists alike—in graphics model generation.

Chapter 4

Face modeling for vision

The last chapter described a generative face model suitable for producing random facial geometries. However, due to its use of constrained optimization techniques, it cannot be used for shape estimation without raising efficiency concerns. This chapter describes a parameterized model of face shape and motion, which is to be used in a model-based vision framework. This face model encodes information about the shape, motion, and appearance of human faces.

Our three-dimensional face model is a polygonal model with a manually designed parameterization. The shape parameterization is specified as a sequence of deformations designed to capture the variabilities in shape and appearance of faces across the human population that are observed in face anthropometry studies [Far94]. Anthropometric statistics are also used to initialize the model with an average shape, and to bias shape estimation toward more "face-like" parameter combinations. The motion of the face (such as head motion or facial expressions) is specified using a small set of parameters. The Facial Action Coding System (FACS) [EF78] describes facial movements in terms of "action units", and motivates the design of the motions of our model. There are parameters for a variety of face motions such as opening of the mouth and raising of the eyebrows. The model is realized using a manually constructed series of parameterized deformations applied to a polygon mesh. The construction process is primarily an engineering task, and the majority of design decisions will be made based on its use as a model for estimation, tracking and geometric reasoning (such as for reasoning about self-occlusion).

4.1 Model construction

The deformable face model described here is shown in Figure 4.1(a). It is a polygon mesh, shown in (b), formed from ten component parts (such as the nose, mouth, or eyes), each shown in (c). These parts are connected together to form a single mesh, where the gaps between the parts are closed by a mesh “zippering” process. The mesh faces filling the gaps are primarily used for geometric inferencing (such as visibility determination), discussed in Section 4.5. With the construction of the model, we assume the observed subject is not wearing eyeglasses, and does not have large amounts of facial hair (such as a beard or mustache) that change the overall face shape.

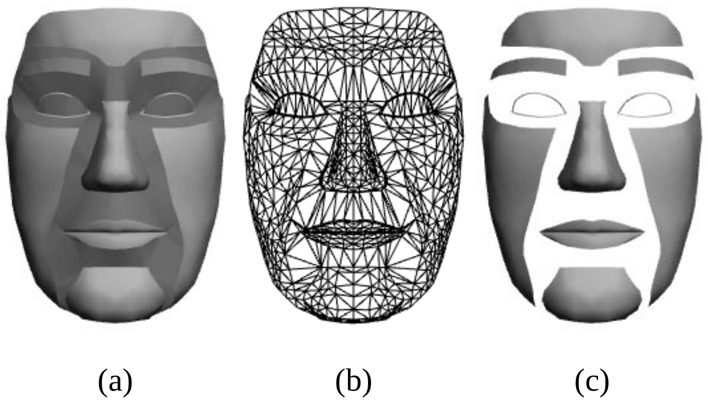


Figure 4.1: The deformable face model

4.2 Model parameterization

The face model constructed here has two kinds of parameters. Shape parameters describe the unchanging features of an observed face and capture variations in appearance across the human population. Motion parameters describe how an observed face changes during a tracking session. This separation produces an easier tracking problem by requiring a smaller description of object state to be estimated in each frame.

This division is often built into face models [BY95, LRF93, MRB95, TW93] to simplify model construction or estimation, while Reynard, et al. [RWBM96] use this separation to permit learning the variability of motions for a class of objects.

As a result of this separation, the parameters in $\mathbf{q}$ are rearranged and separated into $\mathbf{q}_{\mathrm{b}}$, which describe the underlying features of an individual, and into $\mathbf{q}_{\mathrm{m}}$, which describe motion (*both* rigid and non-rigid), so that $\mathbf{q} = (\mathbf{q}_{\mathrm{b}}^{\top}, \mathbf{q}_{\mathrm{m}}^{\top})^{\top}$.

The partition of $\mathbf{q}$ into $\mathbf{q}_{\mathrm{b}}$ and $\mathbf{q}_{\mathrm{m}}$ can also be viewed another way–the parameters in $\mathbf{q}_{\mathrm{b}}$ are a *static* quantity for a particular individual, and specify what a person looks like and how their facial expressions appear. The parameters in $\mathbf{q}_{\mathrm{m}}$ are a *dynamic* quantity, which change when a subject moves their head, opens their mouth, or makes a facial expression. The goal of a shape and motion estimation process is to recover the value of $\mathbf{q}$ from a sequence of frames. During estimation, the change in $\mathbf{q}_{\mathrm{b}}$ should tend to zero as the shape of the face is established. Once this occurs, fitting need only continue for $\mathbf{q}_{\mathrm{m}}$. So for reasons of efficiency, $\mathbf{q}_{\mathrm{b}}$ should include as many parameters as possible.

Also included in $\mathbf{q}_{\mathrm{b}}$ are parameters which specify the character of facial expressions, called *expression-shape* parameters—these parameters do not change the underlying face shape, but rather change the appearance of a particular facial expression. These parameters abstract information related to facial muscle placement. Figure 4.2 contains examples of varying expression-shape parameters that specify how a particular individual smiles. Figure 4.2(a) shows the model in its rest state (not smiling), while (b) and (c) contain differently shaped smiles. The smile in Figure 4.2(c) is more curved (like the Cheshire cat's) by varying some of the expression-shape parameters.

The model $\mathbf{x}$ is formed by applying deformation functions to the underlying polygon mesh $\mathbf{s}$ from Figure 4.1(b). The domain Ω of $\mathbf{s}$ is simply the set of points on its surface. There are separate deformation functions for shape ($\mathbf{T}_b$) and for motion ($\mathbf{T}_m$). For a face,

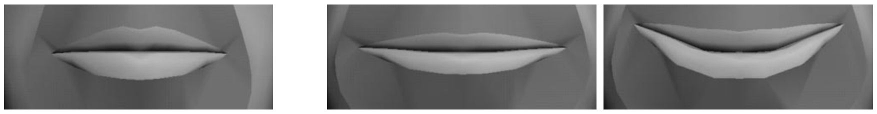

(a) (b) (c)

Figure 4.2: Example smile expression-shape deformations

it makes sense to apply the shape deformations first, so that:

$$\mathbf{x}(\mathbf{q};\mathbf{u}) = \mathbf{T}_m\left(\mathbf{q}_{\mathrm{m}};\ \mathbf{T}_b\left(\mathbf{q}_{\mathrm{b}};\ \mathbf{s}(\mathbf{u})\right)\right) \tag{4.1}$$

The shape deformation $\mathbf{T}_b$ uses the parameters $\mathbf{q}_{\mathrm{b}}$ to deform the underlying face mesh $\mathbf{s}$. On top of this is the motion deformation $\mathbf{T}_m$ with parameters $\mathbf{q}_{\mathrm{m}}$, which includes a rigid head translation and rotation, as well as non-rigid facial deformations. Of course, each of these deformations can be defined using a series of composed functions (see Appendix A), as will be seen in upcoming sections.

4.3 Model deformations

In order to represent the variabilities observed in anthropometric measurements, scaling and bending deformations, in addition to translation and rotation, are used in the construction of the face model. This section provides details on these deformations. The model designer carefully combines the deformations to produce a parameterized face model. The result of this construction is an underlying model (the polygon mesh) which has a series of deformations functions applied to it, each having a small number of parameters, and each is applied to a particular set of face parts, ranging from a single part to the entire face.

Rigid transformations such as translation and rotation are used for the placement of parts on the face. Scaling and bending deformations, shown in Figure 4.3, allow for the representation of a variety of face shapes. Each of these deformations is defined with respect to particular landmark locations in the face mesh. By fixing the deformations into the mesh, the desired effect of any particular deformation is not lost due to the presence of other deformations (since the landmark points are deformed along with the rest of the mesh). Although varying degrees of continuity can be attained for these deformations, each of the following deformations are C^1 continuous.

A shape (before any deformation is applied) which contains the landmark points p_0, p_1 and c is shown in Figure 4.3(a). Figure 4.3(b) shows the effect of scaling this shape along the displayed axis. The center point c is a fixed point of the deformation, while the region between p_0 and p_1 is scaled to have length d (the parameter of the scaling deformation). Portions of the shape outside this region are rigidly translated.

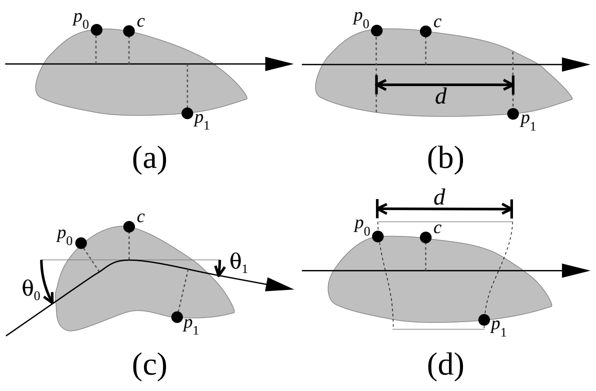


Figure 4.3: Scaling and bending deformations

Bending is applied in Figure 4.3(c), and shows the effect of bending the shape in (a) in a downward direction. The bending is applied to the area between p_0 and p_1, where c is the center of the bending. Outside this area, the shape is rotated rigidly. Each plane perpendicular to the bending axis is rotated by an angle determined by the distance of this plane to the center point c. The amount of bending is specified by the parameters θ_0 and θ_1, which specify the rotation angle at p_0 and p_1, respectively.

In addition, the spatial extent of each of these deformations can be localized, as shown in Figure 4.3(d). The influence of the scaling deformation varies in directions perpendicular to the axis, producing a tapering effect. Near the top of the shape, the object is fully scaled to be the length d, while the bottom of the object is unaffected by the deformation. The ability to restrict the effect of a deformation is vital in specifying the variations of shape seen in the face. We will now see how these deformations can be used to create the model.

4.3.1 Face shape

The underlying shape **s**, which is the polygon mesh shown in Figure 4.1, can take the shape of a variety of faces through the application of a number of spatial deformations. This parameterization of the model is specified by the model designer. The job of the designer is made easier by separating the face into parts, allowing each face model component to be treated separately. Instead of describing the entire model (which would be extremely lengthy and not particularly enlightening), a short description is provided which illustrates the concepts necessary for its construction.

Deformations are defined over a particular set of face model parts, although most deformations affect only one part. Example deformations that parameterize multiple parts, are those affecting the lower-face, which deform the chin and both cheeks. All of the deformations are specified in a particular order, and are applied in sequence to the underlying shape (see Appendix A). All of the parameters to describe the shape of the face at rest (there are approximately 80) are collected together into $\mathbf{q}_{\mathrm{b}}$. The shape deformations are collected together into a single deformation function $\mathbf{T}_b$.

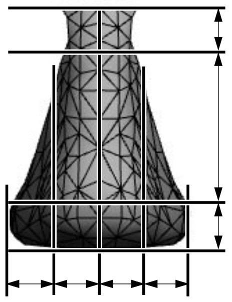


Figure 4.4: Scaling deformations of the nose

Figure 4.4 shows some of the scaling deformations defined for the nose. Each arrow indicates a particular scaling parameter (in the vertical or horizontal direction), that affects the space between the enclosing lines. The results of applying some of the deformations to the nose are shown in Figure 4.5. Figure 4.5(a) and (d) show two views of the default nose. Figure 4.5(b) shows a nose deformed using vertical scaling, while the pulled-up nose in (c) is produced using a localized bending deformation. Figure 4.5(e) and (f) show localized scaling affecting the width of the nose in different places.

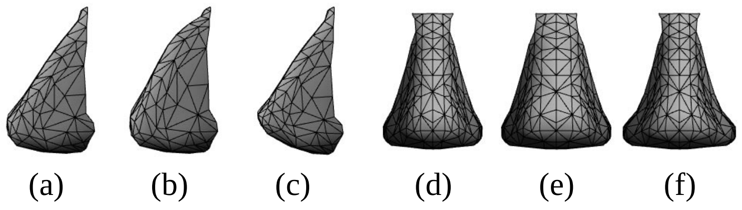


Figure 4.5: Example deformations affecting the nose

Verification of the face parameterization produced by the model designer can be accomplished by fitting the model to a series of randomly generated sets of facial measurements, as in Section 3.2. This is effectively a Monte Carlo method of sampling the space of face measurements. The fitting is easily accomplished, given a set of measurements, using the anthropometric forces described in Section 2.3. The model designer can alter the model parameterization when a particular set of face measurements cannot be satisfied by the model. We obtained a face model capable of representing a wide variety of faces after only a few design iterations.

4.3.2 Face motion

The deformations corresponding to motions (such as facial expressions) are modeled using the same techniques employed for face shape. However, there is no available motion data that corresponds to anthropometric data for shape (although such motion data might be available in the near future [GGW$^+$98]). The motion deformations are applied to the face in rest position—after the shape deformations, as in (2.2). Examples of modeled expressions are displayed in Figure 4.6. The model is capable of opening the mouth as in Figure 4.6(a), smiling (b), raising each eyebrow (c) and frowning each eyebrow (d). This results in a total of 6 expression parameters, each corresponding to a particular FACS action unit [EF78]. In addition to this are the six parameters for rigid head motion (translation and rotation), resulting in a total of 12 parameters in $\mathbf{q}_{\mathrm{m}}$.

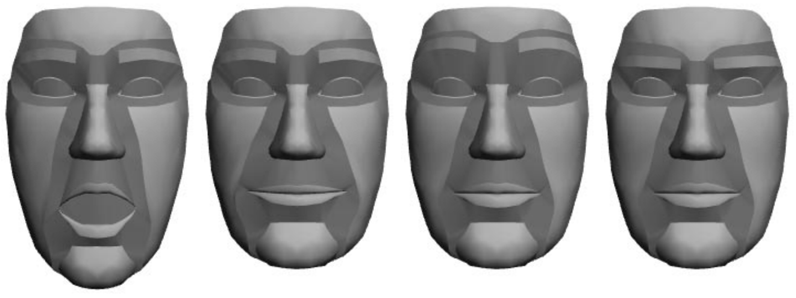


Figure 4.6: Face motion and expression deformations

The construction of expressions is simplified by decomposing each face motion into several component deformations. For example, the mouth opening deformation is decomposed into chin/cheek bending, lip scaling and lip translation. To facilitate tracking of these expressions by reducing the number of motion parameters, there is a single *control* parameter for each expression which uniquely determines all of its component parameters. Given a particular face motion which is constructed using a series of deformations with parameters b_i, the control parameter e determines the value b_i based on the formula:

$$b_i = s_i e \tag{4.2}$$

where s_i is the scaling parameter used to form the linear relationship between b_i and e. These scaling parameters are the expression-shape parameters included in $\mathbf{q}_\mathrm{b}$ (there are about 20 in total). For situations where these parameters are not estimated, these parameters are treated as constants, average values for which are determined by the designer during construction of the model.

The set of face motion parameters $\mathbf{q}_\mathrm{m}$ consists of the control parameters for each of the expressions (which are initially all zero), and the rigid translation and rotation specifying the head position. The parameters b_i are not estimated, but are determined directly by (4.2) using the estimated value of e. The motion deformations are collected together into the deformation $\mathbf{T}_m$.

4.4 Use of anthropometry data

The construction of our face model includes representation of the anthropometric measurements described in Section 2.3. Given the measurement descriptions in [Far94], they are realized using a straightforward set of geometric operations performed using points on the face model: given a value of $\mathbf{q}_{\mathrm{b}}$, a set of measurements can then be taken from the model.

Use of this data allows for a hand-crafted model to be biased towards more likely individuals, and places it in the class of constrained coverage models. For a particular set of model points $\mathbf{x}_1 \ldots \mathbf{x}_n$, a measurement M_j is written as:

$$M_j\big(\mathbf{x}_1, \ldots, \mathbf{x}_n\big) \quad j \in 1..M \tag{4.3}$$

where M is the number of measurements in Farkas' inventory. As an example, a *shortest distance* measurement is simply the following:

$$M_{\mathrm{dist}}\big(\mathbf{x}_1, \mathbf{x}_2\big) = \|\mathbf{x}_1 - \mathbf{x}_2\| \tag{4.4}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ are model points corresponding to the two landmarks used by the measurement. Note that these points depend on the shape parameters $\mathbf{q}_{\mathrm{b}}$, but not on the motion parameters $\mathbf{q}_{\mathrm{m}}$ (which is effectively zeroed when any anthropometric measurements are taken on the model—since this reflects the same "expressionless" conditions under which the data was originally gathered).

The statistical characterization of measurements and proportions, described in Section 2.3, can be built into the model in two ways. First, by using an average set of measurements, a set of parameters specifying the initial model is determined. This initial model is an anthropometrically "average" model, and is shown in Figure 4.1(a). Second, this characterization provides a means of biasing the face model shape parameters ($\mathbf{q}_{\mathrm{b}}$) toward more likely occurring individuals.

Given a particular set of population groups, average measurement values and variances are obtained from [Far94] as:

$$(\mu_j, \sigma_j^2) \quad j \in 1..M \tag{4.5}$$

The biasing of the parameters is performed using three-dimensional spring-like forces (a soft constraint) that are applied to the polygonal face model that softly enforce a measurement on the model. First, an energy is associated with each measurement:

$$E_j = \frac{1}{2}\left(M_j(\mathbf{x}_1, \ldots, \mathbf{x}_n) - \mu_j\right)^2 \tag{4.6}$$

Then, the force resulting from the energy E_j, which is applied to model domain point $\mathbf{u}_i$ (which corresponds to the point $\mathbf{x}_i$ on the model surface), is obtained as:

$$\mathbf{f}_{E_j}(\mathbf{u}_i) = -\frac{\partial E_j}{\partial \mathbf{x}_i} = -\left(M_j(\mathbf{x}_1, \ldots, \mathbf{x}_n) - \mu_j\right)\frac{\partial M_j}{\partial \mathbf{x}_i} \tag{4.7}$$

The penalty formulations of constraints in Section 3.3.2, which were used to formulate constraints for the generative model in Chapter 3, correspond to the energies in (4.6).

The total anthropometric force applied to model domain point $\mathbf{u}_i$ is computed as the *weighted* sum of all measurement forces at $\mathbf{u}_i$:

$$\mathbf{f}_{\mathrm{ant}}(\mathbf{u}_i) = \sum_{j \in 1..M} \left(1 - \frac{e^{-E_j/\sigma_j^2}}{\sqrt{2\pi\sigma}}\right)^{\rho} \mathbf{f}_{E_j}(\mathbf{u}_i) \tag{4.8}$$

Each force is weighted by a quantity which is a power (ρ) of how improbable the current measurement is (assuming a Gaussian distribution on the anthropometric measurements [Far94]). This weighting prevents the model from actually attaining the average set of measurements, but instead is simply biased towards them. For values of ρ around 10, forces on measurements within one standard deviation of the mean for that measurement are effectively ignored.

The weighting on these forces makes it clear that these forces are simply being used as a prior on the shape. It is possible to consider a deformable framework from a purely statistical point of view [BFO95], although re-interpreting force distributions such as these (as well as other more complex techniques borrowed from physics, such as the solution of constraints) is an open research problem.

For proportions, the energy would involve two measurements as:

$$E_{jk} = \frac{1}{2}\left(M_j(\mathbf{x}_1,\ldots,\mathbf{x}_n) - p_{jk}\cdot M_k(\mathbf{x}'_1,\ldots,\mathbf{x}'_{n'})\right)^2 \tag{4.9}$$

where p_{jk} is the mean proportion between measurements μ_j and μ_k. As with the above for measurements, a force distribution for proportion data is obtained using this energy.

4.5 Face feature and edge determination

The tracking of edges in a deformable model framework, as described in Section 2.2.1, is facilitated by knowing what locations of the face model are likely to produce edges in an image. On the face, certain features are likely to produce edges in the image. The particular features chosen are the boundary of the lips and eyes, and the top boundary of the eyebrows. Edges in the polygon mesh which correspond to these features were manually marked during the model construction, and are shown in Figure 4.7(a).

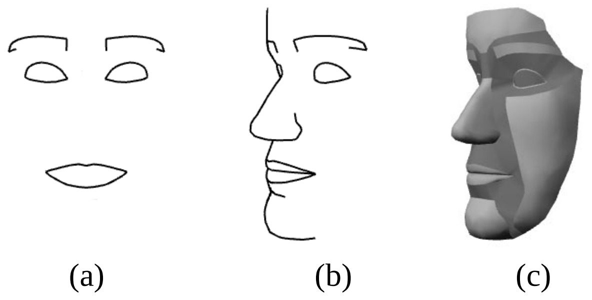


Figure 4.7: Likely face features in an image

Other likely candidates for producing edges are the regions on the model of high curvature. The base of the nose and indentation on the chin are examples of high curvature edges, and can be seen in Figure 4.7(b). Occluding boundaries on the model also produce edges in the image, and can be determined using the three-dimensional model. The location of occlusion boundaries on the model will be useful when determining the quality of selected points for the measurement of optical flow.

Of course, for an edge to be produced in the image, the corresponding region on the face must be visible to the camera. This visibility determination is performed using the model and camera transformation. The model depth information can be used to determine the parts of the model that are visible to the camera (the frontmost regions of the model). Figure 4.7(b) shows visible locations of the model (features, high curvature and occluding edges) that are likely to produce edges, given the model in (c).

Once the locations on the model are known which are likely to produce image edges, two-dimensional edge-based forces [MT93] are applied to the model, as given by (2.12). These forces contribute to the value of $\mathbf{f}_\mathbf{q}$ (affecting parameters in both $\mathbf{q}_\mathrm{b}$ and $\mathbf{q}_\mathrm{m}$) based on (2.9). Over the course of fitting, these edge forces "pull" the model so that the model edges become aligned with their corresponding image edges, as described in Section 2.2.1. The next two chapters describe how this face model is used in a deformable model framework for shape and motion estimation.

4.6 Summary

This chapter contained a description of the model-building process for a deformable face model; the parameterization of the model was built by hand. At the same time, anthropometric data was used to bias the model towards more likely individuals. This 3-D model can then be used to determine probable locations of image edges, as well as information about the model's self-occlusion. All of this results in a value for $\mathbf{f}_\mathbf{q}$: the data forces applied to the model.

Chapter 5

Shape and Motion Estimation

This chapter describes a constraint approach to optical flow within a deformable model framework for shape and motion estimation. We show that the approach can greatly improve the ability to estimate motion, especially by exploiting the distinction between shape and motion built into the parameterization of the model. The work here builds on the deformable model framework described in Section 2.1, on the model-based optical flow work from Section 2.2.2, and on the model-based edge fitting work from Section 2.2.1.

Our approach can be summarized as follows. To start, image velocities in the optical flow constraint equation are interpreted as projections of the model's three-dimensional velocities; this produces a system of optical flow equations that constrain the velocities of the motion parameters of the model. In the theory of dynamic systems [Sha89], velocity constraints such as these are called non-holonomic.

The velocities of the motion parameters are already accounted for as resulting from the application of edge-based forces; finding the equilibrium resulting from these forces amounts to a straightforward optimization problem (which can be solved using a deformable model framework). With the addition of the optical flow constraints, a constrained optimization problem results.

The constrained dynamic system is solved using the method of Lagrange multipliers. This involves converting the optical flow constraints into constraint forces that are combined with other forces (such as edge-based forces) to improve the estimation of the model. The method of Lagrange multipliers recasts the optical flow constraints as two kinds of forces. One provides the standard least-squares model-based solution to the optical flow constraints [NH87, LRF93]. The second is a constraint enforcement term which ensures the optical flow constraint *remains* satisfied when combined with edge forces. In order to provide a means for the combination of different sources of noisy information, an extended Kalman filter [Gel74] is used. The optical flow constraint can be introduced into such a filtering framework, resulting in a standard application of Kalman filtering to a system augmented with constraint information.

This treatment of optical flow offers several advantages. Since our technique is model-based, we avoid the explicit computation of the optical flow field by instead using the optical flow constraint equation at select pixels in the image. Furthermore, using our three-dimensional model, we can avoid choosing pixels on occlusion boundaries (which violate the optical flow constraint equation) by determining their probable locations in the image. (Similarly, we can determine likely locations of edges in the image to produce edge forces on the model.) Finally, the presence of the constraint enforcement term yields a profitable combination of the optical flow solution with the edge forces. Problems with tracking error accumulation are alleviated using these edge forces, which now keep the model aligned with its image without a statistically relevant violation of the optical flow constraint.

5.1 Related Work

A wide variety of techniques have been used in the extraction and recognition of facial expressions in image sequences. Several 2-D face models based on splines or deformable templates [LTC97, MRB95, YCH92] have been developed which track the contours of a face in an image sequence. Terzopoulos and Waters [TW93] and Essa and Pentland [EP97] use a physics-based 3-D mesh with many degrees of freedom, where face motion is measured in terms of muscle activations. Edge forces from snakes are used in [TW93], while in [EP97], the face model is used to regularize an optical flow field that is used in expression recognition. A structure from motion approach is used by Jebara and Pentland [JP97] to track head motion using a small number of image features. The rough 3-D shape of the head is also extracted.

Another approach is to directly use the optical flow field from face images. Yacoob and Davis use statistical properties of the flow for expression recognition [YD94]. Black and Yacoob parameterize the flow field based on the structure of the face under projection [BY95]. Basu, et al [BEP96] extract a flow field, and then regularize it using a 3-D ellipsoid model of the head. Addressing the problem of image coding, Li, et al [LRF93] estimate face motion using a simple 3-D model by a combination of prediction and a model-based least-squares solution to the optical flow constraint equation (without a constraint enforcement term). A render-feedback loop is used to combat error accumulation in tracking.

None of these approaches permit large head rotations due to the use of a 2-D model (or an imprecise 3-D model), and the inability to handle self-occlusion. None of the previous work makes a serious attempt in *extracting* a detailed 3-D *shape* description of the face from an image sequence. At best, the rough shape is determined [JP97], or the boundary of face parts are located to align the model with an image. And most importantly, optical flow has been solved separately from other cues, producing combined solutions which may not respect the optical flow constraint. Our system, first presented in [DM96], uses a 3-D model, and allows the tracking of large rotations by using self-occlusion information from the model. We also extract the shape of the face using a combination of edge forces and anthropometry information. Our optical flow solution was in part motivated by [BY95], but is also superficially similar to [LRF93]. Our formulation, unlike [BY95, LRF93], includes a constraint enforcement term, and allows us to improve our solution by including additional information. Our face model also permits the use of a small number of image points to sample the optical flow field, as well as the computation of edge forces to prevent error accumulation in the motion.

5.2 Optical flow constraints

In the following, the use of optical flow constraints on deformable models is presented. The optical flow constraint equation, which expresses a constraint on the optical flow velocities, is reformulated as a system of dynamic constraints that constrain $\dot{\mathbf{q}}$, the velocity of the deformable model. The resulting information will be combined with the model forces $\mathbf{f_q}$ so that the constraint remains satisfied. The optical flow constraint equation is used at a number of select locations in the image to constrain the motion of the model, instead of explicitly computing the optical flow field. The use of optical flow information greatly improves the estimation of $\mathbf{q}_\mathrm{m}$, the motion parameters of the deformable model.

Hard constraints on a dynamic system (the type of constraints used here) restrict the shape and motion by reducing the number of available degrees of freedom, while soft constraints (such as spring forces) bias the behavior of the system toward a certain goal (involving the system energy). Hard constraints are specified by equations involving $\mathbf{q}$ (or its time derivatives). The technique used here for satisfying a set of hard constraints is the addition of a constraint force to the system, which is determined at each iteration of the system.

Constraints which depend only on $\mathbf{q}$ are called *holonomic* constraints, and constrain the model to a set of allowable positions. They can be used in a deformable model formulation, for instance, to add point-to-point attachment constraints between the parts of an articulated object [MT93]. A holonomic constraint $\mathbf{C}$ has the general form

$$\mathbf{C}(\mathbf{q}, \mathrm{t}) = \mathbf{0} \tag{5.1}$$

Non-holonomic constraints additionally depend on the velocity of the parameters, $\dot{\mathbf{q}}$, and constrain the motion of the model. A non-holonomic constraint **C** has the general form

$$\mathbf{C}(\dot{\mathbf{q}},\mathbf{q},\mathrm{t}) = \mathbf{0} \tag{5.2}$$

In the following, we show how the optical flow constraints take this form and are incorporated into a dynamic system using the method of Lagrange multipliers.

5.2.1 Constraint formulation

The discussion in Section 2.2.2 describes how the optical flow constraint equation can be reformulated in terms of the velocities of the model degrees of freedom. This rewriting uses an identification of the image velocity (u_i, v_i) at pixel i with its corresponding model velocity $\dot{\mathbf{x}}_p(\mathbf{u}_i)$:

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \dot{\mathbf{x}}_p(\mathbf{u}_i) = \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i)\dot{\mathbf{q}}_{\mathrm{m}} \tag{5.3}$$

where $\mathbf{L}_p = \left[\mathbf{L}_{\mathrm{b}p}\,\mathbf{L}_{\mathrm{m}p}\right]$ is the projected model Jacobian that has been split into blocks corresponding to $\mathbf{q}_{\mathrm{b}}$ and $\mathbf{q}_{\mathrm{m}}$. Direct use of the optical flow information only provides motion information, and as a result, only $\mathbf{q}_{\mathrm{m}}$ is affected. To clarify this: any observed motion is caused by dynamic changes in the true value of $\mathbf{q}_{\mathrm{m}}$. The true value of $\mathbf{q}_{\mathrm{b}}$ is a static quantity—the meaning of $\dot{\mathbf{q}}_{\mathrm{b}}$ comes from the analogy of physics, where the value of $\mathbf{q}_{\mathrm{b}}$ improves over the course of fitting (over time).

The non-holonomic constraint equation for the optical flow at a pixel i in the image can be found by rewriting the optical flow constraint equation (2.13) using (5.3):

$$\nabla \mathrm{I}_i \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i)\dot{\mathbf{q}}_{\mathrm{m}} + \mathrm{I}_{t_i} = 0 \tag{5.4}$$

Instead of using this constraint at every pixel in the image, n pixels are selected from the input image (where $n \gg \dim \mathbf{q}_{\mathrm{m}}$). Section 5.4 describes the criterion used to choose these particular points, and also describes how some of the known difficulties in the computation of the optical flow are avoided in this model-based approach.

For the n chosen pixels in the image, the system of equations based on (5.4) becomes:

$$\begin{bmatrix} \nabla \mathrm{I}_1 \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_1) \\ \vdots \\ \nabla \mathrm{I}_n \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_n) \end{bmatrix} \dot{\mathbf{q}}_{\mathrm{m}} + \begin{bmatrix} \mathrm{I}_{t_1} \\ \vdots \\ \mathrm{I}_{t_n} \end{bmatrix} = \mathbf{0} \tag{5.5}$$

which can be written compactly as

$$\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}} = \mathbf{0} \tag{5.6}$$

This equation is simply a model-based version of the optical flow constraint equation, which was discussed in Section 2.2.2. Instead of solving it on its own, however, it is used as a *constraint* on the motion of the model.

5.2.2 Solving the dynamic system

The constrained system of equations (2.8) and (5.6) are solved using the method of Lagrange multipliers [Sha89, Str88]. The Lagrange multiplier technique adds additional degrees of freedom (one for each degree of constraint), to form a larger, unconstrained system. The initial dynamic equation of motion (2.8), now split into two parts corresponding to $\mathbf{q}_{\mathrm{b}}$ and $\mathbf{q}_{\mathrm{m}}$, is modified by adding the constraint force $\mathbf{f}_{\mathrm{c}}$ to $\dot{\mathbf{q}}_{\mathrm{m}}$:

$$\dot{\mathbf{q}}_{\mathrm{b}} = \mathbf{f}_{\mathbf{q}_{\mathrm{b}}}, \qquad \dot{\mathbf{q}}_{\mathrm{m}} = \mathbf{f}_{\mathbf{q}_{\mathrm{m}}} + \mathbf{f}_{\mathrm{c}} \tag{5.7}$$

Adding $\mathbf{f}_{\mathrm{c}}$ ensures the constraint equation is satisfied, and also cancels the components of $\mathbf{f}_{\mathbf{q}_{\mathrm{m}}}$ that would violate the constraint. Using the Lagrange multiplier $\boldsymbol{\lambda}$, the constraint force can be solved for as:

$$\mathbf{f}_{\mathrm{c}} = -\mathbf{B}^{\top}\boldsymbol{\lambda} \tag{5.8}$$

We can combine equations (5.6), (5.7) and (5.8) to form:

$$\mathbf{B}\mathbf{B}^{\top}\boldsymbol{\lambda} = \mathbf{B}\mathbf{f}_{\mathbf{q}_{\mathrm{m}}} + \mathbf{I}_{\mathrm{t}} \tag{5.9}$$

and can now determine the constraint force (by multiplying (5.9) on the left by $\mathbf{B}^+$, the pseudo-inverse [Str88] of $\mathbf{B}$):

$$\mathbf{f}_\mathrm{c} = -\mathbf{B}^+(\mathbf{B}\mathbf{f}_{\mathbf{q}_\mathrm{m}} + \mathbf{I}_\mathrm{t}) = -\mathbf{B}^+\mathbf{I}_\mathrm{t} - \mathbf{B}^+\mathbf{B}\mathbf{f}_{\mathbf{q}_\mathrm{m}} \tag{5.10}$$

which results in the unconstrained dynamic system:

$$\dot{\mathbf{q}}_\mathrm{b} = \mathbf{f}_{\mathbf{q}_\mathrm{b}}, \qquad \dot{\mathbf{q}}_\mathrm{m} = -\mathbf{B}^+\mathbf{I}_\mathrm{t} + \left[\mathbf{1} - \mathbf{B}^+\mathbf{B}\right]\mathbf{f}_{\mathbf{q}_\mathrm{m}} \tag{5.11}$$

The first term of $\dot{\mathbf{q}}_\mathrm{m}$ in (5.11), $-\mathbf{B}^+\mathbf{I}_\mathrm{t}$, is a model-based linear least-squares solution to the optical flow constraint equations (5.6) [LRF93]. A model-based solution to the optical flow constraint equations attributes the flow in the image to motion parameters in the model. This works as follows. A change to any motion parameter induces a characteristic motion field in the image. Figure 5.1 illustrates these vector fields for particular motion parameters of our face model (described in Section 4.3.2). Figure 5.1(a) shows the vector field arising from translation toward the camera; the focus of expansion can be seen in the center of the nose. Figure 5.1(b) shows the field for horizontal translation, Figure 5.1(c) shows the field for rotation about the vertical axis. Finally, Figure 5.1(d) shows the field produced by opening the mouth. Formally, these 2-D vector fields are obtained by considering each column of $\mathbf{L}_{\mathrm{m}p}(\mathbf{u})$ over the entire model (for all $\mathbf{u} \in \Omega$). The linear combination of the fields $\mathbf{L}_{\mathrm{m}p}(\mathbf{u})$ using the weights $-\mathbf{B}^+\mathbf{I}_\mathrm{t}$ best satisfies (5.6) at the sampled pixels in the least-squares sense.

The second term in (5.11) contains the edge forces $\mathbf{f}_{\mathbf{q}_\mathrm{m}}$ scaled by the matrix $(\mathbf{1} - \mathbf{B}^+\mathbf{B})$. (The computation of edge forces was described in Section 2.2.1.) This projection matrix cancels the component of $\mathbf{f}_{\mathbf{q}_\mathrm{m}}$ that violates the constraint (5.6) on $\dot{\mathbf{q}}_\mathrm{m}$. By scaling the edge forces, this term prevents small errors in $\mathbf{q}_\mathrm{m}$ from accumulating.

Solving the system in (5.11) simply involves integrating it over time (we use an Euler step):

$$\mathbf{q}(t+1) = \mathbf{q}(t) + \dot{\mathbf{q}}(t)\Delta t \tag{5.12}$$

The process used to initialize the system (to determine the value of $\mathbf{q}(0)$) is described in Chapter 7.

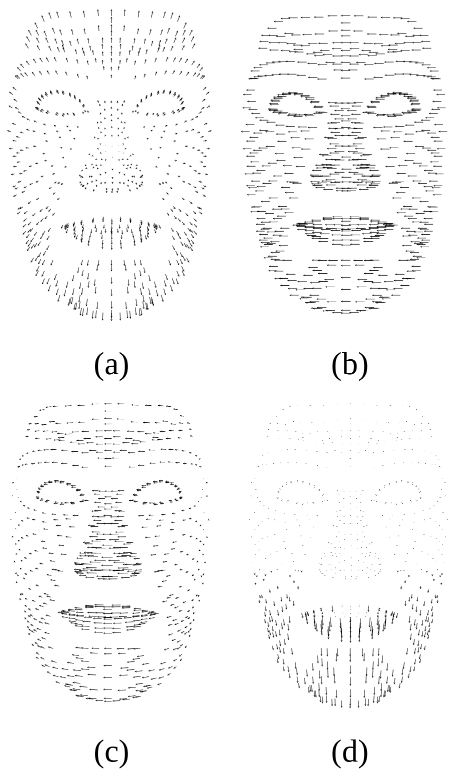


Figure 5.1: Sample 2-D vector fields $\mathbf{L}_{\mathrm{m}p}(\mathbf{u})$

5.3 Kalman Filtering

The optical flow constraint on $\dot{\mathbf{q}}_{\mathrm{m}}$ is imperfect due to noise and estimation errors. It is therefore desirable to have only a partial cancellation of $\mathbf{f}_{\mathbf{q}_{\mathrm{m}}}$; this is accomplished through the use of filtering. This section describes how the computation from the previous section is cast as an extended Kalman filter.

Kalman filtering [Gel74] has become a popular tool in computer vision, and the formulation here is, on the whole, similar to other applications [BC86, Met96]: there is a measurement equation which models the noise inherent in the data gathering process, and there is a process model, which predicts the behavior of the system based on the current state. The initialization and tuning of the filter is accomplished using standard techniques. The significant difference here, is that there is not only the edge data equation (2.8), which has been previously used as a filtering measurement equation [Met96], but there is also a data-based constraint equation (5.6). The first part of this section describes one reasonable way of using this constraint in the measurement equation. Alternative formulations are possible; ours corresponds to the non-stochastic solution in (5.11). The remainder of the section describes an extended Kalman filter based in part on this measurement equation.

By assuming a Gaussian noise model for both the measurements and state, the Kalman filter can maintain an estimate of the state $\mathbf{y}$ and the state covariance $\mathbf{P}$. While the assumption of Gaussian noise might not be particularly accurate in describing the actual noise in the system, it permits a much simpler solution while still capturing a large amount of the uncertainty.

The *measurement* equation for the Kalman filter relates the measurements $\mathbf{z}$ to the state $\mathbf{y}$ using the measurement matrix $\mathbf{H}$. Terms $\mathbf{v}_{\mathbf{f}_\mathbf{q}}$ and $\mathbf{v}_{\mathbf{I}_\mathrm{t}}$ are added to represent the assumed zero-mean Gaussian noise in $\mathbf{f}_\mathbf{q}$ and $\mathbf{I}_\mathrm{t}$; they have covariances $\mathbf{R}_{\mathbf{f}_\mathbf{q}}$ and $\mathbf{R}_{\mathbf{I}_\mathrm{t}}$ respectively:

$$\mathbf{z}(t) = \mathbf{H}(t)\mathbf{y}(t) + \begin{pmatrix} \mathbf{v}_{\mathbf{f}_\mathbf{q}}(t) \\ \mathbf{v}_{\mathbf{I}_\mathrm{t}}(t) \end{pmatrix} \tag{5.13}$$

where the construction of $\mathbf{H}$, $\mathbf{y}$ and $\mathbf{z}$ in (5.14) comes from (5.6), (5.7) and (5.8).

$$\begin{gathered} \mathbf{H} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{B}^\top \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \end{bmatrix}, \quad \mathbf{y} = \begin{pmatrix} \dot{\mathbf{q}}_\mathrm{b} \\ \dot{\mathbf{q}}_\mathrm{m} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \dot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{pmatrix}, \\ \mathbf{z} = \begin{pmatrix} \mathbf{f}_{\mathbf{q}_\mathrm{b}} \\ \mathbf{f}_{\mathbf{q}_\mathrm{m}} \\ -\mathbf{I}_\mathrm{t} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_\mathbf{q} \\ -\mathbf{I}_\mathrm{t} \end{pmatrix} = \begin{pmatrix} \sum_j \mathbf{L}_p(\mathbf{u}_j)^\top \mathbf{f}(\mathbf{u}_j) \\ -\mathbf{I}_\mathrm{t} \end{pmatrix} \end{gathered} \tag{5.14}$$

The state $\mathbf{y}$ consists of the parameter velocities $\dot{\mathbf{q}}$; together with the Lagrange multipliers $\boldsymbol{\lambda}$ used in the optical flow solution. This inclusion is for presentation only, because, as will be seen later, $\boldsymbol{\lambda}$ is effectively not part of the state. The discrete update equation for the state is given by (5.12).

$\mathbf{z}$ consists of the parameter forces $\mathbf{f}_\mathbf{q}$ and the temporal image derivatives $\mathbf{I}_\mathrm{t}$. Note that the spatial image derivatives are not included in the measurements (even though they are used in the formation of $\mathbf{B}$); doing so would greatly complicate the measurement equations. Similar simplifications can be found in image-based optical flow techniques [SAH91] where the noise in the spatial image derivatives are ignored to provide a Gaussian solution. Reasonably accurate estimates of the spatial image derivatives are usually available (especially away from occlusion boundaries), making this a fairly safe assumption. Also note that $\mathbf{H}$ depends on the state $\mathbf{y}$, which makes the measurement equation non-linear. Because of this non-linear dependency, the filter is an extended Kalman filter.

The pseudo-inverse of $\mathbf{H}$ produces the same solution as (5.11):

$$\mathbf{H}^+ = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} - \mathbf{B}^+\mathbf{B} & \mathbf{B}^+ \\ \mathbf{0} & (\mathbf{B}^+)^\top & -(\mathbf{B}^+)^\top \mathbf{B}^+ \end{bmatrix} \tag{5.15}$$

In contrast, the smaller system which does not include the Lagrange multiplers $\boldsymbol{\lambda}$ in the state produces a different solution from (5.11):

$$\begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{B} \end{bmatrix}^+ = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & (\mathbf{B}^\top\mathbf{B}+\mathbf{1})^{-1} & (\mathbf{B}^\top\mathbf{B}+\mathbf{1})^{-1}\mathbf{B}^\top \end{bmatrix} \tag{5.16}$$

Although the solution of the smaller system corresponding to (5.16) approaches that of $\dot{\mathbf{q}}$ in (5.15) if all of the non-zero eigenvalues of $\mathbf{B}^\top\mathbf{B}$ are much greater than 1, this is not the case in the applications presented here (where all actually tend to be much less than 1).

The inclusion of $\boldsymbol{\lambda}$ in (5.14) thus ensures the system reduces to the original unfiltered solution, but some complications arise as well. The presence of $\boldsymbol{\lambda}$ is a result of the constraints on the dynamic system—it should not be considered part of the state. Each λ_j in $\boldsymbol{\lambda}$ is associated with a particular pixel from the optical flow computation. However, there is not necessarily any correspondence between the pixels (and hence the λ_j) across iterations. Even worse, the number of pixels used (the dimension of $\boldsymbol{\lambda}$) varies across iterations. This means a subset of the state parameters are only present at one iteration, and their predicted values at time t are not based on any previously estimated values. An alternative interpretation would be to view these parameters $\boldsymbol{\lambda}$ as having infinite observation noise, or perhaps that the "observability" of $\boldsymbol{\lambda}$ is changing.

The *discrete process* equation for the Kalman filter gives an expression for the prediction of the state $\mathbf{y}(t+1)$ given the previous estimate $\mathbf{y}(t)$. In this case, this equation states that the predicted motion of the observed subject is the same as in the previous iteration, along with the added noise $\mathbf{w}$ (assumed to be independent zero-mean Gaussian noise with covariance $\mathbf{Q}$) to form the primarily data-driven system:

$$\mathbf{y}(t+1) = \mathbf{y}(t) + \mathbf{w}(t) \qquad p(\mathbf{w}) \sim N(\mathbf{0}, \mathbf{Q}) \tag{5.17}$$

The prior estimates of $\mathbf{y}$ and $\mathbf{P}$ used in the computation of the estimated state and covariance at time t are denoted $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{P}}$. Since $\boldsymbol{\lambda}$ is treated as a distinct value each iteration, only the portions of $\tilde{\mathbf{y}}(t-1)$ and $\tilde{\mathbf{P}}(t-1)$ that correspond to $\dot{\mathbf{q}}$ are retained, resulting in:

$$\begin{aligned} \tilde{\mathbf{y}}(t) &= \begin{pmatrix} \dot{\mathbf{q}}(t-1) \\ \mathbf{0} \end{pmatrix}, \\ \tilde{\mathbf{P}}(t) &= \begin{bmatrix} \mathbf{P}_{\dot{\mathbf{q}}}(t-1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \mathbf{Q}(t-1) \end{aligned} \tag{5.18}$$

(where $\mathbf{P}_{\dot{\mathbf{q}}}$ is the block of $\mathbf{P}(t-1)$ corresponding to $\dot{\mathbf{q}}$).

Computing the estimated mean and covariance of $\mathbf{y}$ involves forming the Kalman gain matrix, which is used to combine the solution using the current measurements with the solution from the previous iteration. In the following filtering equations, all quantities are taken at time t, but this dependence is omitted to improve readability. The Kalman gain matrix [Gel74] is computed as:

$$\mathbf{K} = \tilde{\mathbf{P}}\mathbf{H}^\top \left(\mathbf{H}\tilde{\mathbf{P}}\mathbf{H}^\top + \mathbf{R}\right)^{-1} \tag{5.19}$$

The covariance matrix $\mathbf{R}$ is computed as the sum of terms resulting from the noise in $\mathbf{f_q}$ and $\mathbf{I}_\mathrm{t}$:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{\mathbf{f_q}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathbf{I}_\mathrm{t}} \end{bmatrix} \tag{5.20}$$

The addition of (5.20) ensures the invertibility of the matrix $(\mathbf{H}\tilde{\mathbf{P}}\mathbf{H}^{\top} + \mathbf{R})$ in (5.19); this matrix is also symmetric and positive definite.

Using $\mathbf{K}$, the estimated mean is computed as a sum of the current solution $\mathbf{Kz}$ and the weighted prior mean estimate $\tilde{\mathbf{y}}$, or as the sum of the prior estimate $\tilde{\mathbf{y}}$ and the innovation $(\mathbf{z} - \mathbf{H}\tilde{\mathbf{y}})$ weighted by $\mathbf{K}$:

$$\mathbf{y} = \mathbf{Kz} + (\mathbf{1} - \mathbf{KH})\tilde{\mathbf{y}} = \tilde{\mathbf{y}} + \mathbf{K}(\mathbf{z} - \mathbf{H}\tilde{\mathbf{y}}) \tag{5.21}$$

It is easily verified that (5.21) corresponds exactly with the original solution for $\dot{\mathbf{q}}$ in (5.11) when $\mathbf{R} = \mathbf{0}$ and $\tilde{\mathbf{P}} = \mathbf{1}$ so that $\mathbf{K} = \mathbf{H}^{+}$. The estimated covariance [Gel74] is computed from the prior covariance $\tilde{\mathbf{P}}$ as:

$$\mathbf{P} = (\mathbf{1} - \mathbf{KH})\tilde{\mathbf{P}} \tag{5.22}$$

The Kalman filtered solution has a number of advantages over the direct solution from (5.11). Primarily, it makes the framework more robust to noise and small estimation errors. Additionally, it provides a useful means for the combination of the edge forces and optical flow information; the optical flow constraint is now relaxed to a degree based on the error in the optical flow information. The filtered solution includes a control for tuning how much trust goes into the optical flow information relative to the edge information; this control is the relative scale between $\mathbf{R}_{\mathbf{f}_{\mathbf{q}}}$ and $\mathbf{R}_{\mathbf{I}_{\mathrm{t}}}$. Finally, the estimation of the static quantity $\mathbf{q}_{\mathrm{b}}$ will eventually cease as the estimated variance of these parameters converges.

5.4 Feature selection

The construction of the optical flow constraint on $\dot{\mathbf{q}}_{\mathrm{m}}$ required the selection of a set of image pixels from which to measure optical flow information. While it would be possible to use all pixels on the observed object, this would have two problems. Most obviously, it would be expensive to solve the system—it is especially wasteful since it is likely that most pixels do not provide a significant amount of useful information. And secondly, particular points actually provide harmful information—such as those near occlusion boundaries. This section describes our method for the selection of pixels in the construction of (5.5).

Tomasi and Shi [ST94] define good features for tracking by using the following criterion. The outer product of the image gradients at pixel i is summed over a small window around that pixel:

$$\sum_{\text{window}(i)} \nabla \mathrm{I}_i \, \nabla \mathrm{I}_i^\top \tag{5.23}$$

A feature is selected when the smaller eigenvalue of this 2×2 matrix is greater than a threshold value. These features possess significant image gradients in two orthogonal directions, which makes them reliable tracking features, as well as good sources of optical flow information. Features with one very large eigenvalue are also useful in our application, as these image points also provide good optical flow information.

However, not all pixels with significant gradient magnitude should be chosen. In particular, pixels on occlusion boundaries must be avoided, as they violate the optical flow constraint equation. The use of model-based techniques here provides a straightforward solution—assuming the model is at least roughly aligned with the image, pixels anywhere nearby the predicted occlusion boundaries of the model are simply not chosen.

Besides providing the most accurate information possible, the set of chosen points must also adequately sample the facial motion information present in the image. The accurate measurement of a parameter in $\mathbf{q}_\mathrm{m}$ requires a sufficient number of pixels in the image corresponding to model points where the Jacobian of that parameter does not vanish. Note that some motion parameters are defined only over a particular region of the face (such as the mouth-opening motion in Figure 5.1(d) which is non-zero only in the jaw region).

Using too few pixels in the computation results in a loss of accuracy, and can reach the point where the system loses track of the subject. Including too many pixels forces the pixel selection method to include pixels containing little useful information (such as having a small gradient magnitude). It has been determined by experimentation that 10 to 20 pixels per parameter provide sufficient accuracy and robustness for the application of face tracking (at which point the results change negligibly when more pixels are used). Since there can be considerable overlap between the sets of pixels used to measure each parameter, the total number of pixels used can be fairly small. For each of the experiments here, *n* is approximately 120 pixels.

5.5 Discussion

The successful tracking performed by this framework is primarily due to the use of optical flow as a constraint. This was verified to some degree by disabling key components of our tracking system, and observing the resulting performance decrease. Altering (5.11) to use the system in (5.16) (which actually corresponds to standard Kalman filter data fusion), $\mathbf{f}_{\mathbf{q}_\mathrm{m}}$ is no longer scaled by the constraint projection matrix, and this effectively disables constraint enforcement. This produced a much less robust system—especially when many motion parameters were active. Perhaps the constraint enforcement made the edge force optimization problem simpler by projecting away components that would result in local minima. Further investigation on this point is needed.

Using an ordered solution, which alternatively uses the optical flow solution (for a big step), and a edge force solution (for a smaller correction), produced a system which failed quite frequently. Since at each step, the solution depended on a single (perhaps very noisy) source of information, the solution no longer depended solely on the useful component of each information source.

When edge forces are disabled (for $\mathbf{q}_\mathrm{m}$ only), errors in the estimation of $\dot{\mathbf{q}}_\mathrm{m}$ accumulate, causing the model to eventually lose track (a single 60 degree head turn is easily enough to do this). Using only edge forces (and no optical flow information) produces a harder and more expensive problem. Edge forces are most effective when the model is very close to the solution, and require many iterations otherwise. Large changes in many parameters at once often results in a local minimum solution being found.

While using optical flow as a constraint is a clear advantage, the presence of noise in the optical flow information makes having a strictly enforced constraint counterproductive. Canceling the entire component of the edge force which violates the constraint also throws away some potentially useful information. The extended Kalman filter allows for the softening of this constraint based on the reliability of the optical flow information. The addition of edge forces takes into account this reliability, so that the filter weights the edge forces more when the optical flow information is less reliable.

5.6 Summary

This chapter has described a deformable model framework which treats optical flow information as a constraint on the motion of the model. When combined with edge information, the estimation results are greatly improved. This use of edge information combats error accumulation in tracking, as well as allows for the extraction of shape information. The presence of an extended Kalman filter helps deal with noisy input while also providing a useful means for combining the optical flow and edge information. The use of a three-dimensional model accounts for the self-occlusion of the face, and hence permits the tracking of a subject under large amounts of head rotation.

Chapter 6

Error residual minimization

The face model described in Chapter 4 includes an intuitive distinction between shape and motion. The model has motion parameters, which describe both rigid and non-rigid motions, and shape parameters, which describe the basic underlying shape of the model. The purpose of this distinction is to reduce the number of motion parameters. This distinction now leads us to develop a method, initially presented in [DM98], where changes in the image are initially attributed entirely to motion, but then the error in the reconstructed motion is used to more accurately extract both shape and motion parameters of the object being tracked.

This formulation is used in concert with the tracking framework from Chapter 5. In this chapter, we extend this framework so that the face shape is updated also based on the optical flow information. Derivatives of the model Jacobian (second derivatives of the model) determine how changes in the parameters of the model affect its motion parameterization. Using these derivatives in a truncated Taylor series expansion, the model parameters (both shape and motion) are refined by minimizing the residuals from the model-based motion computation. This method simultaneously corrects the shape and motion parameters for each image frame.

For every image in the sequence, we first solve a model-based least squares optical flow solution, which determines the motion parameters. Then, the residual from this computation determines the error in the model parameters using another least squares process, which adjusts the shape and motion parameters of the model. The use of residuals to determine the applicability of a model's assumptions is the subject of regression diagnostics [Bel80]. The method here, however, assumes the model is appropriate, and instead uses the deviations from the model to improve the estimate.

This approach allows a more accurate extraction of the shape and motion. The estimation framework presented in the previous chapter extracted the basic shape of the face using only edge information. Edge information is not always adequate due to poor illumination and self-occlusion. This may result in inaccurate estimation of the basic shape, which can in turn cause error in the motion estimation. This approach also differs from other model-based shape and motion estimation methods [Koc93] where optical flow information was used to directly improve the shape, leading to potentially large shape estimation errors. Our method does not require the extraction of tracked features, but instead uses motion information–in this case, optical flow information. Shape and motion are improved simultaneously.

6.1 Shape and motion estimation

This section describes our new technique for non-rigid shape and motion estimation using the residuals from a least-squares motion estimation. When optical flow is used as the cue for motion estimation, as in Section 5.2, the residuals are in part caused by violations of the optical flow constraints (i.e. specularity), by linearization of the optical flow constraints, and by measurement noise. In a model-based framework, residuals are also produced by errors in the extracted shape and motion of the model. In order for the residuals to be useful, however, a significant error in the shape and motion during tracking must be responsible for the majority of the residual—this is our primary assumption. This assumption is supported by experimental evidence discussed in Section 7.1.3.

The use of a model allows for a model-based computation using these residuals. For the applications here, the deformable face model described in Chapter 4 is used. The optical flow least-squares residuals $\mathcal{R}$ are computed from (5.6):

$$\mathcal{R} = \mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}} = \mathbf{B}(-\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}}) + \mathbf{I}_{\mathrm{t}} = \left(\mathbf{1} - \mathbf{B}\mathbf{B}^{+}\right)\mathbf{I}_{\mathrm{t}} \tag{6.1}$$

The residual is a vector which has dimension *n* (the number of pixels used in the motion computation).

There are a number of approaches to using this residual information–given the assumption above, the goal of these approaches will be to *reduce* this residual. One possible approach is to extract shape information using the same formulation for determining motion as described in Section 5.2, as in:

$$\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{B}_{\mathrm{b}}\dot{\mathbf{q}}_{\mathrm{b}} + \mathbf{I}_{\mathrm{t}} = \mathbf{0}\ . \tag{6.2}$$

where the construction of $\mathbf{B}_{\mathrm{b}}$ is analogous to $\mathbf{B}$, but uses $\mathbf{L}_{\mathrm{b}}$ instead of $\mathbf{L}_{\mathrm{m}}$. The system in (6.2) is decoupled, and is solved for motion first, and then for shape in terms of the residual $\mathcal{R}$:

$$\mathbf{B}_{\mathrm{b}}\dot{\mathbf{q}}_{\mathrm{b}} = -\mathcal{R} \quad \Rightarrow \quad \dot{\mathbf{q}}_{\mathrm{b}} = -\mathbf{B}_{\mathrm{b}}^{+}\mathcal{R} \tag{6.3}$$

This method is closely related to the method described by Koch [Koc93]. It is a reasonable approach in the context of image coding, where image fidelity is of much greater importance than accuracy of face shape estimation—the face shape is deformed to account for the tracking errors in motion. This produces a face shape that results in a higher quality image, but does not necessarily estimate the actual 3-D face shape of the subject.

As stated earlier, in the framework presented here, a clear distinction is made between shape and motion parameters, since the true value of $\mathbf{q}_{\mathrm{b}}$ is a static quantity. Hence, it does not make sense to adjust the shape parameters $\mathbf{q}_{\mathrm{b}}$ directly from observed velocities, as in [Koc93].

Instead of this, our approach is to find what small change in $\mathbf{q}$ would affect the largest reduction in the motion residual. This approach uses the fact that the model Jacobian

$\mathbf{L}_{\mathrm{m}p}(\mathbf{u};\mathbf{q})$ depends on *both* $\mathbf{q}_{\mathrm{b}}$ and $\mathbf{q}_{\mathrm{m}}$ (based on how the model was constructed), so that second derivative information is employed. Let $\Delta\mathbf{q}$ be the current deviation of $\mathbf{q}$ from its true value (not including the motion in $\dot{\mathbf{q}}_{\mathrm{m}}$)—this includes both the shape error and the accumulated motion error. We assume $\Delta\mathbf{q}$ is of sufficiently small magnitude so that the first-order approximation to $\mathbf{L}_{\mathrm{m}}$ using its Taylor-series expansion is sufficiently accurate:

$$\mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i;\mathbf{q}+\Delta\mathbf{q}) \approx \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i;\mathbf{q}) + \frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i;\mathbf{q})}{\partial \mathbf{q}}\Delta\mathbf{q} \tag{6.4}$$

For the case of the face model described in Chapter 4, whose parameterization consists of mostly affine scaling deformations, sufficient linearization accuracy is easily attained. Combining this approximation of $\mathbf{L}_{\mathrm{m}p}$ with the model-based optical flow constraint equation (2.15) results in:

$$\nabla \mathrm{I}_i \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i)\dot{\mathbf{q}}_{\mathrm{m}} + \nabla \mathrm{I}_i \left(\frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_i)}{\partial \mathbf{q}}\Delta\mathbf{q} \right) \dot{\mathbf{q}}_{\mathrm{m}} + \mathrm{I}_{t_i} = 0 \tag{6.5}$$

where $\partial\mathbf{L}_{\mathrm{m}p}/\partial\mathbf{q}$ is part of the model Hessian matrix (a rank 3 tensor). It is used here as a block matrix, written here "curried" with $\Delta\mathbf{q}$ to keep the notation under control (so that the parenthesized sub-expression here is a matrix). The value of $\dot{\mathbf{q}}_{\mathrm{m}}$ is taken as $-\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}}$ as in (6.1); its instance here does not include any other terms (such as edge forces), since they do not affect the residual.

When (6.5) is considered over *n* pixels from the input image, this results in the system:

$$\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + (\mathbf{G}\dot{\mathbf{q}}_{\mathrm{m}})\Delta\mathbf{q} + \mathbf{I}_{\mathrm{t}} = \mathbf{0}\ . \tag{6.6}$$

$$\text{where}\ \ \mathbf{G} = \begin{bmatrix} \left(\nabla \mathrm{I}_1 \frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_1)}{\partial \mathbf{q}}\right)^{\top} \\ \vdots \\ \left(\nabla \mathrm{I}_m \frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_m)}{\partial \mathbf{q}}\right)^{\top} \end{bmatrix} \tag{6.7}$$

The transpositions performed in the construction of $\mathbf{G}$ [1] allow it now to be curried with $\dot{\mathbf{q}}_{\mathrm{m}}$ (this construction transposes the second and third indices for the tensor $\mathbf{G}$). This manipulation allows for the solution of $\Delta\mathbf{q}$, which is found using another least-squares process, given by the equation:

$$(\mathbf{G}\dot{\mathbf{q}}_{\mathrm{m}})\Delta\mathbf{q} = -(\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}}) \tag{6.8}$$

which can be manipulated by substituting $\dot{\mathbf{q}}_{\mathrm{m}} = -\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}}$ and $R = \mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}}$, and then solved:

$$(\mathbf{G}\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}})\Delta\mathbf{q} = R \quad \Rightarrow \quad \Delta\mathbf{q} = (\mathbf{G}\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}})^{+}R \tag{6.9}$$

This least squares solution determines the best set of small changes in $\mathbf{q}_{\mathrm{b}}$ and $\mathbf{q}_{\mathrm{m}}$ that minimize the optical flow residual (6.1), given the linearization of $\mathbf{L}_{\mathrm{m}p}$ in (6.4).

6.2 Solution improvement

The value of $\Delta\mathbf{q}$ from the previous section specifies an absolute update to the state (unrelated to the current timestep Δt)—$\Delta\mathbf{q}$ is simply added to $\mathbf{q}$ after each iteration.

The solution (5.11) from Chapter 5 must be adjusted to accommodate this added term. This involves determining and evaluating the edge forces using the model at the updated location $(\mathbf{q} + \Delta\mathbf{q})$. This greatly reduces overshooting, which would be caused by edge forces contributing corrections which are redundant with those already present in $\Delta\mathbf{q}$. The new system is:

$$\dot{\mathbf{q}}_{\mathrm{b}} = \mathbf{f}_{\mathbf{q}_{\mathrm{b}}}(\mathbf{q} + \Delta\mathbf{q}), \qquad \dot{\mathbf{q}}_{\mathrm{m}} = -\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}} + \left[\mathbf{1} - \mathbf{B}^{+}\mathbf{B}\right]\mathbf{f}_{\mathbf{q}_{\mathrm{m}}}(\mathbf{q} + \Delta\mathbf{q}) \tag{6.10}$$

which is updated over time similarly to (5.12), but with $\Delta\mathbf{q}$ added in:

$$\mathbf{q}(t+1) = \mathbf{q}(t) + \dot{\mathbf{q}}\Delta t + \Delta\mathbf{q} \tag{6.11}$$

Analogous changes can be made to the Kalman filtered solution: the force determination is made at the improved state, and the improvement is added in after each iteration (unfiltered).

[1] The matrix $\mathbf{G}$ is the same as $\mathbf{H}$ in [DM98], but is changed here to avoid overloading with the measurement matrix from Chapter 5.

6.3 Implementation

Due to the linear approximation in (6.4), it is important to determine if the residual actually does decrease with the addition of $\Delta\mathbf{q}$. Once $\Delta\mathbf{q}$ has been computed using (6.9), the model-based motion analysis in (5.6) is re-solved using $\mathbf{q}_{\text{new}} = \mathbf{q} + \Delta\mathbf{q}$, producing an updated residual $\mathcal{R}_{\text{new}}$. If the addition of $\Delta\mathbf{q}$ causes the residual magnitude of $\mathcal{R}_{\text{new}}$ to be larger than $\mathcal{R}$, the results of the shape and motion refinement are discarded ($\Delta\mathbf{q}$ is set to zero). Otherwise, the changes specified by $\Delta\mathbf{q}$ can be used directly. Note that this process does not include any edge forces, since they do not affect the residual.

The efficiency of solving this system is improved by omitting parameters in the construction of $\mathbf{G}$ from (6.7) which cannot be affected based on $\mathbf{q}_{\text{m}}$. For example, if there is no motion extracted in the eyebrow region of the face, then there is no reason to include eyebrow shape parameters in $\mathbf{G}$. At any point in time, typically about half of the shape parameters of the face model can be omitted from the computations.

The process of determining $\Delta\mathbf{q}$ can also be iterated, solving (5.6) and (6.9) repeatedly to obtain a greater improvement. For the applications here, the linear approximation in (6.4) is relatively accurate for the face model described in Chapter 4, due to the fact that most of the model parameterization is linear scaling. As a result, only the single iteration is performed.

The least squares solution to (6.9) is solved using a singular-value decomposition. This avoids any problems associated with the lowering of rank due to the aperture problem or a lack of motion, as well as the problems associated with a non-orthogonal set of parameters.

6.4 Summary

We have presented a novel deformable-model technique which uses residuals from a model-based optical flow solution to refine the shape and motion of the model. By using second derivative information from the model, small improvements to the parameters are made by minimizing the residuals. Besides having greater accuracy than a framework using only optical flow and edges, our framework extracts the shape of the face without needing data from extreme head poses (such as a profile view). Instead, much smaller motions are needed to extract much of the shape information.

Chapter 7

Experiments and Results

7.1 Vision Experiments

This section contains the results from a series of face shape and motion estimation experiments. The first three experiments exhibit the generality of our tracking system (from Chapter 5) on a variety of subjects, while the next six experiments use a common observed subject, and provide a quantitative validation of the shape and motion estimation systems (from both Chapter 5 and Chapter 6).

7.1.1 Initialization

The entire estimation process is automatic, except for the initialization, which requires the manual specification of several landmark features in the first frame of the sequence (the eyebrow centers, eye corners, nose tip, and mouth corners). The subject must also be at rest, and (approximately) facing forward, as in Figure 7.1(a).

Using these marked features, forces are applied to the initial face model (described in Section 2.3) that deform the corresponding points on the face toward the desired locations in the image. The rotation and translation, as well as course-scale face shape parameters (such as those which determine the positions and sizes of the face parts) are fitted using this information, the result of which is shown in Figure 7.1(b). Once roughly in place, both edge and anthropometry forces are applied that pull the face into the correct shape as in Figure 7.1(c). The distance from the initial face to the camera is determined given the assumption that the subject's face is the same size as the model.

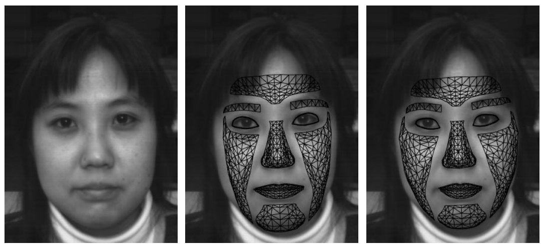


Figure 7.1: Model initialization

The problem of automatically locating the face and its various features has been addressed elsewhere [YD94, YCH92], and could be used to make this process automatic. No markers or make-up are used on the subject (markers are used for the validation of the tracking method, however, as described below). Experience has shown that the initialization process is robust to small displacements (i.e. several pixels) in the selected landmark points.

7.1.2 Tracking experiments

The original image sequences are 8 bit grey images at NTSC resolution (480 vertical lines). In each of the sequences, the width of the face in the image averages 200 pixels, and the range of motion of features across the image sequence is typically 80 to 100 pixels. For each of the tracking examples, several frames from the image sequence are displayed, cropped appropriately. Below each, the same sequence is shown with the estimated face superimposed. In each case, a model initialization is performed as described above. The initialization process usually takes about 2 minutes of computation. Afterwards, processing each frame (using the extended Kalman filter formulation) takes approximately 1.4 seconds each (all computation times are measured on a 175 MHz R10000 SGI O2). When using the error residual computation, processing each frame takes an additional 8 seconds.

The sequence shown in Figure 7.2 was taken on an IndyCam at 5 fps. Figure 7.2 shows a subject turning her head in (a) through (d) and opening her mouth from (d) to (f). Based on the good alignment of the face model with the image, it appears the face model is able to capture the shape of her face, as well as the head rotation and mouth motion. The next two sequences were taken on a higher quality camera at 30 fps[1]. Both Figure 7.3 and Figure 7.4 show a subject smiling and moving forward in (b) and (c), opening their mouth while turning their head in (e) and (f), and turning back, closing their mouth slightly in (g). All of these motions appear to be correctly tracked based on the observed motion. These three experiments involve different subjects, having very different appearances. This suggests the verification of the face model shape parameterization (described in Section 4.3.1) was successful.

7.1.3 Shape estimation validation experiments

The same observed subject is used in both experiments presented here, which provide a validation of the shape estimation accuracy of our system. The shape (determined by $\mathbf{q_s}$) is validated using a Cyberware range scan of the subject, shown in Figure 7.5(a). Experiments using the edge-based shape estimation from Chapter 5 are compared along side with results using the error residuals from Chapter 6.

The shape estimation validation experiment in Figure 7.7 shows the subject making a series of non-rigid face motions: opening his mouth in (b) and (c), smiling in (d) through (e), and finally raising his eyebrows in (f). In each case, the motion parameter values change appropriately, and at the correct times (both techniques extracted virtually the same motion parameter values).

At each frame, Figure 7.8 shows the extracted shape results as compared against the range scan of the subject, for both techniques. Note that for this comparison, all motion parameters are ignored, so that only the shape is compared. The RMS error is computed

[1]We are grateful to Yaser Yacoob and the Center for Automation Research at the University of Maryland College Park for providing these two image sequences.

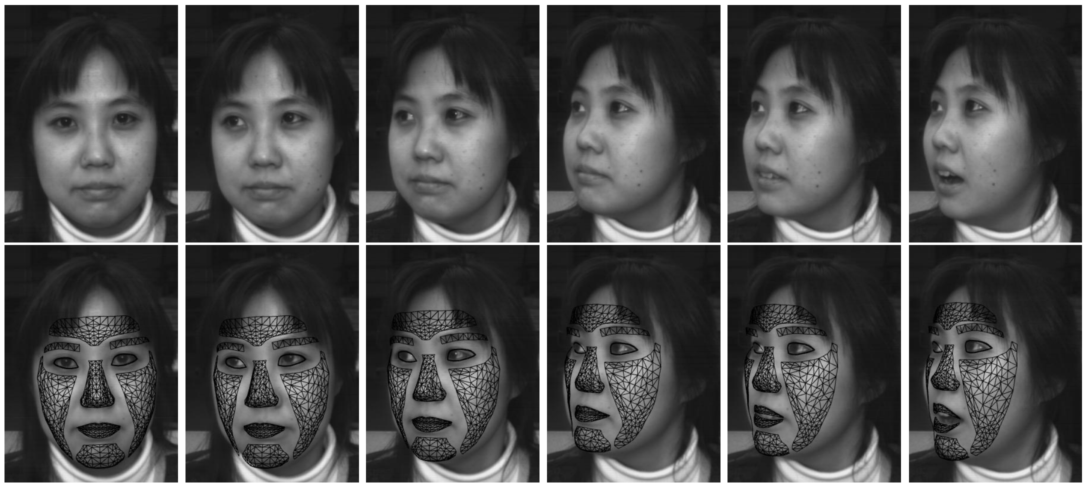

(a) (b) (c) (d) (e) (f)

Figure 7.2: Motion and expression tracking example 1

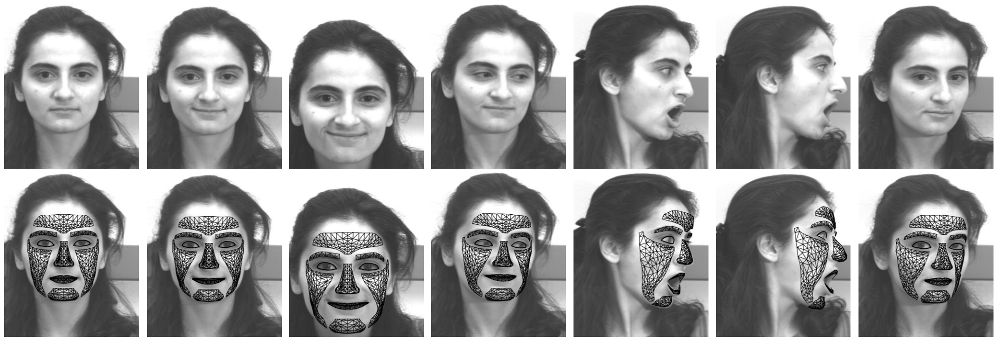

(a) (b) (c) (d) (e) (f) (g)

Figure 7.3: Motion and expression tracking example 2

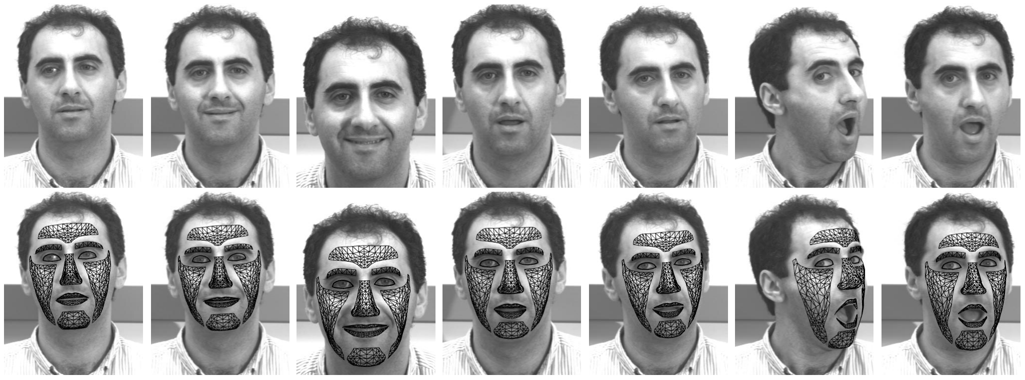

(a) (b) (c) (d) (e) (f) (g)

Figure 7.4: Motion and expression tracking example 3

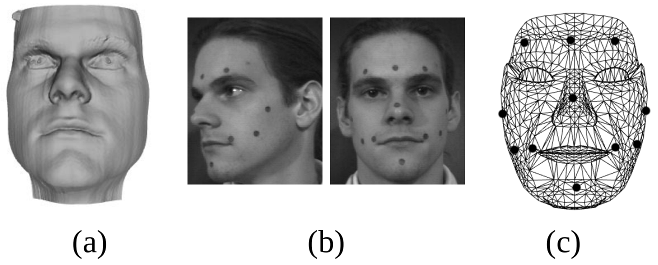


Figure 7.5: (a) Shaded range scan of subject, (b) Marker calibration images, (c) Resulting marked model

using the nodes of the model, and also includes a uniform scaling of the model so that the two faces are the same scale (this eliminates the depth ambiguity—in this case, the estimated model was compared at 96% scale).

For the edge-based estimation system (the dotted line), the RMS error starts at around 1.7 cm after initialization, and shows a steady reduction over the course of the experiment, ending around 1.3 cm. For the system using error residuals (the solid line), the RMS error again starts at around 1.7 cm, but ends with less error (0.85 cm) compared to the edge-based technique.

The experiment in Figure 7.9 shows the subject performing small head motions in (a) through (f) while smiling in (c) and (d), and finishing with a significant head rotation in (g). Using the edge-based method (again, the dotted line), the RMS error starts at around 1.9 cm after initialization, shows a gradual reduction over the course of the experiment, ending just under 1 cm, with the large reduction in error around frame 50 corresponding to when the subject turned his head significantly to the side in Figure 7.9(f) and (g), where the profile view contained good edge information to fit the face shape. For the system using error residuals (the solid line), the RMS error again starts at around 1.9 cm, but this time finishes with just under half of the RMS error as the edge-based technique: around 0.4 cm. In addition, this lower level was reached fairly quickly, showing the advantage of using the error residual technique.

Besides estimating the shape more accurately, the technique using the optical flow error residuals also estimates expression-shape parameters. This allows the extraction of the correct curve of the smile expression for this subject, as in Figure 7.6(a), compared to treating these values as constants, as was the case for edge-based fitting, which is shown in Figure 7.6(b).

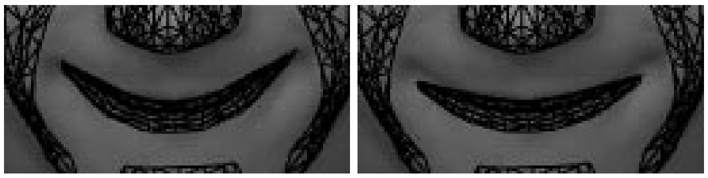
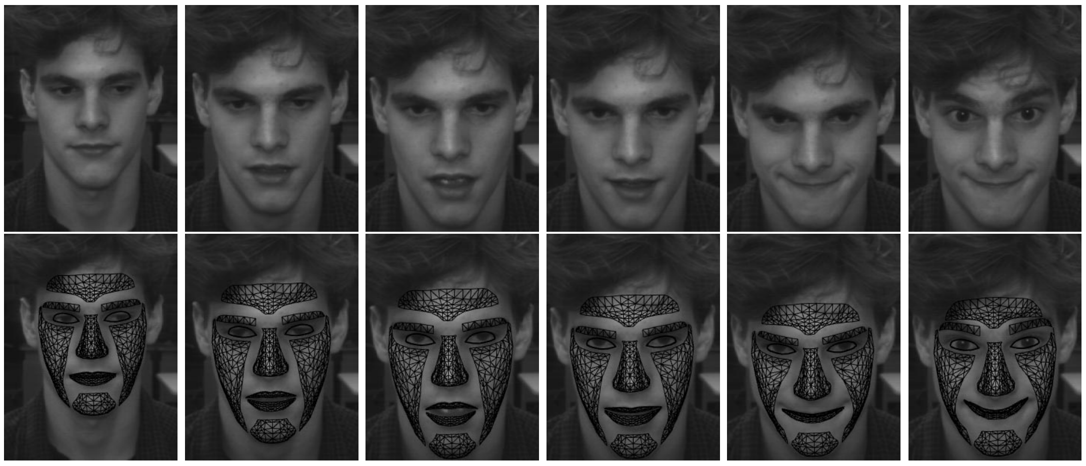

(a) (b)

Figure 7.6: Fitting expression-shape parameters (a) using error residuals; using average value (b) with edge-based estimation

The derivation of the method using the residuals in Section 6.1 assumes that shape error is the leading contributor to the residuals from the motion computation. During the experiments, the residual magnitudes started fairly high (initially around 0.18 for the first experiment, and 0.24 for the second), and ended up around 0.050 (for both experiments) by the end of motion sequence (this is for the residual-based method). (Note that these values are the magnitude of R , and is not a shape difference measure). In order to estimate what portion of the residuals are caused by shape error, both experiments were run again (for the residual-based method only); this time, the initial model shape was taken from the range scan of the subject (so that shape error is eliminated). The residuals that resulted from these experiments had a fairly small and constant magnitude, which averaged around 0.035 (pixel intensity units—for pixels in the range $[0,1]$). This enforces the validity of our assumption that shape error is responsible for the bulk of the residual.

7.1.4 Tracking validation experiments

The next four experiments use markers to allow for the validation of the motion tracking of the techniques from Chapter 5 and Chapter 6. The same subject is used in each of the experiments. Eleven small circular markers were placed on the face of a subject. Analysis of the accuracy of the motion estimation in $\mathbf{q}_{\mathrm{m}}$ is performed using these markers on the



(a) Frame 1 (b) Frame 9 (c) Frame 13 (d) Frame 20 (e) Frame 24 (f) Frame 32

Figure 7.7: Shape validation experiment 1

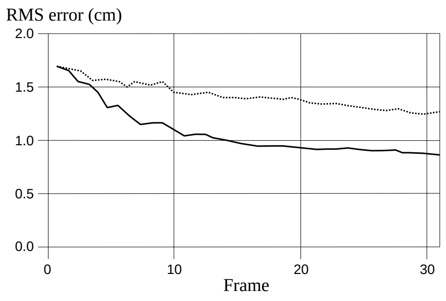
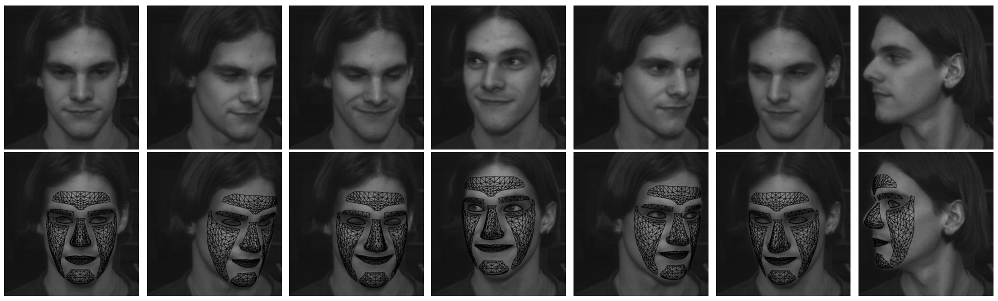


Figure 7.8: Results of shape validation experiment 1



(a) Frame 1 (b) Frame 11 (c) Frame 18 (d) Frame 24 (e) Frame 35 (f) Frame 46 (g) Frame 57

Figure 7.9: Shape validation experiment 2

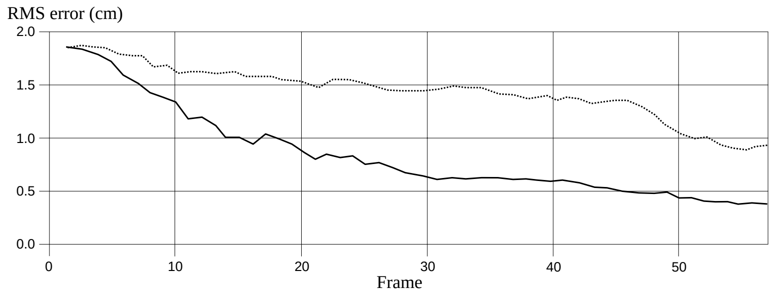


Figure 7.10: Results of shape validation experiment 2

subject, which allow for alignment verification in the image plane (ground truth motion in 3-D is not available).

For these experiments, no shape estimation is performed. Instead, the face shape is provided by an off-line fitting of the face model to the range scan in Figure 7.5(a)—this way, any deviation can be attributed primarily to motion error, not shape error. In addition, the fixed locations of the markers on the model are determined using some additional images taken of the subject, shown in Figure 7.5(b). The markers are fixed into particular locations of the polygon mesh (they have fixed coordinates in Ω). The model resulting from this fitting and marker placement is shown in Figure 7.5(c), with the marker locations shown as dark circles. The RMS error of the extracted model (comparing the extracted model with the range scan) is 0.26 cm.

First, the image locations of each of the markers from the image sequence is obtained using a semi-automatic tracking system. The rough location of the markers is tracked using the KLT[2] package (which is based on [ST94]), and was fine tuned using a deformable ellipse template. Simple calibration tests suggest this tracking technique has a variance of 0.35 pixels in measuring the center of a marker (which are usually about 8 pixels across) in the image.

Care was taken so that the presence of the markers did not significantly affect the motion estimation, since these markers could provide useful information for tracking. The pixel selection method for the optical flow information was modified so that no points were selected that were within 3 pixels (the radius of the spatial derivative filters) of any point on a marker. In addition, any edges used to produce edge forces were similarly limited to be distant from markers. Given that the markers were not placed directly on top of important facial features, it is unlikely that the presence of the markers detrimentally affected the experiment results.

In each of the following four motion validation experiments, there is an accompanying graph showing the displacement error for each frame. This displacement error of a marker is the Euclidean distance (in pixels) between the image location of the marker (if visible), and the predicted image location of the marker given the model (which is the projected image location of the model marker). The dark line on the graph shows the mean displacement error of all visible markers (one standard deviation is indicated by the gray region surrounding it). The dotted lines indicate the minimum and maximum displacement error.

The first three sequences were taken using an IndyCam at 5 fps. The final sequence was taken on a high quality camera (Pulnix TM-9701; greyscale, progressive scan) at 30 fps. Also note that this final sequence was taken at a different time than the first three—the markers were re-applied to the subject, and their locations were determined again, as in Figure 7.5(b) and (c). Their new locations were roughly the same as in the earlier validation experiments (at most 1.5 cm difference).

The sequence in Figure 7.11 shows the subject making a series of (nearly) rigid head

[2]Stan Birchfield's KLT package is available at `http://vision.stanford.edu/~birch/klt`

motions. The subject turns to his right in (a) through (c), back to his left by (e) and then faces forward in (f). The average of the deviation errors for this sequence, shown in Figure 7.12, is roughly between 2 and 3.5 pixels, which given the face is approximately 200 pixels across in the image, amounts to less than 2%. The maximum error of around 7 pixels is around 3.5% (roughly 0.5 cm).

The motion in the second sequence in Figure 7.13 is predominantly non-rigid motion (facial expressions). The subject moves forward and frowns his eyebrows in (b), moves back and produces a surprise expression in (d), followed by a smile in (f). The average error shown in Figure 7.14, is slightly higher for this experiment, averaging between 2 and 4 pixels, with a maximum again at about 7 pixels. The largest error is produced during the smile expression; possible reasons for this are discussed below.

The third sequence in Figure 7.15 is a combination of rigid and non-rigid motions. The subject turns his head from (a) through (d) while smiling, returning to rest position in (f). The displacement error shown in Figure 7.16 is also somewhat higher, averaging from 2 to 4 pixels (but being closer to 4 for a longer period), reaching a maximum of just over 7 pixels. The largest error is produced when the smile is viewed from the side, and is concentrated in the mouth area.

The last sequence in Figure 7.17 is primarily a rigid-motion sequence that is significantly longer than the other experiments (760 frames). It includes head rotations in a variety of directions, as well as some large head translation (side-to-side and away from the camera). Eyebrow raises and a smile are also present. This sequence demonstrates the ability of the system to maintain track over a long sequence, without experiencing failure due to tracking drift. In this sequence, the face is approximately 140 pixels across in the image (somewhat smaller than in the previous experiments). The average pixel deviations shown in Figure 7.18, range between 1.5 and 2.8 pixels, with a maximum error at 4.6 pixels, corresponding to about the same absolute distance error as with the previous experiments (roughly 0.5 cm). Hence, the apparently lower pixel deviations for this sequence amount to approximately the same error in actual distance. During the sequence, some of the motions were very close to the maximum limits of tracking speed (pixel velocities were about the same size as the derivative filter width). In particular, the turning motion at frames 250–320 is the most serious, with other occurrences at frames 430–450 and 610–620. These motions manifest themselves in Figure 7.18 as larger displacement errors. However, during the successive motions (which are well below this maximum velocity), the system recovers from these errors, and improves the fit using edge information, returning to the baseline deviation amount of around 2 pixels.

While the edge information prevented drift in this example, it only works to a certain extent. Should the baseline deviation be significantly higher, the alignment of the model and image can be poor enough to cause tracking failure (drift seems to be the cause). [3]

This tracking experiment was run again (a number of times) to experimentally determine the minimum baseline deviation that causes tracking failure. After each iteration, Gaussian noise was added (of increasing variance until tracking failed) to the rigid motion parameters in $\mathbf{q}_{\mathrm{m}}$. Tracking failure became common as average pixel deviation values went above 4.6 (the incidence of failure went from non-existent below 4.5, to prevalent by 4.7). Alternatively, adding Gaussian noise directly to the images (of increasing variance until tracking failed) produced a similar value (average pixel deviation of 4.4, with a corresponding image noise variance of 15.5% of intensity).

Each of these motion tracking experiments were also run using the motion improvement technique from Chapter 6 (the shape improvement was disabled, since these experiments do not perform shape estimation; however, the residual method still included shape parameters in its computation, so as to preserve the "intent" of the method).

To my initial dismay, the results were indistinguishable (under 0.1 pixel deviation difference; sometimes lower, sometimes higher) from the experiments which did not use this method. After some analysis, it became apparent that the edge forces were far surpassing any benefit the motion improvement performed. This explanation was supported by

[3]Tracking failure is simply defined as reaching a 10 pixel deviation–at this point, further tracking may still produce reasonable velocity results, but only because the head is roughly an ellipsoid; the deviation typically increases once this point is reached, with tracking being re-gained only by luck.

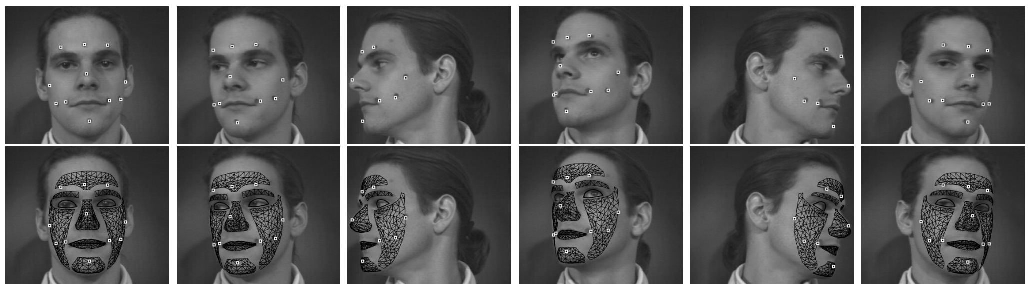

(a) Frame 1 (b) Frame 7 (c) Frame 15 (d) Frame 23 (e) Frame 41 (f) Frame 48

Figure 7.11: Motion validation experiment 1

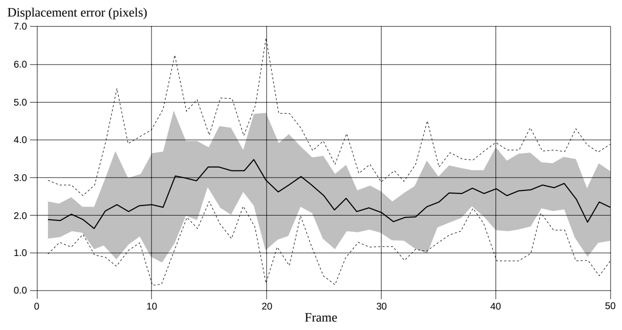
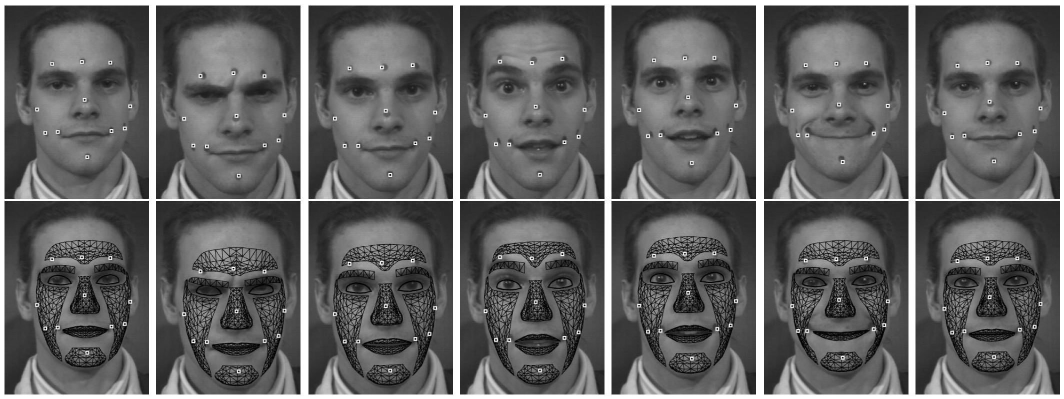


Figure 7.12: Results of motion validation experiment 1

another experiment, which involved comparing the results with and without the motion improvement method from Chapter 6; but this time, edge forces were disabled in both. In both cases, the system lost track. However, when the motion improvement method was present, it retained track until about frame 180, whereas without this improvement method, tracking lasted only until frame 120. Hence, while the motion improvement method does not seem to contribute noticeably to the results in the presence edge forces, it still nudges the system in the right direction. Of course, the shape improvement method still contributes a great deal (as seen earlier), even in the presence of edge forces.

Considering all the experiments, the error in the tracking results can have other (non-noisy) sources, besides motion estimation error. One possibility is that it can be caused by poorly extracted marker locations (although this distance is less than a pixel). Another source can be the discrepancy between the face shape used and the shape of the observed subject. The RMS error between the face shape and the range scan for *only* the marker



(a) Frame 1 (b) Frame 11 (c) Frame 16 (d) Frame 18 (e) Frame 24 (f) Frame 29 (g) Frame 40

Figure 7.13: Motion validation experiment 2

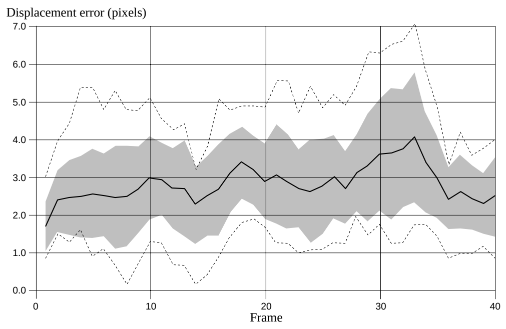
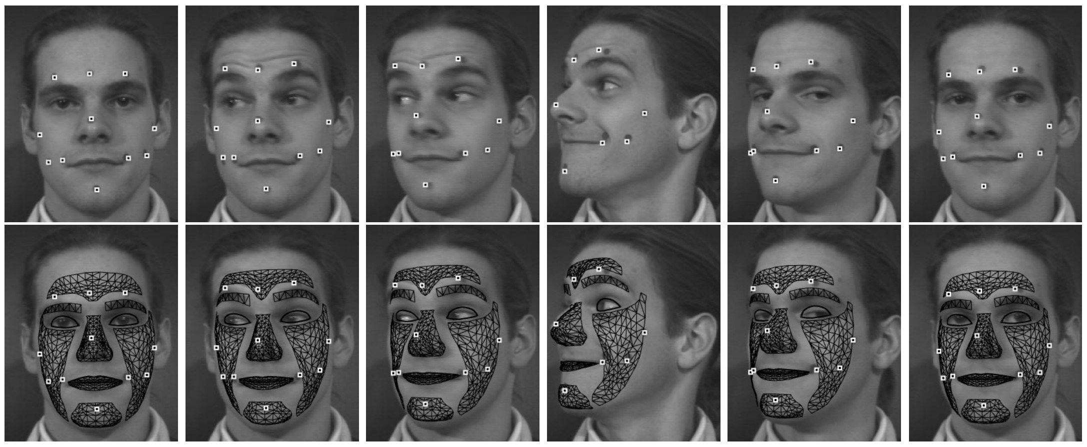


Figure 7.14: Results of motion validation experiment 2



(a) Frame 1 (b) Frame 16 (c) Frame 18 (d) Frame 27 (e) Frame 39 (f) Frame 43

Figure 7.15: Motion validation experiment 3

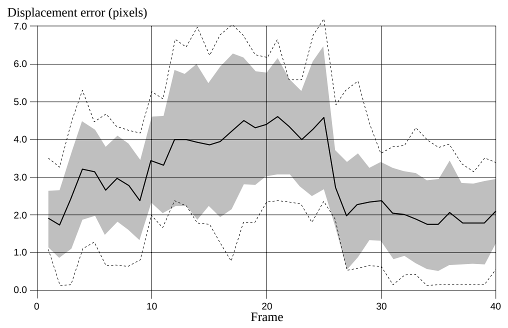


Figure 7.16: Results of motion validation experiment 3

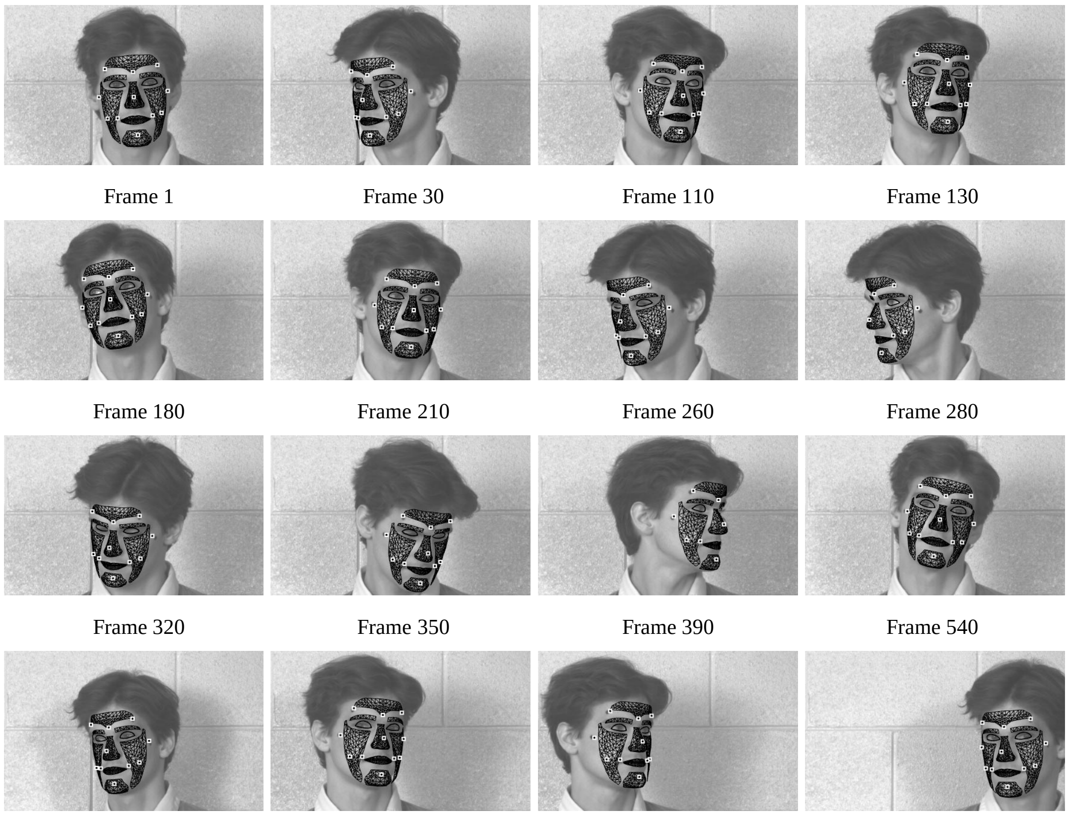


Figure 7.17: Motion validation experiment 4

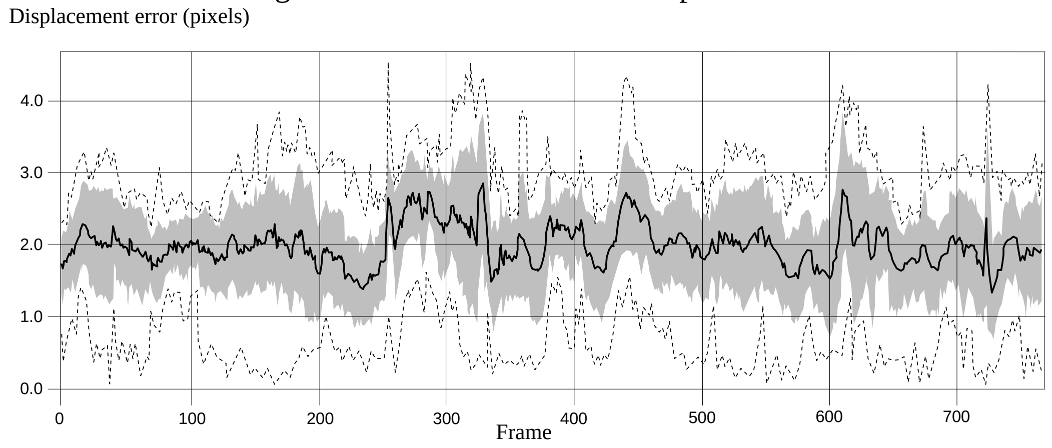


Figure 7.18: Results of motion validation experiment 4

points is much lower than that from the whole model; it is 0.1 cm, which will cause at most 1 pixel of deviation in marker locations. Violation of the assumption of perspective projection is also a possible contributor to error, although in this case is minimal, given the small depth range of the face compared to the distance of the face to the camera. From this, it can be concluded that a significant portion of the errors present here are from motion estimation.

Upon closer examination, it can be seen that the larger errors which are present during non-rigid motions (in particular, smiling), are caused by the smile produced by the model not matching the smile on the subject. Although the estimation of the expression-shape parameters will help for markers on or near the smile boundary, the model deformations do not affect the surrounding surface as much as their corresponding expressions affect the surrounding tissue.

Judging by the good performance here, it seems that both techniques are relatively insensitive to optical flow constraint equation errors (such as violations of the brightness constancy assumption [NY93], or the truncation of higher order image-derivative terms [Nag83]). Any remaining problems appear to be corrected by edge forces, which prevent drift from accumulating.

7.1.5 Limitations

The many experiments in this section show the capabilities of the shape estimation and tracking systems described in Chapter 5 and Chapter 6. On the other hand, they also say a lot about what the limitations of the system are.

First, some of the limitations of the system come directly from the assumptions made during design. Most obvious is the assumption of brightness constancy during optical flow computation. Major lighting changes can cause tracking failure. Specularities also cause small problems, but tend not to affect the entire model, since they tend to be fairly localized. In some cases, poor lighting will also lead to tracking failure. Typically, these occur in situations where edges are washed out (opening the aperture too wide on a camera will do this).

Second, is to simply exceed the maximum tracking speed (determined by the derivative filter width). This problem can be addressed simply by using multi-scale optical flow methods.

Third, are deviations from the model—where the images go past the coverage limits of the model. Attempting to track motions that are not represented produces relatively unpredictable effects. For example, lip puckering is not modeled: tracking this facial motion produces the best fit using the existing motion parameters (often quite far off). This causes poor model-image alignment, which can lead to tracking failure. Occlusions produce similar problems. There is hope for these problems—as these violations first appear as large increases in the error residual, perhaps these regions can be automatically ignored.

Finally, are the problems associated with the tracking of multiple, simultaneous motions. In the validation experiments, situations where head rotation was accompanied by a non-rigid expression deformation often produced higher pixel deviations. On occasion, this deviation can be serious enough to cause tracking failure. This is caused by the linearization in the model-based optical flow solution, which could perhaps be alleviated by using the iterative method from Section 2.2.2, or the image warping method from [BAHH92].

Chapter 8

Contributions

We have described techniques for the construction of face models for both computer graphics and computer vision applications, and describe how information gathered using these models in a vision system can be extracted and combined.

More specifically, in the construction of the face models here, the use of face anthropometry data was introduced. For computer graphics, this allowed for the generation of random individuals from a particular population. For computer vision, it improved the estimation of shape, by ensuring the likelihood that the extracted face can actually exist, and also by providing a reasonable starting point (the average face).

In addition, techniques were described for using such a face model for model-based shape and motion estimation. In particular, the use of optical flow information as a constraint on the model motion allowed for the combination of edge data with the optical flow information. This solution was sufficient to prevent tracking drift. The use of a model-based optical flow solution also resulted in a technique that improves the shape and motion estimate by reducing the error residual of this solution. Finally, a number of experiments, some of which provide validation of these techniques, have been performed. These experiments raised some interesting issues concerning the use of model-based optical flow methods.

8.1 Conclusions and Future Work

8.1.1 Face generation

The generation model presented in Chapter 3 must ultimately be more richly represented. Possible extensions might apply variational techniques to construct the face surface and the interior skull simultaneously; this would form the basis of a face animation model as in [LTK95]. Similarly, landmarks on the face could be used to drive texture synthesis, deriving distinct but plausible patterns of skin and hair.

Acquiring better data is also an avenue of improvement. In this work, proportions were used since they were the *best available* resource to model the correlations that exist between measurements. Having access to the raw data (per individual) would allow for a covariance analysis, as well as the fitting of probability distributions (since they probably aren't Gaussian).

In the meantime, our work already suggests new computational approaches for tasks that rely on anthropometric results, like forensic anthropology, plastic surgery planning, and child aging. It could also figure in a user interface for editing face models, by allowing features to be edited while related features systematically changed—preserving natural proportions or ensuring that faces respect anthropometric properties common to their population group. Both tasks underscore the importance of continuing to gather and analyze anthropometric data of diverse human populations.

8.1.2 Face shape estimation and tracking

The end of Chapter 7 described a number of limitations in the tracking system detailed in Chapter 5. The most significant of which is the idealization of the optical flow constraint equation. For instance, the problems of photometric variation and self-shadowing, which violate the optical flow constraint equation, are not addressed. The presence of a three-dimensional model could prove to be useful when addressing these problems. Another limitation is in tracking large motions; at the moment, motions larger than the width of the derivative filters will not be tracked correctly. Multi-scale optical flow techniques can be applied here, although will need to be modified to work in a model-based framework. It should also be possible to warp the current image based on the prior motion estimate, and perform a residual flow computation as an innovations process in the Kalman filtering framework. These subjects require further investigation.

Investigation of the recognition of faces using the shape parameterization, or of facial motion using the motion description is worth pursuing. And of course, additional detail in the motion parameterization of the model will allow for the tracking of more complex facial motions. This might prove more difficult than it seems, from two ends. First, the modeling (by hand) would be quite difficult; more automated approaches would be advisable. And second, it is possible that as the number of non-rigid motion parameters increases, it will become more difficult to distinguish between them. Perhaps the tracking of multiple hypotheses will be necessary to ensure model-image alignment.

Appendix A

Modularization of global deformations

The shape model $\mathbf{x}$ is defined through the repeated application of n global deformations $\mathbf{T}_k : \mathbb{R}^3 \to \mathbb{R}^3$, where $k \in 1 \ldots n$, to the underlying shape $\mathbf{s}$ as:

$$\mathbf{x}(\mathbf{q};\mathbf{u}) = \mathbf{T}_n(\mathbf{q}_{\mathbf{T}_n};\ldots\mathbf{T}_1(\mathbf{q}_{\mathbf{T}_1};\mathbf{s}(\mathbf{q}_\mathbf{s};\mathbf{u}))) \tag{A.1}$$

where $\mathbf{q}_{\mathbf{T}_k}$ are the parameters used by $\mathbf{T}_k$. The parameters used by all of the global deformations are accumulated into the vector $\mathbf{q}_\mathbf{T}$ as in:

$$\mathbf{q}_\mathbf{T} = (\mathbf{q}_{\mathbf{T}_1}^\top, \ldots, \mathbf{q}_{\mathbf{T}_n}^\top)^\top \tag{A.2}$$

so that $\mathbf{q}$ can now be grouped as:

$$\mathbf{q} = (\mathbf{q}_\mathbf{s}^\top, \mathbf{q}_\mathbf{T}^\top)^\top \tag{A.3}$$

For a particular set of deformation functions, closed form expressions for the resulting shape can be derived. From these complex expressions, the Jacobian matrix can be derived (see [MT93] for an example), although this method is tedious and non-modular.

Instead of this, a single expression for the resulting shape is not derived, but rather each deformation is applied separately given the definition in (A.1). The Jacobian matrix can be calculated in a similar way using the chain rule. First, define the deformation τ_k as the composition of the first k deformation functions $\mathbf{T}_1$ through $\mathbf{T}_k$:

$$\tau_k(\mathbf{q}_\mathbf{T};\mathbf{p}) = \mathbf{T}_k(\mathbf{q}_{\mathbf{T}_k};\ldots\mathbf{T}_1(\mathbf{q}_{\mathbf{T}_1};\mathbf{p})) \qquad \mathbf{p} \in \mathbb{R}^3, \quad k \in 1 \ldots n \tag{A.4}$$

with τ_0 defined to be the identity. Given this definition of τ_k, it follows how to compute $\mathbf{J}_\mathbf{x}$, the Jacobian of $\mathbf{x}$ with respect to $\mathbf{q}$, using the following recurrence:

$$\begin{aligned}
\mathbf{J}_{\tau_0} &= \mathbf{J}_\mathbf{s} = \frac{\partial \mathbf{s}}{\partial \mathbf{q}_\mathbf{s}} \\
\mathbf{J}_{\tau_k} &= \left[\frac{\partial \mathbf{T}_k(\mathbf{p})}{\partial \mathbf{p}} \mathbf{J}_{\tau_{k-1}} \;\middle|\; \frac{\partial \mathbf{T}_k}{\partial \mathbf{q}_{\mathbf{T}_k}} \right] \quad k \in 1 \ldots n
\end{aligned} \tag{A.5}$$

so that $\mathbf{J}_\mathbf{x} = \mathbf{J}_{\tau_n}$. The left block in (A.5) uses the chain rule, so that the matrix $\partial \mathbf{T}_k(\mathbf{p})/\partial \mathbf{p}$ "deforms" the individual columns of the Jacobian matrix $\mathbf{J}_{\tau_{k-1}}$. The right block in (A.5) contains the derivatives of the outermost deformation $\mathbf{T}_k$ with respect to its parameters.

A naive technique for computing $\mathbf{J}_\mathbf{x}$ using this recurrence from the bottom-up (which starts with $\mathbf{J}_\mathbf{s}$), is particularly expensive in terms of both time and space complexity. This is particularly a problem since the Jacobian needs to be re-evaluated at each iteration, over many points on the model. Instead, the quantity $\mathbf{J}^\top \mathbf{f}$ is computed, given an applied force $\mathbf{f}$ such as in (2.9). The quantity $\mathbf{J}^\top \mathbf{f}$ can be computed efficiently in a top-down fashion as:

$$\mathbf{f}_n = \mathbf{f}\,, \qquad \mathbf{f}_{k-1} = \left(\frac{\partial \mathbf{T}_k(\mathbf{p})}{\partial \mathbf{p}} \right)^\top \mathbf{f}_k \quad k \in 1 \ldots n \tag{A.6}$$

$$\mathbf{J}_\mathbf{s}^\top \mathbf{f} = \left(\frac{\partial \mathbf{s}}{\partial \mathbf{q}_\mathbf{s}} \right)^\top \mathbf{f}_0\,, \qquad \mathbf{J}_{\mathbf{T}_k}^\top \mathbf{f} = \left(\frac{\partial \mathbf{T}_k}{\partial \mathbf{q}_{\mathbf{T}_k}} \right)^\top \mathbf{f}_k \quad k \in 1 \ldots n \tag{A.7}$$

If the actual columns of $\mathbf{J}_\mathbf{x}$ are required, as is the case for the optical flow computation (2.15), they can be found by three applications of the above technique using the unit vectors $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$, and $\hat{\mathbf{k}}$ in the x, y and z directions, respectively, as:

$$\mathbf{J}_\mathbf{x}^\top = (\mathbf{J}_\mathbf{x}^\top \hat{\mathbf{i}})\hat{\mathbf{i}}^\top + (\mathbf{J}_\mathbf{x}^\top \hat{\mathbf{j}})\hat{\mathbf{j}}^\top + (\mathbf{J}_\mathbf{x}^\top \hat{\mathbf{k}})\hat{\mathbf{k}}^\top \tag{A.8}$$

since $\hat{\mathbf{i}}\hat{\mathbf{i}}^\top + \hat{\mathbf{j}}\hat{\mathbf{j}}^\top + \hat{\mathbf{k}}\hat{\mathbf{k}}^\top = \mathbf{1}$. For the optical flow computation, this construction is only required for the motion parameters in $\mathbf{q}_\mathrm{m}$.

Besides global deformations, it is also useful to include rigid motions (translations and rotations) and even camera projections. For the case of camera projections, however, the mapping becomes $\mathbf{T} : \mathbb{R}^3 \to \mathbb{R}^2$, and (A.8) uses only $\hat{\mathbf{i}}$ and $\hat{\mathbf{j}}$, since the image forces are two-dimensional. The formulation of the projected Jacobians in (2.5) and (2.6) is simply an instance of the left block of (A.5).

This modular technique for computing the Jacobian matrix allows for significantly easier implementation at little computational expense. It is also a more modular approach, since the choice of which deformations used can be made on the fly.

Bibliography

[Adi85] G. Adiv. Determining 3-d motion and structure from optical flow generated by several moving objects. *IEEE Pattern Analysis and Machine Intelligence*, 7(4):384–401, July 1985.

[ASR93] T. Akimoto, Y. Suenaga, and R.Wallace. Automatic creation of 3D facial models. *IEEE Computer Graphics and Applications*, 13(5):16–22, September 1993.

[Azu96] F. Azuola. *Error in representation of standard anthropometric data by human figure models*. PhD thesis, University of Pennsylvania, 1996.

[BA96] K. Bush and O. Antonyshyn. 3-dimensional facial anthropometry using a laser-surface scanner–validation of the technique. *Plastic and reconstructive surgery*, 98(2):226–235, August 1996.

[BAHH92] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings ECCV '92*, pages 237–252, 1992.

[BC86] T.J. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Pattern Analysis and Machine Intelligence*, 8(1):90–99, January 1986.

[Bel80] D. Belsley. *Regression diagnostics : identifying influential data and sources of collinearity*. Wiley, 1980.

[BEP96] S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. In *Proceedings ICPR '96*, page C8A.3, 1996.

[BFO95] T. Boult, S. Fenster, and T. O'Donnell. Physics in a fantasy world vs. robust statistical estimation. In *NSF/ARPA Workshop on 3D Object Representation for Computer Vision*, pages 277–296, 1995.

[BN92] T. Beier and S. Neely. Feature-based image metamorphosis. In *Proceedings SIGGRAPH '92*, volume 26, pages 35–42, July 1992.

[Boo89] F. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Pattern Analysis and Machine Intelligence*, 11(6):567–585, 1989.

[Boo91] F. Bookstein. *Morphometric Tools for Landmark Data: Geometry and Biology*. Cambridge University Press, 1991.

[BY95] M. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *Proceedings ICCV '95*, pages 374–381, 1995.

[CAHT94] C. Choi, K. Aizawa, H. Harashima, and T. Takebe. Analysis and synthesis of facial image sequences in model-based image coding. *IEEE Circuits and Systems for Video Technology*, 4(3):257–275, 1994.

[CG91] G. Celniker and D. Gossard. Deformable curve and surface finite elements for free-form shape design. In *Proceedings SIGGRAPH '91*, volume 25, pages 257–266, 1991.

[CMD94] M. Chan, D. Metaxas, and S. Dickinson. Physics-based tracking of 3D objects in 2D image sequences. In *Proceedings ICPR '94*, pages A:432–436, 1994.

[dC76] M. P. do Carmo. *Differential Geoemtry of Curves and Surfaces*. Prentice-Hall, 1976.

[DiP91] S. DiPaola. Extending the range of facial types. *Journal of Visualization and Computer Animation*, 2(4):129–131, 1991.

[DM96] D. DeCarlo and D. Metaxas. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *Proceedings CVPR '96*, pages 231–238, 1996.

[DM98] D. DeCarlo and D. Metaxas. Deformable model-based shape and motion analysis from images using motion residual error. In *Proceedings ICCV '98*, pages 113–119, 1998.

[DMS98] D. DeCarlo, D. Metaxas, and M. Stone. An anthropomtric face model using variational techniques. In *Proceedings SIGGRAPH '98*, pages 67–74, 1998.

[Doo82] M. Dooley. Anthropometric modeling programs – a survey. *IEEE Computer Graphics and Applications*, 2:17–25, November 1982.

[DPR92] S. Dickinson, A. Pentland, and A. Rosenfeld. 3-d shape recovery using distributed aspect matching. *IEEE Pattern Analysis and Machine Intelligence*, 14(2):174–198, February 1992.

[EF78] P. Ekman and W. Friesen. *The Facial Action Coding System*. Consulting Psychologist Press, Inc., 1978.

[EP97] I.A. Essa and A.P. Pentland. Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Pattern Analysis and Machine Intelligence*, 19(7):757–763, July 1997.

[Far87] L. Farkas. *Anthropometric Facial Proportions in Medicine*. Thomas Books, 1987.

[Far93] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 1993.

[Far94] L. Farkas. *Anthropometry of the Head and Face*. Raven Press, 1994.

[GC95] S. Gortler and M. Cohen. Hierarchical and variational geometric modeling with wavelets. In *1995 Symposium on Interactive 3D Graphics*, pages 35–42, April 1995.

[Gel74] A. Gelb. *Applied Optimal Estimation*. MIT Press, 1974.

[GGW$^+$98] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. Making faces. In *Proceedings SIGGRAPH '98*, pages 55–66, July 1998.

[Gib85] A. Gibbons. *Algorithmic Graph Theory*. Cambridge University Press, 1985.

[GL89] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1989.

[Gor89] C. Gordon. *1988 anthropometric survey of U.S. Army personnel: methods and summary statistics*. United States Army Natick Research, Development and Engineering Center, 1989.

[GQB89] M. Grosso, R. Quach, and N. Badler. Anthropometry for computer animated human figures. In N. Magnenat-Thalmann and D. Thalmann, editors, *State-of-the-art in Computer Animation: Proceedings of Computer Animation '89*, New York, 1989. Springer-Verlag.

[GW93] Michael Gleicher and Andrew Witkin. Supporting numerical computations in interactive contexts. In *Proceedings of Graphics Interface '93*, pages 138–146, Toronto, Ontario, Canada, May 1993. Canadian Information Processing Society.

[HKD93] M. Halstead, M. Kass, and T. DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. In *Proceedings SIGGRAPH '93*, volume 27, pages 35–44, August 1993.

[Hor86] B. Horn. *Robot Vision*. McGraw-Hill, 1986.

[Hrd72] A. Hrdlicka. *Practical anthropometry*. AMS Press, 1972.

[HW88] B. Horn and E. Weldon. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, June 1988.

[JP97] T. Jebara and A. Pentland. Parametrized structure from motion for 3D adaptive feedback tracking of faces. In *Proceedings CVPR '97*, pages 144–150, 1997.

[Koc93] R. Koch. Dynamic 3-D scene analysis through synthesis feedback control. *IEEE Pattern Analysis and Machine Intelligence*, 15(6):556–568, June 1993.

[KS96] J. Kolar and E. Salter. *Craniofacial Anthropometry: Practical Measurement of the Head and Face for Clinical, Surgical and Research Use*. Charles C. Thomas Publisher, LTD, 1996.

[Lew89] J. P. Lewis. Algorithms for solid noise synthesis. *Proceedings SIGGRAPH '89*, 23(3):263–270, 1989.

[LRF93] H. Li, P. Roivainen, and R. Forchheimer. 3-D motion estimation in model-based facial image coding. *IEEE Pattern Analysis and Machine Intelligence*, 15(6):545–555, June 1993.

[LTC97] A. Lanitis, C.J. Taylor, and T.F. Cootes. Automatic interpretation and coding of face images using flexible models. *IEEE Pattern Analysis and Machine Intelligence*, 19(7):743–756, July 1997.

[LTK95] Y. Lee, D. Terzopoulos, and K.Waters. Realistic face modeling for animation. In *Proceedings SIGGRAPH '95*, pages 55–62, 1995.

[Met96] D. Metaxas. *Physics-Based Deformable Models : Applications to Computer Vision, Graphics, and Medical Imaging*. Kluwer Academic Publishers, 1996.

[MRB95] Y. Moses, D. Reynard, and A. Blake. Robust real time tracking and classificiation of facial expressions. In *Proceedings ICCV '95*, pages 296–301, 1995.

[MS92] H. Moreton and C. Séquin. Functional optimization for fair surface design. In *Proceedings SIGGRAPH '92*, volume 26, pages 167–176, 1992.

[MT93] D. Metaxas and D. Terzopoulos. Shape and nonrigid motion estimation through physics-based synthesis. *IEEE Pattern Analysis and Machine Intelligence*, 15(6):580–591, June 1993.

[MTMdAT89] N. Magnenat-Thalmann, H. Minh, M. de Angelis, and D. Thalmann. Design, transformation and animation of human faces. *The Visual Computer*, 5(1/2):32–39, March 1989.

[Nag83] H.H. Nagel. Displacement vectors derived from second-order intensity variations in image sequences. *CVGIP*, 21(1):85–117, January 1983.

[NH87] S. Negahdaripour and B. Horn. Direct passive navigation. *IEEE Pattern Analysis and Machine Intelligence,* 9(1):168–176, January 1987.

[NLL90] H. Nowacki, D. Liu, and X. Lu. Fairing Bézier curves with constraints. *Computer Aided Geometric Design,* 7(1-4):43–56, 1990.

[NS85] A. Netravali and J. Salz. Algorithms for estimation of three-dimensional motion. *AT&T Technical Journal*, 64:335–346, 1985.

[NY93] S. Negahdaripour and C.H. Yu. A generalized brightness change model for computing optical flow. In *ICCV93*, pages 2–11, 1993.

[Par82] F. Parke. Parameterized models for facial animation. *IEEE Computer Graphics and Applications*, 2(9):61–68, 1982.

[PB88] J. Platt and A. Barr. Constraint methods for flexible models. In *Proceedings SIGGRAPH '88*, volume 22, pages 279–288, 1988.

[PS91] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

[PTVF92] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[PW91] M. Patel and P. Willis. FACES: The facial animation construction and editing system. In *Eurographics '91*, 1991.

[PW96] F. Parke and K. Waters. *Computer Facial Animation*. A K Peters, 1996.

[Rog84] S. Rogers. *Personal Identification from Human Remains*. Charles C. Thomas Publisher, LTD, 1984.

[RWBM96] D. Reynard, A. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *Proceedings ECCV '96*, pages I:357–368, 1996.

[SAH91] E. Simoncelli, E. Adelson, and D. Heeger. Probability distributions of optical flow. In *Proceedings CVPR '91*, pages 310–315, 1991.

[Sha89] A. Shabana. *Dynamics of Multibody Systems*. Wiley, 1989.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings CVPR '94*, pages 593–600, 1994.

[Str88] G. Strang. *Linear algebra and its applications*. Harcourt, Brace, Jovanovich, 1988.

[TQ94] D. Terzopoulos and H. Qin. Dynamic nurbs with geometric constrains for interactive sculpting. *ACM Transactions on Graphics*, 13(2):103–136, 1994.

[TW93] D. Terzopoulos and K. Waters. Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Pattern Analysis and Machine Intelligence*, 15(6):569–579, 1993.

[TWK87] D. Terzopoulos, A. Witkin, and M. Kass. Symmetry-seeking models and 3D object reconstruction. *International Journal of Computer Vision*, 1(3):211–221, 1987.

[TWK88] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988.

[VP97] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. *IEEE Pattern Analysis and Machine Intelligence*, 19(7):733–742, 1997.

[WW92] W. Welch and A. Witkin. Variational surface modeling. In *Proceedings SIGGRAPH '92*, volume 26, pages 157–166, 1992.

[WW94] W. Welch and A. Witkin. Free–Form shape design using triangulated surfaces. In *Proceedings SIGGRAPH '94*, volume 28, pages 247–256, July 1994.

[YCH92] A.L. Yuille, D.S. Cohen, and P. Halliman. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8:104–109, 1992.

[YD94] Y. Yacoob and L.S. Davis. Computing spatio-temporal representations of human faces. In *Proceedings CVPR '94*, pages 70–75, 1994.

[Zie77] O. Zienkiewicz. *The Finite Element Method*. McGraw-Hill, 1977.