Fitting Sophisticated Facial Animation Models to Image Data

P. Fua and C. Miccio
Computer Graphics Lab (LIG)
EPFL
CH-1015 Lausanne
Switzerland

Abstract

We show that we can effectively fit arbitrarily complex animation models to noisy image data. Our approach is based on the use a set of progressively finer control triangulations and takes advantage of two complementary sources of information: stereo data and silhouette edges.

1 Introduction

In this paper, we show that we can automatically fit a complex facial animation model to image data obtained using regular video cameras as opposed to sophisticated sensors such as laser range finders.

In recent years much work has been devoted to the modelling of faces from image and range data. Automated approaches can be roughly classified into the following two categories:

- Some concentrate on tracking the head motion and some features. They typically use a fairly coarse face model that is too simple for realistic face animation (e.g. [2]).
- Others use sophisticated face models with large numbers of degrees of freedom that are suitable for animation purposes but require very clean data–such as a Cyberware^{tm} scanner—to instantiate them (e.g. [6]).

Our approach aims at bridging the gap between these two kinds of approaches by fitting a detailed face model that has successfully been used to animate virtual actors to actual image data.

We typically start with a set of stereo image pairs or a video sequence. In this work, we assume that the monochrome images we use are registered and that precise camera models are available. This assumption is reasonable because there are well established photogrammetric techniques, such as bundle-adjustment, that allow the computation of these models as needed. We then go through the following three steps:

- We compute disparity maps for each stereo pair or each consecutive pair in the video sequences, fit local surface patches to the corresponding 3–D points, and use these patches to compute a central 3–D point and a normal vector.
- We attach a coarse control mesh to the animation model and perform a least squares adjustment of this control mesh so that the model matches the previously computed stereo data. We weigh the data points according to the closeness of their normals to that of the model and use an iterative reweighting technique to eliminate the outliers. We then subdivide the control mesh and repeat the procedure to refine the result.
- We use the original images to compute an optimal facet albedo for each facet of the model to achieve the closest possible resemblance to those images.

In this way, we can generate with very limited manual intervention a physical model of the surface—that is, one that includes both geometry and reflectance properties—that can then be used to animate the face. Because we use robust fitting techniques and take advantage of our rough knowledge of a face's shape, we obtain reliable results even from noisy data acquired with a cheap and entirely passive technique.

2 Least Squares Framework

In this work, we use the facial animation model that has been developed at MIRALab and LIG [5]. It can produce the different facial expressions arising from speech and emotions. Its multilevel configuration reduces complexity and provides independent control for each level. At the lowest level, a deformation controller simulates muscle actions using rational free form deformations. At a higher level, the controller produces animations given corresponding to abstract entities such as sentences and emotions.

The corresponding skin surface is shown in its rest position in Figure 1(a). We will refer to it as the *surface triangulation*. From a fitting point of view, this model embodies a rough knowledge about the face's shape and can be used to constrain the search space. Our goal is to deform the surface—without changing its topology—so that if conforms to the image data. In standard least-squares fashion, we will use this data to write *nobs* observation equations of the form

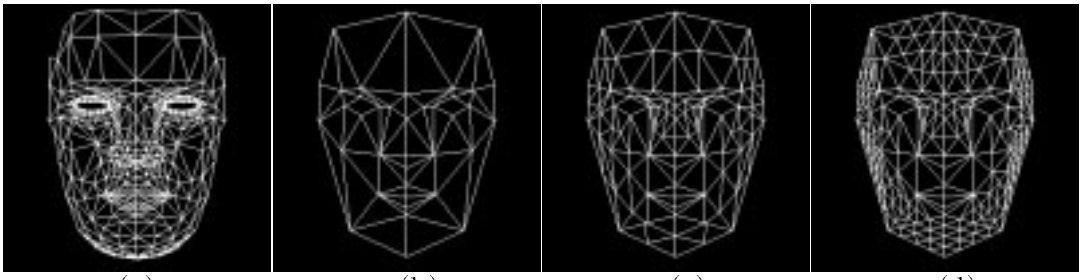
$$f_i(P) = obs_i + \epsilon_i \quad , 1 \leq i \leq nobs \quad , \tag{1}$$

where P is a parameter vector that defines the shape of the surface. We will then minimize

$$\sum_{1 \leq i \leq nobs} w_i \epsilon_i^2 \quad , \tag{2}$$

where w_i is a weight associated to each observation.

In theory we could take the parameter vector P to be the vector of all x, y, and z coordinates of the surface triangulation. However, because the image data is very noisy, we would have to impose a very strong regularization constraint. For example, we have tried to treat the surface triangulation as finite element mesh. Due to its great irregularity and



(a) (b) (c) (d)

Figure 1: Face models. (a) The model used to animate faces. (b,c,d) Increasingly refined control triangulations.

its large number of vertices, we have found the fitting process to be very brittle and the smoothing coefficients difficult to adjust.

For increased robustness, we have therefore implemented the following scheme. Instead of directly modifying the vertex positions during the minimization, we introduce *control triangulations* such as the ones shown in Figure 1(b,c,d). The vertices of the surface triangulation are "attached" to the control triangulation and the range of allowable deformations of the surface triangulation is defined in terms of weighted averages of displacements of the vertices of the control triangulation.

More specifically, we project each vertex of the surface triangulation onto the control triangulation. If this projection falls in the middle of a control facet, we "attach" the vertex to the three vertices of the control facets and compute the corresponding barycentric coordinates. If this projection falls between two facets, we "attach" the vertex to the vertices of the corresponding edge. In effect, we take one of the barycentric coordinates to be zero.

Given these attachments, we define the shape of the surface triangulation in terms of displacements of the vertices of the control triangulation. The 3–D position P_i of vertex i of the surface triangulation is taken to be

$$P_i = P_i^0 + l_1^i \delta_{j1}^i + l_2^i \delta_{j2}^i + l_3^i \delta_{j3}^i \quad , \tag{3}$$

where P_i^0 is its initial position, $\delta_{j1}, \delta_{j2}, \delta_{j3}$ are the deformation vectors associated to the control triangulation vertices to which vertex i is attached, and l_1^i, l_2^i, l_3^i are the precomputed barycentric coordinates.

In this fashion, the shape of the surface triangulation becomes a function of the δ_j and the parameter vector P of Equation 1 is taken to be the vector of the x, y and z components of these δ_j. Because the control triangulations have fewer vertices that are more regularly spaced than the surface triangulation, the least-squares optimization has better convergence properties. Of course the finer the control triangulation, the less smoothing it provides. By using increasingly refined control triangulations, we implement a hierarchical fitting scheme that has proved very useful when dealing with noisy data, as discussed in Section 3.

Because there may be gaps in the image data, it is necessary to add a small stiffness term into the optimization to ensure that the δ_j of control vertices located where there is little or no data are consistent with their neighbors. If the surface was continuous, we could

take this term to be

$$\iint \left(\frac{\partial}{\partial u} \delta(u,v) \right)^2 + \left(\frac{\partial}{\partial v} \delta(u,v) \right)^2 du\, dv \quad .$$

However, because our control triangulation is discrete, we can treat its facets as C^0 finite elements and write our stiffness term as

$$E_S = \Delta_x^t K \Delta_x + \Delta_y^t K \Delta_y + \Delta_z^t K \Delta_z \tag{4}$$

where K is a stiffness matrix and Δ_x, Δ_y and Δ_z are the vectors of the x, y and z coordinates of the displacements δ. The term we actually optimize becomes

$$E = \sum_{1 \leq i \leq nobs} w_i \epsilon_i^2 + \lambda_S E_S \quad , \tag{5}$$

where λ_S is a small positive constant. This is achieved very simply in the least squares framework by incrementing the appropriate elements of the matrix that appears in the normal equations by those of the stiffness matrix K.

3 From Image Data to Observations

In this section we focus on stereo range data and silhouettes because they can be readily acquired from image sequences and form two complementary sources of information: Stereo can be expected to give good results where the surface more or less faces the cameras while silhouettes appear where the surfaces slopes away from the camera planes.

3.1 Stereo Data

We use several sets of stereo pairs or triplets of a given face as our input data such as those of Figure 2. We assume that the images are monochrome and registered so that their relative camera models are known *a priori*. Since we are interested in reconstructing surfaces, we start the process by using a simple correlation-based algorithm [3] to compute a disparity map for each pair or triplet and by turning each valid disparity value into a 3–D point. If other sources of range data were available, they could be used in a similar fashion. Because, these 3–D points typically form an extremely noisy and irregular sampling of the underlying global 3–D surface, we begin by robustly fitting surface patches to the raw 3–D points. This first step eliminates some of the outliers and generates meaningful local surface information for arbitrary surface orientation and topology. For additional details on this procedure, we refer the interested reader to an earlier publication [4].

Note however, that because it is extremely difficult to design a stereo algorithm that never produces correlated artifacts, we cannot expect any robust fitting technique to exclude all erroneous 3–D points. Furthermore, fitting local surfaces to the initial data amounts to smoothing and may result in spurious patches that appear to line up with legitimate ones.

The center of each patch is treated as an attractor. The easiest way to handle it is to model it as a spring attached to the mesh vertex closest to it. This, however, is inadequate if

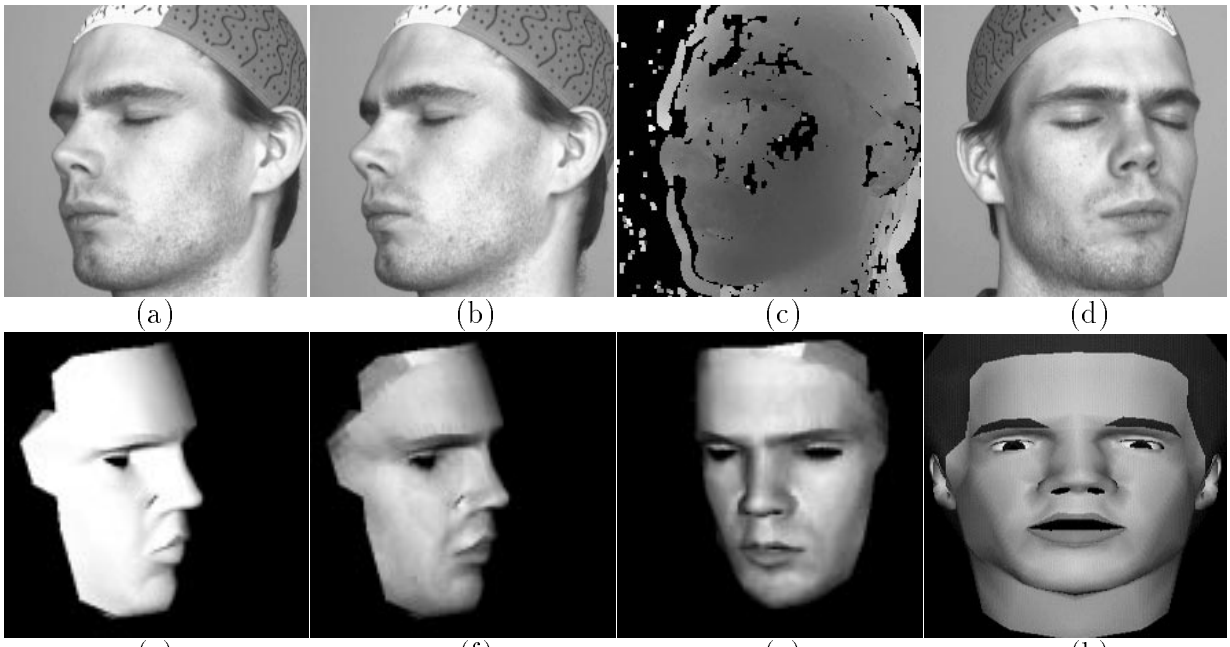


Figure 2: Modeling a head from a video sequence of forty images (Courtesy of IGP, ETHZ). (a,b) Consecutive images treated as a stereo pair. (c) Corresponding disparity map. Black indicates that no disparity value was computed; lighter areas are further away than darker ones. Note that the disparities around the occluding contour on both the left and right sides of the head are erroneous. (d) Image taken from different viewpoint. (e) Shaded view of the reconstructed model. (f,g) Shaded views using the facet reflectances derived from the images. (h) Adding synthetic hair and using the animation model to generate a new facial expression.

one wishes to use facets that are large enough so that attracting the vertices, as opposed to the surface point closest to the attractor, would cause unwarranted deformations of the mesh. This is especially important when using a sparse set of attractors. In our implementation, this is achieved by writing the observation equation as

$$d_i^a = 0 + \epsilon_i \quad , \tag{6}$$

where d_a is the orthogonal distance of the attractor to the closest facet and can be computed as a function of the x,y, and z coordinates of the vertices of the facet closest to the attractor.

Finding this "closest facet" is computationally expensive if we exhaustively search the list of facets for the one that initially minimizes the observation error of Equation 6. However, the search can be made efficient and fast if we assume that the 3–D points can be identified by their projection in an image, as is the case with stereo data. For each image, we compute what we call a "Facet-ID image:" We encode the index i of each facet f_i as a unique color, and project the surface into the image plane, using a standard hidden-surface algorithm. We can then trivially look up the facet that projects at the same place as a given point.

We recompute these attachments at each stage of the hierarchical fitting scheme of Section 2, that is each time we introduce a new control triangulation. Because a number of the patches derived from stereo may be spurious, we use a variant of the Iterative Reweighted

Least Squares [1] technique. Each time we recompute the attachments, we also recompute the weight w_i of observation i and take it to be inversely proportional to the initial distance d_i^a of the data point to the surface triangulation. More specifically we compute w_i as

$$w_i = exp(\frac{-d_i^a}{\overline{d^a}}) \quad \text{for } 1 \leq i \leq n \tag{7}$$

where $\overline{d^a}$ is the median value of the d_i. We use $\overline{d^a}$ as an estimate of the noise variance and we discount the influence of points that are more than a few standard deviations away.

Its robustness can be further increased by multiplying the w_i by the dot product of the normal of the surface patches used to derive the attractors by the current estimate of the normal vector of the facet to which the facet is attached. In this manner, the influence of patches whose orientation is very different from that of the attached facet are discounted.

3.2 Silhouette Data

Contrary to 3–D edges, silhouette edges are typically 2–D features since they depend on the viewpoint and cannot be matched across images. However, they constrain the surface tangent. Each point of the silhouette edge defines a line that goes through the optical center of the camera and is tangent to the surface at its point of contact with the surface. The points of a silhouette edge therefore define a ruled surface that is tangent to the surface. In terms of our facetized representation, this can be expressed as follows. Given a silhouette point (u_s, v_s) in an image, there must be a facet with vertices $(x_i, y_i, z_i)_{1 \leq i \leq 3}$ whose image projections $(u_i, v_i)_{1 \leq i \leq 3}$, as well as (u_s, v_s), all lie on a single line. This can be enforced by writing for each silhouette point three observation equations:

$$\begin{vmatrix} u_i & u_j & u_s \\ v_i & v_j & v_s \\ 1 & 1 & 1 \end{vmatrix} = 0 + \epsilon_{ij} \quad , 1 \leq i \leq 3 \ , \ i \leq j \leq 3 \tag{8}$$

where the (u_i, v_i) are derived from the (x_i, y_i, z_i) using the camera model.

As with the 3–D attractors of Section 3.1, we can use the iterative reweighting scheme described above and the main problem is to find the "silhouette facet" to which the constraint applies. As before, we can either exhaustively search the facets of the surface triangulation for those that initially minimize the observations errors of Equation 8 or use the Facet-ID image to speed up the process: Since the silhouette point (u_s, v_s) can lie outside the projection of the current estimate of the surface, we search the Facet-ID image in a direction normal to the silhouette edge for a facet that minimizes e_s and that is therefore the most likely to produce the silhouette edge. This, in conjunction with our coarse-to-fine optimization scheme, has proved a robust way of determining which facets correspond to silhouette points.

3.3 Reflectance Data

To estimate the reflectance of the facets of the model, we project them into the images and compute the mean gray-level of the pixels belonging to these projections. Here again

we take advantage of the Z-buffering capability of our machines to perform this operation quickly while taking occlusions into account.

4 Results

We first illustrate the effectiveness of the fitting technique using surface patches computed from good-quality stereo data as described in Section 3.1.

Figure 2 depicts a sequence of forty 512x512 images that were acquired with a video camera over a period of a few seconds by turning around the subject who was trying to stand still. Camera models were later computed using standard photogrammetric techniques at the Institute for Geodesy and Photogrammetry, ETH-Zürich. We ran our correlation-based algorithm [3]—once for each consecutive pair of images in the sequence— stored the resulting 3–D points in a 80x80x80 set of voxels and instantiated patches in all voxels containing at least 200 points. Fitting the animation model of Figure 1 to these patches produced the results depicted by the second row of Figure 2.

Similarly, we can model the face of the character of Figure 3 using a stereo pair taken at close range. Because we use the same models for the two faces, it becomes trivial to morph one into the other as shown in Figure 4.

(a) (b) (c)

Figure 3: High quality stereo pair (Courtesy of INRIA). (a,b) Left and right images (c) Shaded view of the reconstructed model

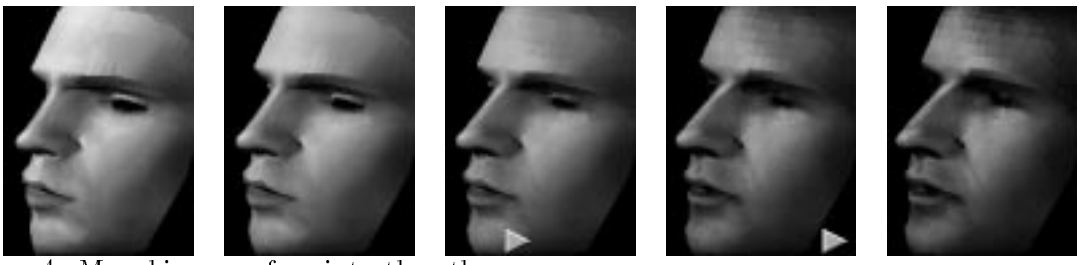
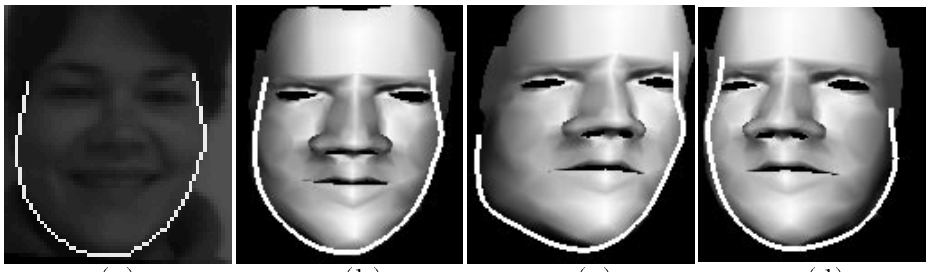


Figure 4: Morphing one face into the other

The image of Figure 5 is of much lower resolution that the ones shown above. As a result, our correlation-based stereo information does not provide any meaningful information for the side of the face and chin. However, by outlining the silhouette of the chin using snakes and treating these silhouette points as observations, we were able to reconstruct the complete face in a way that is consistent with those silhouettes.



(a) (b) (c) (d)

Figure 5: Using Silhouettes. (a) One image of a triplet. The silhouette of the chin is outlined in white (Courtesy of INRIA). (b,c,d) Shaded views of the reconstructed model shown in the same perspective as the original images with overlaid silhouettes.

5 Conclusion

We have presented a technique that allows us to fit a complex animation model to noisy image data with very limited manual intervention. As a result, these models can be produced cheaply and fast. Furthermore, because our approach relies on the use of a coarse to fine control triangulations, it can be used to fit arbitrarily complex models whose topology is designed for animation purposes and are not necessarily well suited for surface reconstruction.

In future work, we intend to extend the approach to the modelling of dynamic faces and more importantly to the estimation not only of the parameters that control the shape of the surface but also of those that control the various facial expressions.

References

[1] A. E. Beaton and J.W. Turkey. The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data. *Technometrics*, 16:147–185, 1974.

[2] D. DeCarlo and D. Metaxas. The Integration of Optical Flow and Deformable Models with Applications to Human Face Shape and Motion Estimation. In *Conference on Computer Vision and Pattern Recognition*, pages 231–238, 1996.

[3] P. Fua. A Parallel Stereo Algorithm that Produces Dense Depth Maps and Preserves Image Features. *Machine Vision and Applications*, 6(1):35–49, Winter 1993.

[4] P. Fua. From Multiple Stereo Views to Multiple 3–D Surfaces. *International Journal of Computer Vision*, 1996. Accepted for publication.

[5] P. Kalra, A. Mangili, N. Magnenat Thalmann, and D. Thalmann. Simulation of Facial Muscle Actions Based on Rational Free Form Deformations. In *Eurographics*, 1992.

[6] Y. Lee, D. Terzopoulos, and K. Waters. Realistic Modeling for Facial Animation. In *Computer Graphics, SIGGRAPH Proceedings*, pages 191–198, Los Angeles, CA, August 1995.