RuCCS TR-51
August, 1999

Adjusting Model Parameters using Model-Based Optical Flow Residuals

Doug DeCarlo
decarlo@cs.rutgers.edu
Rutger's University
Department of Computer Science

Center for Cognitive Science

Dimitris Metaxas
dnm@central.cis.upenn.edu
University of Pennsylvania
Department of Computer and Information Science

Technical Report TR-51
Center for Cognitive Science
Psych Bldg Addition, Busch Campus
Rutgers University - New Brunswick
152 Frelinghuysen Road
Piscataway, NJ 08854-8020

Adjusting Model Parameters using Model-Based Optical Flow Residuals

Douglas DeCarlo
Department of Computer Science
and Center for Cognitive Science
Rutgers University
decarlo@cs.rutgers.edu

Dimitris Metaxas
Department of Computer and Information Science
University of Pennsylvania
dnm@central.cis.upenn.edu

Abstract

We present a method for estimating the shape and motion of a deformable model using the least-squares residuals from a model-based optical flow computation. This method is built on top of an estimation framework using optical flow and image features, where optical flow affects only the motion parameters of the model. Using the results of this computation, our new method adjusts all of the parameters (both shape and motion) so that the residuals from the flow computation are minimized. We present experiments that demonstrate that this method is a considerable improvement over a framework using only optical flow and features, especially in the estimation of the shape.

1 Introduction

Analyzing the error in model-based tracking frameworks can lead to further improvements of the parameter estimates. Typical model-based tracking methods choose a least-squares solution to set the value of those parameters that the measurements directly determine. In over-determined situations, this approach can leave behind a significant residual. Inevitably, this residual reflects deviations that can be caused by idealizations inherent in the model as well as noisy measurements.

But importantly, while parameters affected by the measurements are set as well as possible, parameters that remain unaffected by the measurements may be inaccurate, and so also contribute to the residual. Within this modeling framework, this is a problem we can actually correct.

Consider a model-based optical flow computation, for example. The model parameters are split in those which model motion, and those which model the underlying and unchanging shape. Observed motion directly determines only the motion parameters; the true values of the shape parameters do not change over time. However, inaccurate shape estimates make it impossible for the model to explain the observed motion using the motion parameters—this results in an increased flow residual. In this case, we can adjust the shape parameters to improve the entire estimate.

The challenge is to be faithful to the distinction between shape and motion parameters as the adjustment is computed. For example, it's insufficient to simply stage the computation, and use the leftovers from the motion parameter computation to feed a computation which determines how the shape parameters could have changed over time to explain the remaining observed motion [14]. This method, while effective in image-coding, simply treats shape parameters as second-class motion parameters. We propose computing an adjustment to the shape parameters that minimizes the residual of the motion estimate. In other words, we determine a new configuration for which the motion parameters would have produced a smaller residual in the first place.

1.1 Related work

Residuals from one computation are often used as input to another that follows it. For instance, coarse-to-fine methods typically compute a solution at a particular level of detail, and continue on to finer levels after subtracting away its current guess. The method presented here proceeds somewhat differently. In this case, we start with a model-based optical flow computation which provides estimates of motion parameters. The residuals from this computation are then used to adjust *both* the shape and motion parameters of the model.

Our results should also be distinguished from the large body of work on structure from motion which estimates shape and motion using an optical flow field, reviewed in [2]. There has also been a great deal of work on the structure from motion problem using feature correspondences, which is surveyed in [13]. Another approach aligns locations on a model with image features using displacements or gradient fields by solving a template alignment optimization problem [16, 17, 25, 26].

This paper describes an alternative to these structure from motion methods. Our method is coupled to a model-based optical flow computation using a deformable model. Instead of performing direct surface reconstruction from optical flow, our method indirectly adapts model parameters so that the residual of the model-based optical flow is reduced. Section 4 contains a discussion of precisely how these methods differ from our technique.

1.2 Shape and motion separation

The starting point for our method is an intuitive distinction between shape and motion; the process of model design must encode information about a class of objects by allowing for the categorization of model parameters as describing either variation in shape or motion. The shape parameters are a *static* quantity for a particular observed object, and describe its unchanging geometric features. The motion parameters are a *dynamic* quantity, which change when the observed object moves or deforms. Of course, there is no guarantee that the shape and motion of some class of objects is separable; this is a simplifying assumption that we make, and will only apply to a certain degree of accuracy. For example, with human faces, shape parameters describe an individual's appearance, while motion parameters encode the location of their head, as well as their facial displays and expressions. This division is often built into face models [4, 15, 18, 24] to simplify model construction or estimation, and has been used to facilitate learning the variability of motions for a class of objects [22].

The ultimate goal of this separation is to produce a simpler estimation problem. During esti-

mation, the change in the shape parameters should tend to zero as the shape of the observed object is established. Once this occurs, fitting need only continue for the motion parameters. Therefore, during model design, the separation into shape and motion should encode as many of the model deformations with shape parameters as possible. This decision also leads to a more efficient tracking system. This distinction leads us to develop a method where changes in the image are initially attributed entirely to motion, but then the residual from the reconstructed motion is used to correct errors in the shape and motion parameters.

For models with separate parameters for shape and motion, certain cues such as optical flow are appropriately used only for the estimation of motion parameters (and not shape parameters). While in some cases updating all the parameters (shape and motion) based on the flow can result in smaller deviations [14], this is missing the point of separating shape and motion in the first place, and is in conflict with the view of the shape parameters as having static values.

1.3 Using residuals

A significant error in the current model estimate will interfere with the optical flow estimates of the motion, since the model and image will be misaligned. For example, if the estimate of a "nose protrusion" parameter is inaccurate, the pattern of motion that the model would predict during a head turn would be incorrect in the local region of the nose. However, this interference is quite systematic, which enables the adjustment of the current shape and motion estimate. This adjustment aims to correct the error which interfered with the flow computation. Continuing with the above example, we would aim to adjust the nose protrusion parameter in a way that improves the motion model's accuracy. We will perform this adjustment using the residual from the optical flow constraint equation.

From this residual, each pixel used in the optical flow computation supplies one piece of information which is then used to determine how the parameters can be corrected to minimize the in-

terference. However, some pixels will not supply any useful information. And even worse, many of the pixels will include distracting information resulting from optical flow linearization, optical flow constraint violations (such as lighting changes, shadows, or specularities), motion estimation errors, and noise. As a result, we must be sure to use a sufficient number of pixels, as well as to avoid adjusting the parameters based on distracting information. This second point is addressed by the following empirically determined result: residual contributions which result from small errors in the estimated model parameters are significantly larger than those caused by distracting sources (such as optical flow constraint violations and linearization). We demonstrate this empirical result in the context of face tracking, and it is likely to apply in other domains where model-based optical flow based tracking is successful.

Our method is built on top of the model-based face tracking framework described in [7]. This framework used a model-based optical flow computation as a hard constraint on the motion of a deformable model. Aside from flow, the motion of the deformable model was determined by aligning the model with image features (edges). Using features prevented the accumulation of tracking error, which would have otherwise been a problem using flow alone. In Section 5, we will show how using residuals improves the shape and motion estimates of a face. When used with [7], most of the improvement is in the shape parameters, as the motion parameters are already quite accurate based on the estimates using the optical flow and edges.

1.4 Outline

In our method, 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. After a brief review of deformable models in Section 2, we discuss existing approaches to model-based optical flow in Section 3. Section 4 describes our method for adjusting the model parameters using residuals from a model-based optical flow computation. Finally, we

present experiments in Section 5 demonstrating our method, along with some discussion.

2 Deformable models

Deformable models [17, 21, 25] are parameterized shapes that deform due to forces according to physical laws. For vision applications, physics provides a useful analogy for treating shape estimation [17], 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 shape of the deformable model $\mathbf{x}$ is parameterized by a vector of values $\mathbf{q}$ 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 goal of shape and motion estimation is to recover the value of $\mathbf{q}$ over time from a sequence of images. For this paper, we will be using the three-dimensional parameterized face model from [7].

As stated earlier, to distinguish the processes of shape estimation and motion tracking, the parameters in $\mathbf{q}$ are rearranged and separated into $\mathbf{q}_\mathrm{b}$ (the basic shape of the object) and $\mathbf{q}_\mathrm{m}$ (rigid and non-rigid motion), so that $\mathbf{q} = (\mathbf{q}_\mathrm{b}^\top, \mathbf{q}_\mathrm{m}^\top)^\top$. With our face model, $\mathbf{q}_\mathrm{b}$ describes an individual's appearance, while $\mathbf{q}_\mathrm{m}$ encodes the location of their head, as well as their facial displays and expressions. Figure 1 shows examples of the face model undergoing various shape deformations (showing four different individuals), motion deformations (showing brow raising and frowning, smiling, and mouth opening) and finally two examples of when several deformations are applied at once.

The model $\mathbf{x}$ is formed by applying deformation functions to the underlying shape $\mathbf{s}$. For this paper, the underlying face model $\mathbf{s}$ is a polygon mesh (shown in the center of Figure 1). There are separate deformation functions for shape ($\mathbf{T}_b$) and for motion ($\mathbf{T}_m$). The shape deformation is

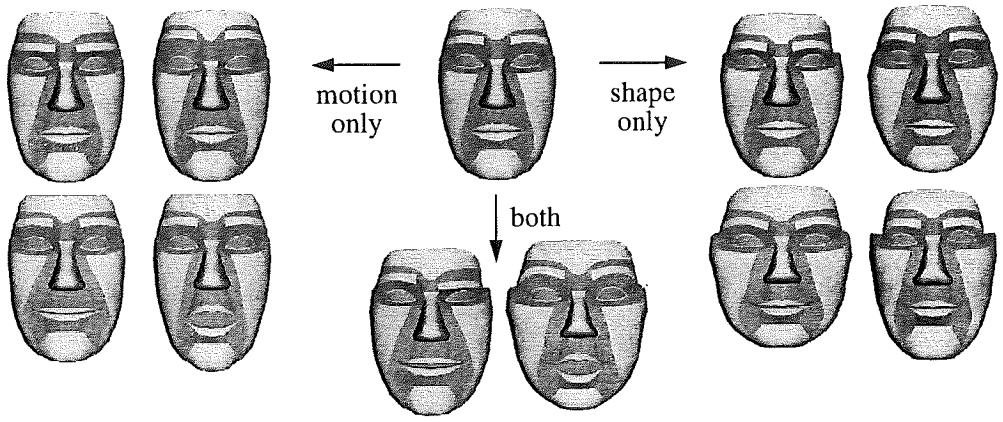

Figure 1: Example parameterized deformations of the face model (with separate parameters for shape and motion)

applied first, so that:

$$\mathbf{x}(\mathbf{q};\mathbf{u}) = \mathbf{T}_m(\mathbf{q}_\mathrm{m};\, \mathbf{T}_b(\mathbf{q}_\mathrm{b};\, \mathbf{s}(\mathbf{u}))) \tag{1}$$

The shape deformation $\mathbf{T}_b$ uses the parameters $\mathbf{q}_\mathrm{b}$ to deform the underlying shape $\mathbf{s}$. On top of this is the motion deformation $\mathbf{T}_m$ with parameters $\mathbf{q}_\mathrm{m}$, which includes a rigid translation and rotation (head motion), as well as non-rigid deformations (facial expressions and displays).

When modeling an object viewed in images, $\mathbf{x}$ must include a camera projection, resulting in a two-dimensional model called $\mathbf{x}_p$, which is projected flat from the original three-dimensional model.

2.1 Kinematics and Dynamics

The kinematics of the model are 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:

$$\dot{\mathbf{x}}(\mathbf{u}) = \mathbf{L}(\mathbf{q};\mathbf{u})\dot{\mathbf{q}} \tag{2}$$

where $\mathbf{L} = \partial\mathbf{x}/\partial\mathbf{q}$ is the model Jacobian [17]. For reasons of conciseness, the dependency of $\mathbf{L}$ on $\mathbf{q}$ is often omitted.

We view $\mathbf{L}$ as consisting of components that correspond to $\mathbf{q}_\mathrm{b}$ and $\mathbf{q}_\mathrm{m}$, so that it can be written as $[\mathbf{L}_\mathrm{b}\ \mathbf{L}_\mathrm{m}]$. The Jacobian of the projected model $\mathbf{x}_p$ is written as $\mathbf{L}_p$, and is decomposed into components for $\mathbf{q}_\mathrm{b}$ and $\mathbf{q}_\mathrm{m}$ as $\left[\mathbf{L}_{\mathrm{b}p}\ \mathbf{L}_{\mathrm{m}p}\right]$. In Section 3, we use this distinction to attribute observed motion as resulting solely from changes in the motion parameters.

The models defined above are useful for applications such as shape and motion estimation when used in a physics-based framework [17]. 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. These equations are simplified in a standard manner [17], and in this case result in:

$$\dot{\mathbf{q}} = \mathbf{f}_\mathbf{q} \tag{3}$$

where the applied forces $\mathbf{f}_\mathbf{q}$ are computed from two-dimensional image forces $\mathbf{f}_\mathrm{image}$ as:

$$\mathbf{f}_\mathbf{q} = \sum_j \mathbf{L}_p(\mathbf{u}_j)^\top \mathbf{f}_\mathrm{image}(\mathbf{u}_j) \tag{4}$$

The distribution of forces on the model is based in part on forces computed from the edges of an input image [17]. With that, and given an adequate model initialization, these forces will align features on the model with image features, thereby determining appropriate parameter values. The dynamic system in (3) is solved by integrating over time, using standard (explicit) differential equation integration techniques, such as Euler integration:

$$\mathbf{q}(t+\Delta t) = \mathbf{q}(t) + \dot{\mathbf{q}}(t)\Delta t \tag{5}$$

The initialization which specifies the value of $\mathbf{q}(0)$ is described in Section 5.

3 Model-based optical flow

The optical flow is defined as the apparent motion of brightness patterns across an image [11]. Attempting to use this information in applications such as object tracking requires assumptions about the objects being viewed. Most common is the assumption that particular locations on viewed objects do not change in brightness. This brightness constancy assumption leads to the formulation of the well-known optical flow constraint equation for the image $\mathbf{I}$ at a particular pixel (the assumption manifests itself as the zero on the right-hand-side):

$$\nabla \mathrm{I} \begin{bmatrix} u \\ v \end{bmatrix} + \mathrm{I_t} = 0 \tag{6}$$

where $\nabla \mathrm{I} = \begin{bmatrix} \mathrm{I_x} & \mathrm{I_y} \end{bmatrix}$ are the spatial derivatives and $\mathrm{I_t}$ is the temporal derivative of the image intensity. u and v are the components of the image velocities.

The model-based optical flow constraint equation is a reformulation of (6) in terms of a model's motion parameters $\mathbf{q}_\mathrm{m}$. When viewing a model under projection, there exists a unique model point $\mathbf{u} \in \Omega$ which corresponds to a particular pixel (except on occluding boundaries and situations involving transparency). In a model-based approach, the image velocities u and v are specified by projected velocities of points on the model $\dot{\mathbf{x}}_p(\mathbf{u})$:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \dot{\mathbf{x}}_p(\mathbf{u}) = \mathbf{L}_{\mathrm{m}p}(\mathbf{u})\dot{\mathbf{q}}_\mathrm{m} \tag{7}$$

Note that only the changes resulting from the motion parameters $\mathbf{q}_\mathrm{m}$ are included, as optical flow velocities do not reflect changes in the shape parameters $\mathbf{q}_\mathrm{b}$, so that only $\mathbf{L}_{\mathrm{m}p}$ is used here (and not

all of $\mathbf{L}_p$). The model-based optical flow constraint equation rewrites (6) using (7):

$$\nabla \mathrm{I}\, \mathbf{L}_{\mathrm{m}p}(\mathbf{u})\dot{\mathbf{q}}_{\mathrm{m}} + \mathrm{I}_{\mathrm{t}} = 0 \tag{8}$$

When considered over a set of n pixels, a stacked set of instances of (8) can be written in matrix form (where each row corresponds to an individual pixel):

$$\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}} = 0 \tag{9}$$

Formulations similar to (9) (although superficially appearing quite different) can be found in [1, 3, 6, 12, 14, 15, 19, 20].

The value of $\dot{\mathbf{q}}_{\mathrm{m}}$ in (9) can be determined by least-squares minimization:

$$\min_{\dot{\mathbf{q}}_{\mathrm{m}}} [\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}}]^2 \tag{10}$$

where $[\mathbf{v}]^2 = \mathbf{v}^\top \mathbf{v}$. An iterative approach to solving this is taken in [3, 12, 14, 19, 20]. Alternatively, it can be solved in a single step using the pseudo-inverse (where $\mathbf{B}^+$ is the pseudo-inverse of $\mathbf{B}$ [23]) [6, 20, 15]:

$$\dot{\mathbf{q}}_{\mathrm{m}} = -\mathbf{B}^+ \mathbf{I}_{\mathrm{t}} \tag{11}$$

This is the linear least-squares solution; it linearizes by assuming $\mathbf{L}_{\mathrm{m}p}$ is constant (ignoring its dependency on $\mathbf{q}$). This is the solution method we employ here.

The derivation of (6) involves the truncation of a Taylor series, and as a result requires relatively small motions between frames. To address problems with estimating larger motions (or to implement coarse-to-fine methods), some iterative approaches transform the model geometry at each iter-

ation using the previous motion estimate [14], while others undo the previous estimate by warping the input images [3].

The most serious difficulty for these techniques is combating tracking drift. Using only velocity information, small estimation errors accumulate over time. The solution to this problem is to include other information (such as features or edges) to prevent errors from building up [7, 14, 15].

3.1 Optical Flow Residuals

Unlike image-based optical flow techniques, these model-based methods do not require assumptions about the smoothness of the flow field to determine a solution, as the number of pixels providing useful information is sufficiently greater than the number of motion parameters. Of course, now that the solution is over-determined, there will be a residual from the least-squares solution, given the solution $\dot{\mathbf{q}}_{\mathrm{m}}$:

$$R = \mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + \mathbf{I}_{\mathrm{t}} \tag{12}$$

The residual R is a vector having dimension n (the number of pixels used in the flow computation).

The residual comes from many sources. Aside from measurement noise, most obviously are linearization errors, resulting from ignoring the higher order terms in (6) (which were truncated in the Taylor series). With the linear least squares solution, the linearization of $\mathbf{L}_{\mathrm{m}p}$ is also a culprit. Others include violations of the brightness constancy assumption, such as lighting changes, shadows, and specularities. Finally, shape and motion estimation errors (deviation between $\mathbf{q}$ and its actual value) will prevent the model from properly aligning with the image, and will cause a sizable increase in the residual.

In fact, we claim that if significant errors are present in the estimated shape and motion, they will be the primary contributors to the residual. We support this claim empirically in Section 5. This

means that the residual is a valuable piece of information that can be used for estimating shape and motion. The next section describes how to compute small adjustments to the shape and motion parameters which reduce the residual.

4 Adjusting parameters using residuals

To provide insight into this method, it's worth comparing the use of a three-dimensional model under projection in a model-based optical flow framework [7, 15], with a two-dimensional model of image motion [4]. For a particular value of $\mathbf{q}$, the 3D model, when projected as a 2D model in the image, is actually quite comparable with a 2D image motion model. The key difference is that the form of the 3D model when projected into the image changes with $\mathbf{q}$—so that changes in parameters of the 3D model directly affects its projected motion in the image. So we can now ask: how could this 2D motion parameterization have been different (by changing $\mathbf{q}$), which would have resulted in a smaller residual? The remainder of this section describes how we answer this question by adjusting the model's parameters using the residuals.

There are various approaches to using this residual information. One possible approach explains the residual as directly resulting from shape deviation—this is basically a structure from motion approach. In other words, the leftover motion not accounted for by motion parameters is used to update the shape using the same formulation as for determining motion (from Section 3), as in:

$$\mathbf{B}\dot{\mathbf{q}}_m + \mathbf{B}_b\dot{\mathbf{q}}_b + \mathbf{I}_t = \mathbf{0} \tag{13}$$

where the construction of $\mathbf{B}_b$ is analogous to $\mathbf{B}$, but uses $\mathbf{L}_b$ instead of $\mathbf{L}_m$. Instead of solving one large system, the system in (13) is decoupled, and is solved for motion first, and then for shape in

terms of the residual R:

$$\mathbf{B}_b\dot{\mathbf{q}}_b + R = \mathbf{0} \quad \Rightarrow \quad \dot{\mathbf{q}}_b = -\mathbf{B}_b^+ R \tag{14}$$

This method is closely related to one described by Koch [14]. In that work, shape parameters are actually updated in two steps; first using discrepancies parallel to the line of sight, then those that are perpendicular to the line of sight. This problem involves solving the following minimization:

$$\min_{\dot{\mathbf{q}}_b} [\mathbf{B}\dot{\mathbf{q}}_m + \mathbf{B}_b\dot{\mathbf{q}}_b + \mathbf{I}_t]^2 \qquad (\text{given } \dot{\mathbf{q}}_m) \tag{15}$$

This is basically a staged version of the problem in (10), where $\dot{\mathbf{q}}_m$ is determined first, followed by $\dot{\mathbf{q}}_b$.

This is a reasonable approach in the context of image coding, where image fidelity is of much greater importance than the accuracy of the face shape estimate—the face shape is deformed to account for the tracking errors in motion. This produces a face shape that results in a better image, but does not necessarily estimate the actual face shape of the subject. Plus, given our distinction between shape and motion parameters, it does not make sense to adjust the shape parameters $\mathbf{q}_b$ directly from observed velocities, as in [14], since the true value of $\mathbf{q}_b$ is a static quantity.

Instead of this, our approach determines the small change in $\mathbf{q}$ that effects the largest reduction in R. Let $\Delta\mathbf{q}$ be the current deviation of $\mathbf{q}$ from its true value (not including the motion extracted in $\dot{\mathbf{q}}_m$)—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 $\mathbf{L}_m$ using its Taylor-series expansion is sufficiently accurate:

$$\mathbf{L}_{\mathrm{m}p}(\mathbf{u};\mathbf{q}+\Delta\mathbf{q}) \approx \mathbf{L}_{\mathrm{m}p}(\mathbf{u};\mathbf{q}) + \frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u};\mathbf{q})}{\partial \mathbf{q}}\Delta\mathbf{q} \tag{16}$$

Combining this approximation of $\mathbf{L}_{\mathrm{m}p}$ with the model-based optical flow constraint equation (8) results in:

$$\nabla \mathrm{I}\, \mathbf{L}_{\mathrm{m}p}(\mathbf{u})\dot{\mathbf{q}}_{\mathrm{m}} + \nabla \mathrm{I}\left(\frac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u})}{\partial \mathbf{q}}\Delta\mathbf{q}\right)\dot{\mathbf{q}}_{\mathrm{m}} + \mathrm{I}_{\mathrm{t}} = 0 \tag{17}$$

where $\partial \mathbf{L}_{\mathrm{m}p}/\partial \mathbf{q}$ is part of the model Hessian matrix (a rank 3 tensor). It is written here "curried" with $\Delta\mathbf{q}$ so that the parenthesized sub-expression here is a matrix.

When (17) is considered over n pixels from the input image, this results in the system:

$$\mathbf{B}\dot{\mathbf{q}}_{\mathrm{m}} + (\mathbf{G}\dot{\mathbf{q}}_{\mathrm{m}})\,\Delta\mathbf{q} + \mathbf{I}_{\mathrm{t}} = \mathbf{0} \tag{18}$$

$$\text{where } \mathbf{G} = \begin{bmatrix} \left(\nabla \mathrm{I}_1 \dfrac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_1)}{\partial \mathbf{q}}\right)^{\mathsf{T}} \\ \vdots \\ \left(\nabla \mathrm{I}_n \dfrac{\partial \mathbf{L}_{\mathrm{m}p}(\mathbf{u}_n)}{\partial \mathbf{q}}\right)^{\mathsf{T}} \end{bmatrix} \tag{19}$$

The subscripts $[1...n]$ in the construction of $\mathbf{G}$ correspond to a particular row in (18). The transpositions performed in the construction of $\mathbf{G}$ allow it now to be curried with $\dot{\mathbf{q}}_{\mathrm{m}}$ (this construction transposes the second and third indices of the tensor $\mathbf{G}$). We can now rewrite (18) using the residual

(12) and the value of $\dot{\mathbf{q}}_m$ supplied by the model-based optical flow solution in (11):

$$-\left(\mathbf{G}\mathbf{B}^{+}\mathbf{I}_t\right)\Delta\mathbf{q}+R=\mathbf{0} \tag{20}$$

Then, $\Delta\mathbf{q}$ is determined using the pseudo-inverse:

$$\Delta\mathbf{q}=(\mathbf{G}\mathbf{B}^{+}\mathbf{I}_t)^{+}R \tag{21}$$

This least squares solution determines the best set of small changes in $\mathbf{q}_b$ and $\mathbf{q}_m$ that minimize the optical flow residual (12), given the linearization of $\mathbf{L}_{mp}$ in (16). This effectively solves the following minimization:

$$\min_{\Delta\mathbf{q}}\left[\mathbf{B}(\mathbf{q}+\Delta\mathbf{q})\,\dot{\mathbf{q}}_m+\mathbf{I}_t\right]^2 \qquad (\text{given } \dot{\mathbf{q}}_m) \tag{22}$$

where $\mathbf{B}(\mathbf{q}+\Delta\mathbf{q})$ is approximated as $\mathbf{B}(\mathbf{q})+\partial\mathbf{B}/\partial\mathbf{q}\cdot\Delta\mathbf{q}$. The intuition for this analysis—that we're solving a minimization process that is faithful to the distinction between shape and motion parameters—is realized here in the formulation of a minimization problem very different from (15). Note that it is possible that the new value of R would be smaller if (15) is used over (22), but this goes against the assumption that $\mathbf{q}_b$ are static parameters, and would result in an inappropriate estimate.

4.1 Solution improvement

The value of $\Delta\mathbf{q}$ from the previous section specifies an absolute update to the state (unrelated to the current time step Δt)—we could simply add $\Delta\mathbf{q}$ after each iteration.

$$\mathbf{q}(t+\Delta t)=\mathbf{q}(t)+\dot{\mathbf{q}}\Delta t+\Delta\mathbf{q} \tag{23}$$

However, if information is available about the uncertainty in the estimate of $\dot{\mathbf{q}}$, we can add in $\Delta\mathbf{q}$ in a more principled manner.

In [9], a model-based optical flow solution is combined with edge information using an extended Kalman filter, which results in a a filtered estimate (using both the flow and edge information) of $\dot{\mathbf{q}}$ and a covariance estimate $\Lambda_{\dot{\mathbf{q}}}$.

If we assume the distracting sources in R (aside from $\Delta\mathbf{q}$) can be modeled as (zero mean) Gaussian disturbances, then from (20), the covariance of $\Delta\mathbf{q}$ is:

$$\Lambda_{\Delta\mathbf{q}} = \left((\mathbf{G}\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}})^{\top}\Lambda_{R}^{-1}(\mathbf{G}\mathbf{B}^{+}\mathbf{I}_{\mathrm{t}})\right)^{-1} \tag{24}$$

where Λ_R is the covariance of R (which we choose to be a diagonal matrix with diagonal entry σ_R). The value of σ_R represents the contributions to R from sources other than shape and motion estimation errors, and is determined in Section 5.3 from experiments where $\mathbf{q}_{\mathrm{b}}$ (the shape) is known in advance. Using this covariance information ($\Lambda_{\dot{\mathbf{q}}}$ and $\Lambda_{\Delta\mathbf{q}}$), the new value of $\mathbf{q}$ is found using cue integration techniques [5, 10], which weight $\dot{\mathbf{q}}\Delta t$ and $\Delta\mathbf{q}$ together based on their uncertainty.

The method in (23), which does not use the covariance information, is not particularly robust. At first [8], this was attributed to there being a poor linear approximation of $\mathbf{L}_{\mathrm{m}p}$, which was tested for by determining if the residual actually does decrease with the addition of $\Delta\mathbf{q}$. It seems this was actually due to the distracting sources having a significant effect on the value of $\Delta\mathbf{q}$.

4.2 Implementation

Solving (21) is made more efficient by omitting parameters in the construction of $\mathbf{G}$ which cannot be affected based on $\dot{\mathbf{q}}_{\mathrm{m}}$. 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 $\mathbf{G}$. Another example is when the extent of a parameter is simply not visible in the image. 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 (11) and (21) repeatedly to obtain a greater improvement. For the applications here, the linear approximation in (16) is relatively accurate with this face model, due to the fact that most of the model parameterization is linear scaling. As a result, only the single iteration is performed.

5 Experiments

This section describes a number of face tracking experiments on two image sequences. We demonstrate how the adjustment method using residuals is a significant improvement over the framework in [9]. We also justify our assumption that parameter estimation errors are the leading contributor to the residuals. For the remainder of this section, we will be comparing two frameworks. First, is the original framework from [9], which uses optical flow and edges (and is basically an Extended Kalman filter version of [7]). Second is the framework presented here, which is used on top of the framework of [9]. It determines $\Delta\mathbf{q}$ using (21) which is statistically combined with the original solution from [9] using (24).

5.1 Setup

The original image sequences are 8 bit gray images at NTSC resolution (480 vertical lines). In the sequences, the width of the face in the image averages 200 pixels. A single subject is used in both experiments presented here. The shape (determined by $\mathbf{q}_b$) is validated using a Cyberware range scan of the subject, shown in Figure 2.

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) fac-

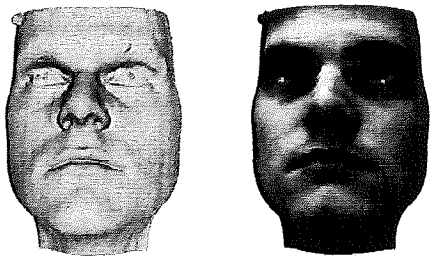

Figure 2: Range scan of the subject (shaded and textured)

ing forward. Experience has shown that the initialization process is robust to small displacements (i.e. several pixels) in the selected landmark points. Details of this initialization process are provided in [7].

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 below. The initialization process usually takes about 2 minutes of computation. Afterwards, processing each frame using the method in [9] takes approximately 1.4 seconds each. The error residual computation adds an additional 8 seconds per frame (all computation times are measured on a 175 MHz R10000 SGI O2). For both methods, 120 pixels are used in the optical flow computation (the value of n).

5.2 Estimation experiments

The shape estimation validation experiment in Figure 3 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). At each frame, Figure 4 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 (ground truth for motion is not available). The RMS error is computed 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,