

# From Regular Images to Animated Heads: A Least Squares Approach

P. Fua and C. Miccio

Computer Graphics Lab (LIG)  
EPFL  
CH-1015 Lausanne  
Switzerland  
fua@lig.di.epfl.ch

**Abstract.** We show that we can effectively fit arbitrarily complex animation models to noisy image data. Our approach is based on least-squares adjustment using of a set of progressively finer control triangulations and takes advantage of three complementary sources of information: stereo data, silhouette edges and 2-D feature points.

In this way, complete head models—including ears and hair—can be acquired with a cheap and entirely passive sensor, such as an ordinary video camera. They can then be fed to existing animation software to produce synthetic sequences.

## 1 Introduction

In this paper, we show that we can effectively fit a complex head animation model, including ears and hair, to image data obtained using simple video or CCD 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. There are many effective approaches to recovering face geometry. They rely on stereo [5], shading [14], structured light [19], silhouettes [20] or low-intensity lasers. Some of these systems such as the C3D<sup>tm1</sup> or the Cyberware<sup>tm</sup> scanner are commercially available. However, recovering a head as a simple triangulated mesh does not suffice: To animate the face, one must further fit an actual animation model to the data.

Automated approaches to this task 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. [4]).
- Others use sophisticated face models with large numbers of degrees of freedom that are suitable for animation purposes but require very clean data—the kind produced by a laser scanner or structured light—to instantiate them (e.g. [15]).

---

<sup>1</sup> Turing Institute, Glasgow

The approach proposed here bridges the gap between these two classes by fitting to actual image data a detailed face model that has successfully been used to animate virtual actors. 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.

Our contribution is twofold:

- 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. In particular, we will show that a complete model can be constructed from a sequence acquired with a simple hand-held video camera.
- The least-squares framework we have developed allows us to pool several kinds of heterogeneous information sources—stereo or range, silhouettes and 2-D feature locations—without deterioration in the convergence properties of the algorithm. This is in contrast to typical optimization approaches whose convergence properties tend to degrade when using an objective function that is the sum of many incommensurate terms [10, 9].

These attributes of our approach allow us to derive convincing models even from relatively low-resolution images.

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. Furthermore recent work in the area of autocalibration [6, 21, 17] could be brought to bear to automate the process. 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. Optionally, we also use semi-automated techniques to extract silhouettes and a small number of feature points.
- 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 data. We weigh the data points according to how close—in the least squares sense—they are to 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 the remainder of the paper, we first introduce our optimization framework. We then show how the various sources of information are handled. Finally we present reconstruction and animation results on a number of different heads.

## 2 Least Squares Framework

In this work, we use the facial animation model that has been developed at University of Geneva and EPFL [12]. 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 corresponding to abstract entities such as speech 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 it conforms to the image data. In standard least-squares fashion, we will use this data to write *nobs* observation equations of the form

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

where  $P$  is a parameter vector that defines the shape of the surface and  $\epsilon_i$  is the deviation from the model. We will then minimize

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

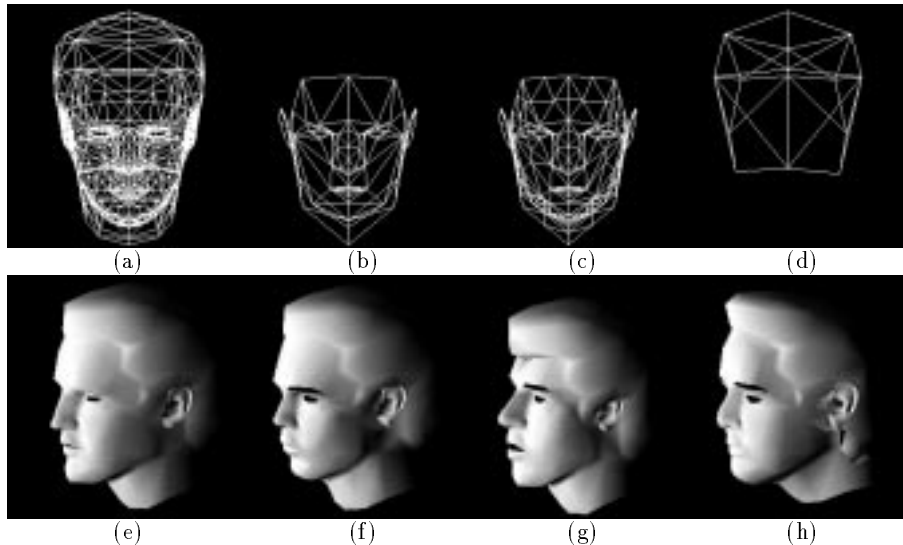
where  $w_i$  is a weight associated to each observation. The optimization is then performed using the Levenberg-Marquardt algorithm [18].

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 its large number of vertices, we have found the fitting process to be very brittle and the smoothing coefficients difficult to adjust. Therefore, we have developed the following scheme to achieve robustness.

### 2.1 Control Triangulation

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. The triangulation can thus be deformed to fit image data and to produce results such as those shown in the second row of Figure 1.

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



**Fig. 1.** Animation model. (a) The model used to animate complete heads. (b,c) Two levels of refinement of the control mesh used to deform the face. (d) The control mesh used to deform the hair. (e) A shaded view of the model in its default state. (f,g,h) Shaded views of the model deformed to resemble three of the people shown in the result section.

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, the surface triangulation’s shape is defined by deformation vectors associated to the vertices of the control triangulation. The 3-D position  $P_i$  of vertex  $i$  of the surface triangulation is taken to be

$$P_i = P_i^0 + l_1^i \delta_{j_1} + l_2^i \delta_{j_2} + l_3^i \delta_{j_3} \quad , \quad (3)$$

where  $P_i^0$  is its initial position,  $\delta_{j_1}, \delta_{j_2}, \delta_{j_3}$  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  $\delta_j$  and the parameter vector  $P$  of Equation 1 is taken to be the vector of the  $x, y$  and  $z$  components of these  $\delta_j$ . 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 a precomputed set of increasingly refined control triangulations, we implement a hierarchical fitting scheme that has proved very useful when dealing with noisy data, as shown in Section 3. Two levels of refinement of the control mesh used to deform the face and the coarsest level of the control mesh used to deform the hair are shown in the top row of Figure 1.

## 2.2 Stiffness Matrix

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  $\delta_j$  of the 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

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

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

$$E_S = \Delta_x^t K \Delta_x + \Delta_y^t K \Delta_y + \Delta_z^t K \Delta_z \quad (4)$$

where  $K$  is a stiffness matrix and  $\Delta_x, \Delta_y$  and  $\Delta_z$  are the vectors of the  $x, y$  and  $z$  coordinates of the displacements  $\delta$ . The term we actually optimize becomes

$$E = \sum_{1 \leq i \leq n_{obs}} w_i \epsilon_i^2 + \lambda_S E_S , \quad (5)$$

where  $\lambda_S$  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  $K$ .

## 2.3 Weighing the Observations

As will be discussed in Section 3, our system must be able to deal with many observations coming from different sources that may not be commensurate with each other. Formally we can rewrite the observations equations of Equation 1 as

$$f_i^{type}(P) = obs_i^{type} + \epsilon_i , 1 \leq i \leq n_{obs} , \quad (6)$$

with weight  $w_i^{type}$ , where *type* is one of the possible types of observations we use. In this paper, *type* may be stereo, silhouette position or 2-D feature location. The least-squares minimization procedure involves iteratively solving normal equations and solving linear systems of the form

$$A^t A X = A^t Y$$

where the  $A$  matrix is formed by summing the elements of the gradient vectors  $w_i^{type} \nabla f_i^{type}(P)$ . To ensure that these matrices are well conditioned and that the minimization proceeds smoothly we multiply the weight  $w_i^{type}$  of the  $n_{type}$  individual observations of a given type by a global coefficient  $w_{type}$  computed as follows:

$$G_{type} = \frac{\sqrt{\sum_{1 \leq i \leq n_{obs}, j = type} w_i^{type} \|\nabla f_i^j(P)\|^2}}{n_{type}} \quad (7)$$

$$w_{type} = \frac{\lambda_{type}}{G_{type}}$$

where  $\lambda_{type}$  is a user supplied coefficient between 0 and 1 that indicates the relative importance of the various kinds of observations. This guarantees that, initially at least, the magnitudes of the gradient terms for the various types have the appropriate relative values.

As shown in Section 4, the final result is relatively insensitive to the exact values of the  $\lambda_{type}$  and we have used the same set of values to generate all of our results. Because the complete objective function is not convex, no numerical optimization technique can be guaranteed to find the global optimum at each level of refinement of the control triangulations. The experiments described in Section 4, however, indicate that the Levenberg-Marquardt optimizer consistently yields better minima and needs fewer iterations at each level than other methods such as the Euler-Lagrange methods that have been extensively used in the Computer Vision field [13]—the so-called snake approaches—and one of the best known implementations of a quasi-newton approach in the field of Operations Research [16].

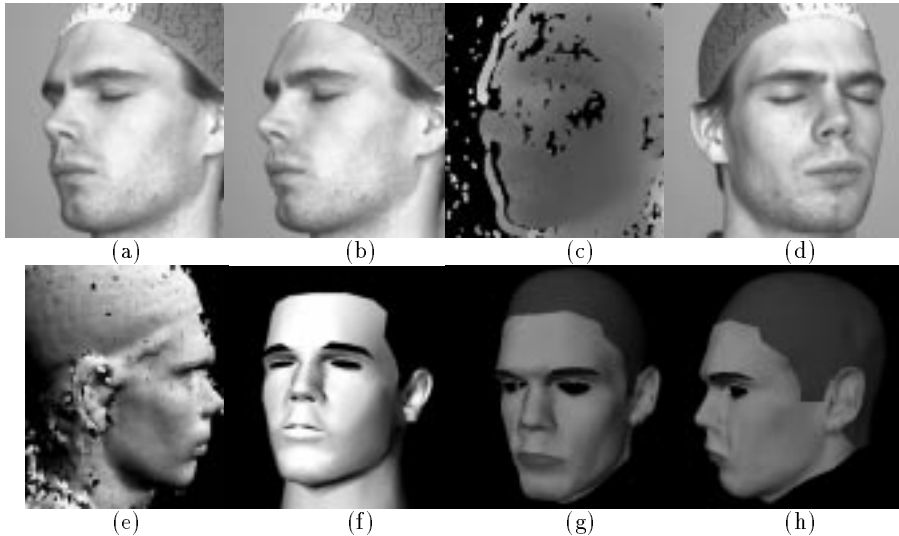
### 3 From Image Data to Observations

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. We also use a few key feature points to pin down some of the crucial elements of the model.

#### 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 [7] 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. Figure 2(e) depicts the result of this procedure. For additional details, we refer the interested reader to an earlier publication [8].

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



**Fig. 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 a different viewpoint. (e) Planar patches extracted from the stereo data represented as disks and shaded according to the orientation of their normals. The shape is recovered but there still are a number of spurious patches and the model cannot be animated as this stage of the processing. (f) Shaded view of the animation model fitted to the face only. (g,h) Shaded views using the facet reflectances derived from the images for the complete head.

a sparse set of attractors. In our implementation, this is achieved by writing the observation equation as

$$d_i^a = 0 + \epsilon_i \quad , \quad (8)$$

where  $d_i^a$  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.

The normal vector to a facet can be computed as the normalized cross product of the vectors defined by two sides of that facet, and  $d_i^a$  as the dot product of this normal vector with the vector defined by one of the vertices and the attractor. Letting  $(x_i, y_i, z_i)_{1 \leq i \leq 3}$  be the three vertices of a facet, consider the polynomial  $D$  defined as

$$D = \begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_a & y_a & z_a & 1 \end{vmatrix} = C_x x_a + C_y y_a + C_z z_a \quad (9)$$

where  $C_x, C_y$ , and  $C_z$  are polynomial functions of  $x_i, y_i$ , and  $z_i$  for  $1 \leq i \leq 3$ . It is easy to show that the facet normal is parallel to the vector  $(C_x, C_y, C_z)$  and that the orthogonal distance  $d_i^a$  of the attractor to the facet becomes

$$d^a = D / \sqrt{C_x^2 + C_y^2 + C_z^2}$$

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 8. 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 use the Z-buffering capability of our machines to 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 some 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  $w_i^{stereo}$  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  $w_i^{stereo}$  as

$$w_i^{stereo} = \exp\left(\frac{-d_i^a}{\bar{d}^a}\right) \quad \text{for } 1 \leq i \leq n \quad (10)$$

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

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 with 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 is 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:

$$\begin{vmatrix} u_i & u_j & u_s \\ v_i & v_j & v_s \\ 1 & 1 & 1 \end{vmatrix} = 0 + \epsilon_{ij} \quad , 1 \leq i \leq 3 \quad , \quad i \leq j \leq 3 \quad (11)$$

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 exhaustively search the facets of the surface triangulation for those that initially are close to being parallel to the ruled surface defined by the silhouette and minimize the observations errors of Equation 11. Alternatively, we can 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 must search the Facet-ID image in a direction normal to the silhouette edge for a facet that minimizes the initial observation error. 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 2-D Location of Features

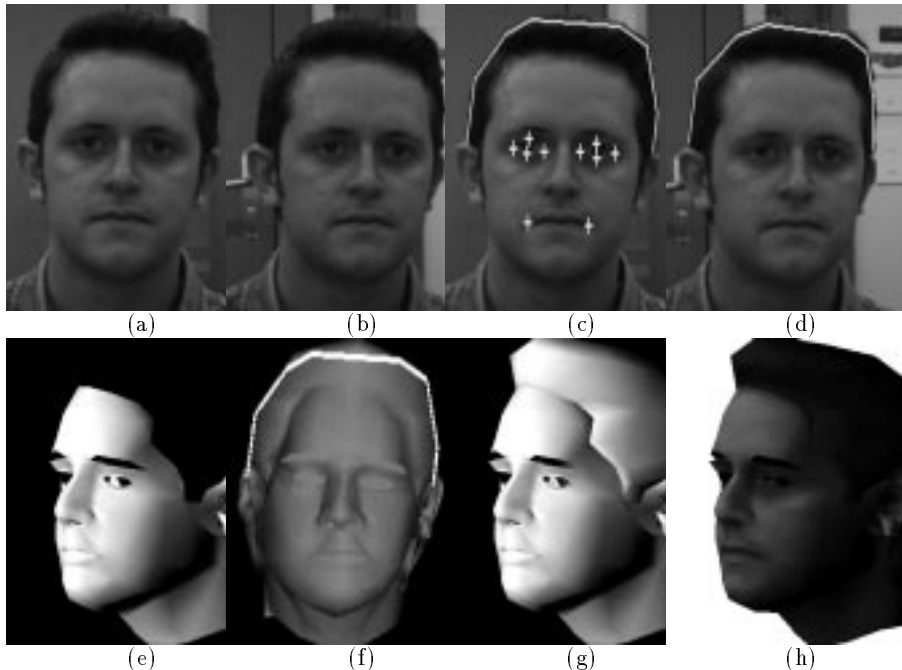
In order to ensure proper animation, it is important to guarantee that important features of the model—mouth and corners of the eyes especially—project at the correct locations in the face. Because there now are automated techniques to effectively extract those [2, 3, 11], a system like ours must be able to take advantage of those whenever available. For each vertex  $x_i, y_i, z_i$  of the *surface triangulation* whose 2-D projection  $u_i, v_i$  is known, we can write two observation equations:

$$\begin{aligned} Pr_u(x_i, y_i, z_i) &= u_i + \epsilon_i^u \\ Pr_v(x_i, y_i, z_i) &= v_i + \epsilon_i^v \quad , \end{aligned} \quad (12)$$

where  $Pr_u$  and  $Pr_v$  stand for the projection in  $u$  and  $v$ . In this way we do not need the explicit 3-D position of these feature points, only their 2-D image location.

### 3.4 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.



**Fig. 3.** Modelling the complete head. (a,b) Two images from a triplet. (c,d) 2-D feature points and hair outlines (e) Reconstructed face. (f) Hair optimized to conform to the outlines of (c) and (d). (g) Shaded view of the complete head. (h) Shaded view using the albedoes computed from the images.

## 4 Results

Figures 2(f,g,h) demonstrates the effectiveness of the fitting technique using only high-quality stereo data as described in Section 3.1.

To illustrate the use of silhouettes and feature points, we use the images of Figure 3. By supplying a small number of feature points and providing an outline for the hair—where stereo fails—we were able to reconstruct both the complete face and the hairdo as follows: We start with three image triplets and use stereo and feature data to reconstruct the face to yield the result shown in Figure 3(e). The hair outlines of Figure 3(c) and (d) can then be treated as silhouettes. They are used, in conjunction with the stereo data and the *hair control triangulation* of Figure 1(d) to deform the part of *the surface triangulation* that corresponds to hair and yield the results of Figure 3(f,g,h) Using the same approach and the same parameters and three stereo pairs for each person, we obtained the three complete heads of Figure 4.

In all cases the heads can be animated by invoking the animation controller [12], as depicted by Figure 5.

The results of Figures 2, 3, and 4 have all been computed using the same value (0.005) for the stiffness parameter  $\lambda_S$  of Equation 5. In the general case,

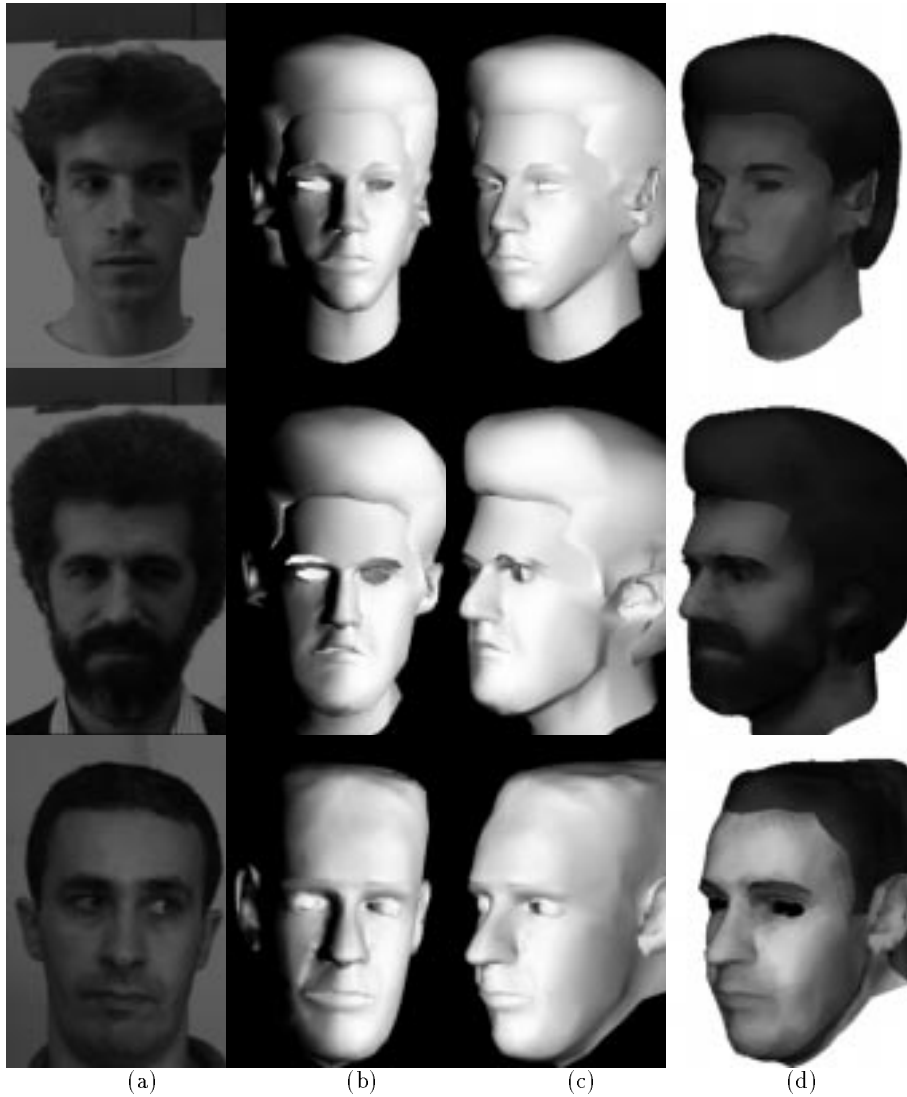


Fig. 4. Three LIG researchers. (a) Left images of one of the three stereo pairs used for each person. (b,c) Shaded views of the complete head. (d) Shaded view using the albedoes computed from the images. All the models have the same topology and can be animated as those of Figure 1.

these results may involve the combination of several sources of information—stereo, silhouette position and 2-D feature location—and require the setting of relative weights, the  $\lambda_{type}$  of Section 3.2. In all cases we have used  $\lambda_{stereo} = 1.0$  and  $\lambda_{silhouette} = \lambda_{features} = 0.3$ . To demonstrate the relative insensitivity of the results to the precise value of these weighing parameters, we have run the face re-

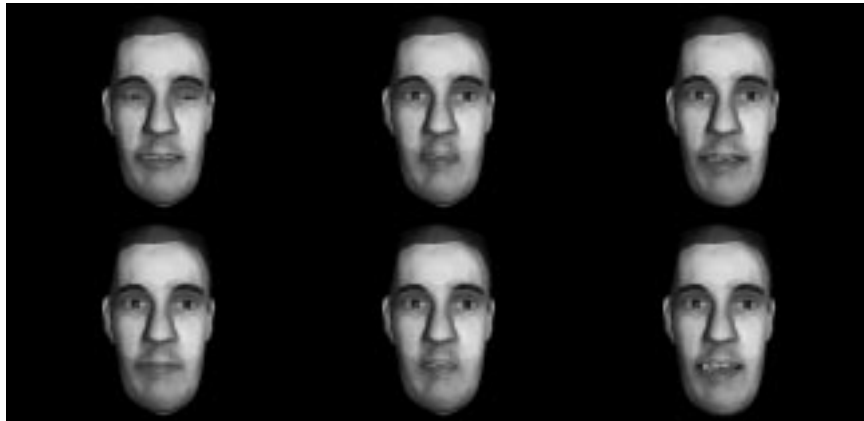


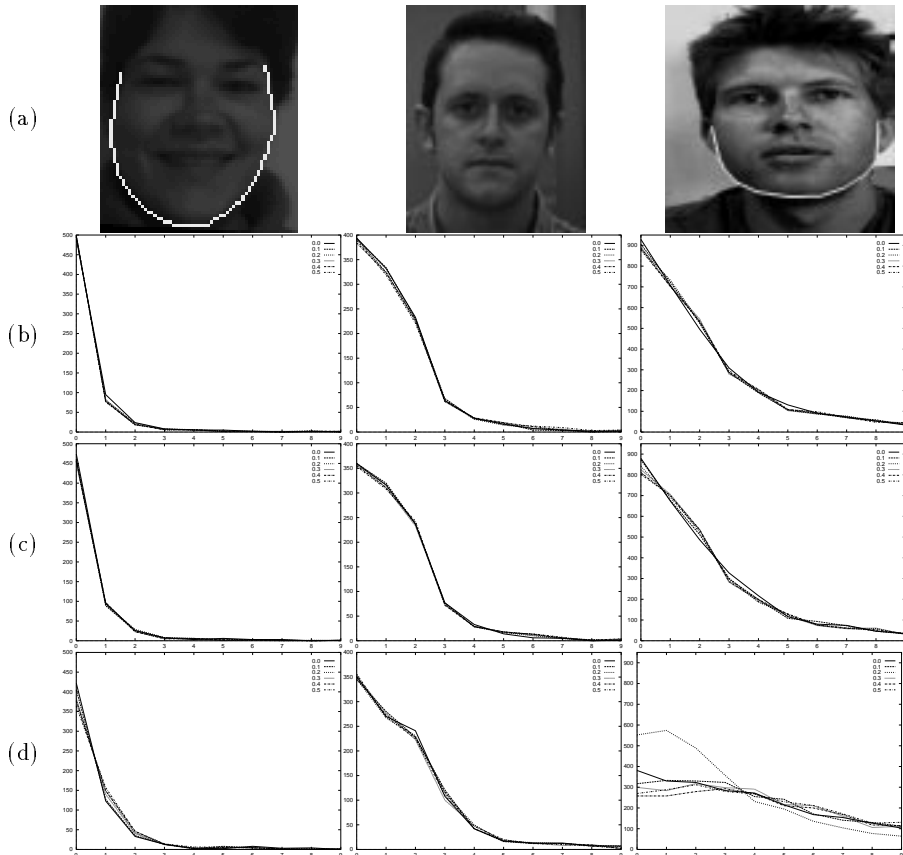
Fig. 5. Animating the face: the third character of Figure 4 opens his eyes and talks.

construction algorithm on the images of Figure 6(a) several times with different parameters setting. We have used both chin silhouettes and feature points, with values of  $\lambda_{silhouette}$  and  $\lambda_{features}$  ranging from 0.0 to 0.5 and  $\lambda_{stereo}$  remaining equal to 1.0. In Figure 6(b), we histogram the distances of the 3-D data points derived from stereo to the resulting face surfaces. Note that for each face, the five plots are almost indistinguishable. In other words, the observations derived from the points and the silhouettes affect the result for the surface vertices on which they act, without negatively affecting the convergence properties of the remaining parts the surface. For comparison's sake, we have tried to recompute the face surface's shape using the *same* weighing parameters and the *same* schedule for refining the control triangulations but different optimization techniques:

1. the Euler Lagrange approach found in many snake-like approaches,
2. the MINOS nonlinear optimizer [16] which is considered as one of the most effective such algorithm in the field of operations-research.

We plot the results in Figure 6(c) and (d).

A perfect result is one where the majority of 3-D data points are very close to the recovered surface and therefore their distance to it is almost zero and where a minority of points, the outliers in the stereo data, are much further. The plots of Figure 6(b) are closer to that ideal than those of Figure 6(c,d), especially those derived using MINOS. Although the difference is less obvious, the graphs derived using the Euler-Lagrange approach also exhibit slightly worse convergence properties: there are fewer points whose distance to the surface is almost zero. Furthermore, as shown in Figure 7, the median value of the distances of those points to the surface is also slightly higher. A theoretical understanding of this behavior would require a very sophisticated applied mathematics treatment of the objective function and is therefore beyond the scope of this paper. We conjecture that the use of a stiffness matrix by both the standard least squares approach and the Euler-Lagrange one allows a rapid propagation of the



**Fig. 6.** Gauging the performance of the optimizer: (a) Three people and corresponding silhouette data for the chin. For all three, we have used the feature points depicted by Figure 3(c). (b) For each person’s face, we histogram in the distance of the optimized face surface to the 3-D data-points derived from stereo for  $\lambda_{stereo} = 1.0$  and  $\lambda_{silhouette} = \lambda_{feature} = x$ ,  $0.0 \leq x \leq 0.5$ . Note that for each of the three images, the various plots are almost indistinguishable thereby illustrating the relative insensitivity of the result to exact value of  $\lambda_{silhouette}$  and  $\lambda_{feature}$ . (c) Graphs similar to those shown in row (b) but derived by performing the optimization using the Euler-Lagrange approach instead of the Levenberg-Marquardt approach. There are fewer points in the histogram bins that denote a distance close to zero than in the corresponding ones in row (b). (d) Graphs derived using the MINOS optimizer.

smoothness constraints across the triangulation and thus improves convergence, as was claimed in the original snake paper [13]. We further hypothesize that the difference between these two methods, lies in the fact that the Levenberg-Marquardt algorithm does not explicitly compute the gradient of the sum of the squares of the observations. Instead, it computes the individual gradients of the observations—and not their squares—and uses them to construct the matrices

Levenberg-Marquardt	2.33	6.38	0.55
Euler-Lagrange	2.58	6.87	0.60
MINOS	2.95	7.40	1.40

**Fig. 7.** Median distance. For the three images of Figure 6(a), we indicate the median distance of the 3-D points derived from stereo to the final face surface for  $\lambda_{stereo} = 1.0$  and  $\lambda_{silhouette} = \lambda_{feature} = 0.0$ .

that appear in the normal equations.

## 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 triangulation, 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. We will also incorporate self-calibration techniques into our framework so that we can achieve this result using ordinary video sequences filmed using regular camcorders.

Furthermore, the face model we use has been one of the starting points for the facial animation parameters defined in the MPEG-4 FBA work. When standardization is complete, it will therefore be easy to make the parameters of our model conformant with the MPEG-4 norm for facial animation and the work presented here will become directly relevant to video transmission.

## Acknowledgements

We wish to thank Prof. Nadia Magnenat Thalmann, Prof. Daniel Thalmann and Dr. Prem Kalra for having made their facial animation model available to us. We are also indebted to Prof. Grün for sharing with us his insights about least-squares technology.

## 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. A. Blake and M. Isard. 3D Position Attitude and Shape Input Using Video Tracking of Hands and Lips. In *Computer Graphics, SIGGRAPH Proceedings*, pages 71–78, July 1994.

3. T.F. Cootes and C.J. Taylor. Locating Objects of Varying Shape Using Statistical Feature Detectors. In *European Conference on Computer Vision*, Cambridge, England, April 1996.
4. 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.
5. F. Devernay and O. D. Faugeras. Computing Differential Properties of 3–D Shapes from Stereoscopic Images without 3–D Models. In *Conference on Computer Vision and Pattern Recognition*, pages 208–213, Seattle, WA, June 1994.
6. O.D. Faugeras, Q.-T. Luong, and S.J. Maybank. Camera self-calibration: theory and experiments. In *European Conference on Computer Vision*, pages 321–334, Santa-Margherita, Italy, 1992.
7. 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.
8. P. Fua. From Multiple Stereo Views to Multiple 3–D Surfaces. *International Journal of Computer Vision*, 24(1):19–35, 1997.
9. P. Fua and C. Brechbühler. Imposing Hard Constraints on Deformable Models Through Optimization in Orthogonal Subspaces. *Computer Vision and Image Understanding*, 24(1):19–35, February 1997.
10. P.E. Gill, W. Murray, and M.H. Wright. *Practical Optimization*. Academic Press, London a.o., 1981.
11. T.S. Jebara and A. Pentland. Parametrized Structure from Motion for 3D Adaptive Feedback Tracking of Faces. In *Conference on Computer Vision and Pattern Recognition*, pages 144–150, Porto Rico, June 1997.
12. P. Kalra, A. Mangili, N. Magnenat Thalmann, and D. Thalmann. Simulation of Facial Muscle Actions Based on Rational Free Form Deformations. In *Eurographics*, 1992.
13. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1(4):321–331, 1988.
14. Y. G. Leclerc and A. F. Bobick. The Direct Computation of Height from Shading. In *Conference on Computer Vision and Pattern Recognition*, Lahaina, Maui, Hawaii, June 1991.
15. 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.
16. B. Murtagh and M.A. Saunders. Minos 5.4 User’s Guide. Technical Report SOL 83-20R, Department of Operations Research, Stanford University, 1983.
17. M. Pollefeys, R. Koch, and L. VanGool. Self-Calibration and Metric Reconstruction In Spite of Varying and Unknown Internal Camera Parameters. In *International Conference on Computer Vision*, 1998.
18. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes, the Art of Scientific Computing*. Cambridge U. Press, Cambridge, MA, 1986.
19. M. Proesmans, L. Van Gool, and A. Oosterlinck. Active acquisition of 3D shape for Moving Objects. In *International Conference on Image Processing*, Lausanne, Switzerland, September 1996.
20. L. Tang and T.S. Huang. Analysis-based facial expression synthesis. *ICIP-III*, 94:98–102.
21. B. Triggs. Autocalibration and the Absolute Quadric. In *Conference on Computer Vision and Pattern Recognition*, pages 609–614, 1997.