Face Models from Uncalibrated Video Sequences

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

Abstract— **We show that we can effectively fit a very complex facial animation models to uncalibrated video sequences, without benefit of targets, structured light or any other active device.**

Our approach is based on regularized bundle-adjustment followed by least-squares adjustment using a set of progressively finer control triangulations. It takes advantage of three complementary sources of information: stereo data, silhouette edges and 2–D feature points.

In this way, complete head models 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.

Keywords— **3–D Modeling, Facial Animation, Uncalibrated, Bundle-Adjustment**

I. Introduction

In this paper, we show that we can effectively and automatically fit a complex facial animation model to uncalibrated video sequences, without benefit of targets, structured light or any other active device.

To do so, we have developed a three-step process. First, our algorithm automatically recovers the head's relative motion with respect to the camera. Second, it computes disparity maps for each pair of consecutive images, derives clouds of 3–D points and fits local surface patches to these raw 3–D points. Finally, it uses a least-square adjustment technique to fit a generic animation mask to the 3–D data.

In recent years much work has been devoted to the modeling of faces from image and range data. There are many effective approaches to recovering face geometry. They rely on stereo [1], shading [2], structured light [3], silhouettes [4] or low-intensity lasers. Some of these systems such as the C3D^{tm}[1] or the Cyberware^{tm} 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. [5]).
- Others use sophisticated face models with large numbers of degrees of freedom that are suitable for animation purposes but require very clean data to instantiate them. Such data can be produced by a laser scanner or structured light [6], [3] or come from semi-automatically entered feature points and silhouettes [7].

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 complete head model that can then be fed to existing animation software to create synthetic sequences.

Our contribution is twofold:

- We have introduced model-based regularization constraints in our bundle adjustment procedure that allows the effective recovery of the motion parameters, even though the quality of the points matches cannot be expected to very good.
- We have developed a robust fitting technique that allows the fitting of an animation mask with a great many degrees of freedom to noisy stereo data. The least-squares framework we propose 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 [8], [9].

This has proved essential in dealing with ordinary video sequences of faces that typically exhibit relatively little texture. Our approach thus allows the automated instantiation of sophisticated animation models using a cheap, entirely passive and readily available sensor. As more and more people have video cameras attached to their computers, our approach could be used to quickly produce clones for video-conferencing purposes. It will also allow us to exploit ordinary movies to reconstruct the faces of actors or famous people that cannot be easily be scanned using active techniques, for example because they are unavailable or long dead.

In the remainder of the paper, we first introduce the facial animation model we use. We then present our camera motion recovery algorithm. Finally we describe the model fitting algorithm and present reconstruction results for a number of different heads.

[1] Turing Institute, Glasgow

II. Face Model

In this work, we use the facial animation model that has been developed at University of Geneva and EPFL [10]. 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,b). We will refer to it as the *surface triangulation*. Our goal is to deform the surface without changing its topology. This is important because the facial animation software depends on the model's topology and its configuration files must be recomputed everytime it is changed, which is hard to do on an automated basis.

III. Relative Motion Recovery

To perform our computation, we first need to estimate the relative motion of the face with respect to the camera. In this work, we choose sequences in which the subjects keep a fairly neutral facial expression and we treat the head as a rigid object.

There are well established photogrammetric techniques, such as bundle-adjustment, to achieve this goal given reliable correspondences between the images [11]. In recent years, there also has been much work in the area of autocalibration [12], [13], [14] and effective methods to compute the epipolar geometry [15] and the trifocal tensor [16] from point correspondences have been devised. However, most of these methods assume that it is possible to run an interest operator such as a corner detector [14], [16] to extract from one of the images a sufficiently large number of points that can be reliably matched in the other images.

In our case, however, because of the lack of texture on faces, we cannot depend on such interest points and must expect that whatever points we extract can only be matched with relatively little precision and a high probability of error. To overcome this problem, we take advantage of our rough knowledge of the face's shape and regularize the standard bundle-adjustment technique as described below.

We assume that the intrinsic camera parameters remain constant throughout the video sequence. In theory, given high precision matches, bundle-adjustment can recover both intrinsic parameters and camera motion. In practice however, we cannot expect out points matches to be very precise. We have therefore chosen to roughly estimate these intrinsic parameters and to concentrate on the computation of the extrinsic ones: We use an approximate value for the focal length and assume that the principal point remains in the center of the image. By so doing we generate 3-D models that are affine transforms of the real heads. Furthermore, we have verified experimentally that, when the estimate of the camera's focal length is not too different from the true value, this affine transform is relatively close to being a rotation and a scaling.

A. Generic Bundle Adjustment

Let us assume that we are given n 3–D points that project into m images. We will refer to these points as *tie* points. For each point x_i, y_i, z_i that can be seen in image j, we write two observation equations:

$$\begin{aligned} Pr_u^j(x_i, y_i, z_i) &= u_i^j + \epsilon_{u_i^j} \\ Pr_v^j(x_i, y_i, z_i) &= v_i^j + \epsilon_{v_i^j} \end{aligned} \tag{1}$$

where $Pr_u^j(x, y, z)$ and $Pr_v^j(x, y, z)$ denote the two image coordinates of the projection of point (x, y, z) in image j and $\epsilon_{u_i^j}, \epsilon_{v_i^j}$ the projection errors to be minimized. For each j, Pr_u^j and Pr_v^j depend on the six external parameters that define the camera position and orientation.

These external parameters can be recovered by minimizing the observation errors in the least square sense, that is by minimizing the objective function $\mathcal{E}$:

$$\mathcal{E} = \sum_{1 \le i \le n} w_i \sum_{1 \le j \le m} \delta_i^j ((Pr_u^j(x_i, y_i, z_i) - u_i^j)^2 + (Pr_v^j(x_i, y_i, z_i) - v_i^j)^2)), \tag{2}$$

with respect to the value of these parameters and of the x, y and z coordinates. δ_i^j is either zero or one depending on whether the tie point i is visible in image j or not. w_i is a weight associated to point i. It is initially taken to be 1 for all points and can then be adjusted as discussed below. The solution can only be found up to a global rotation, translation and scaling. To remove this ambiguity, we fix the position of the first camera and one additional parameter such as the distance between the first and the second camera.

B. Initialization

One well known limitation of bundle adjustment algorithms is the fact that, in order to ensure convergence, one must provide initial values for both camera positions and x, y and z coordinates that are not too far from their true values.

To fulfill this requirement, we begin by retriangulating the surface of the generic face model introduced in Section II to produce the regular mesh shown in Figure 1(c). We will refer to it as the *bundle-adjustment triangulation*. We take its vertices to be our tie points and use their x, y and z coordinates as the initial values of the $(x_i, y_i, z_i)_{1 \le i \le n}$ of Equation 2.

To initialize the process for a video sequence such as the one shown in Figure 2, we manually supply the approximate position of five feature points in one reference image: nose tip, outer corners of the eyes and outer mouth corners as shown in Figure 3(a). We usually choose as our reference image one in which the subject faces the camera and we take this image to be image number one.

We represent the camera models for all the cameras as 3x4 projection matrices. To compute them, the algorithm then goes through the following steps:

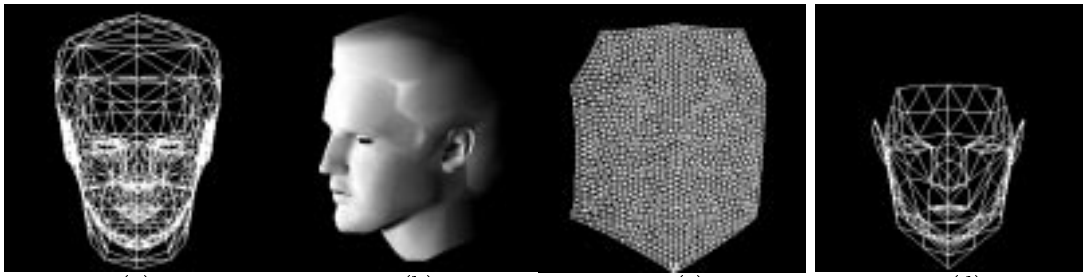


Fig. 1. Animation model. (a) Wireframe model used to animate complete heads. (b) Shaded view of the model (c) Regular sampling of the face used to perform bundle adjustment. (d) Control triangulation used to deform the face.

Fig. 2. Input video sequence: 6 of a sequence of 9 consecutive images of a video sequence used as input.

1. Generate an initial 3x4 projection matrix for the first camera with the principal point in the center of the image.

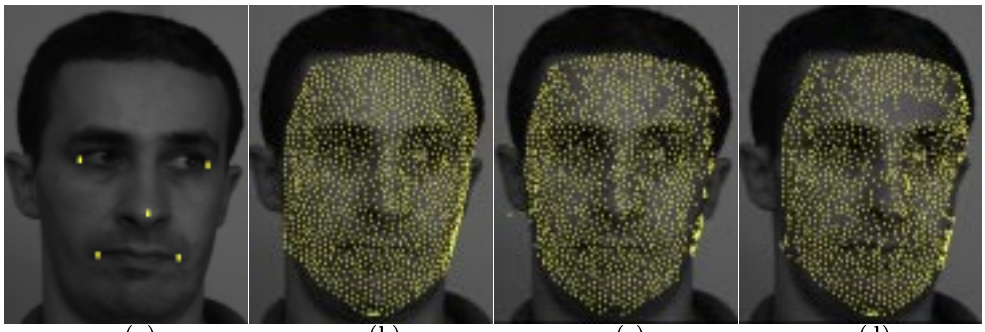
$$Prj = \left| \begin{array}{cccc} \mathrm{f} & 0 & \mathrm{xdim}/2 & 0 \\ 0 & \mathrm{f} & \mathrm{ydim}/2 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right| \quad , \qquad (3)$$

where $xdim, ydim$ are the image dimensions and f is an estimate of the focal length.

2. Compute a 4x4 rotation translation matrix M such that the five keypoints of Figure 3(a) once multiplied by this matrix project as close as possible to the hand-picked locations in the central image. The 3x4 matrix that represents the camera model for the first image is then be taken to be

$$Tr = Prj * M \quad , \qquad (4)$$

ensuring that the five keypoints project in the vicinity of the five hand-picked locations. As a consequence, all the other vertices of the bundle-adjustment triangulation also project on the face, as shown in Figure 3(b). These projections are taken to be the u_i^1, v_i^1 of Equa-



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

Fig. 3. Tie points: (a) The five manually supplied keypoints used to compute the orientation of the first camera. (b) The projections of the vertices of the bundle-adjustment triangulation of Figure 1(c) into the image of Figure 2(d). (c,d) The points in the images of Figure 2(c,e) that match those of (b). These matching points have been computed using a simple correlation algorithm.

tion 2. We then match these points in the two images that immediately precede and succeed the central image in the video sequence. We use a simple correlation-based algorithm [17] to obtain the $(u_i^j, v_i^j)_{2\leq j\leq 3}$ of Equation 2. Figure 3(c,d) depicts the results. Note that not all the points are matched and that there is a number of erroneous matches.

3. Take the initial positions of the other cameras to be equal to that of the first.
4. Given these initial values, use the Levenberg-Marquardt algorithm [18] to minimize the objective function $\mathcal{E}$ of Equation 2 with respect to the camera positions and the tie points' 3–D coordinates.

This yields the camera models for the two images on either side of the central image and an estimate of the bundle-adjustment triangulation's shape. To compute the following camera position, the image immediately succeeding the central image becomes the new central image. We project the bundle-adjustment triangulation's vertices into it, compute the matching points in the image that follows in the sequence and rerun the bundle-adjustment algorithm to compute the position of the corresponding camera. We then iterate until the end of the sequence. We proceed similarly for the images that precede the central image in the sequence.

C. Robust Bundle Adjustment

The last step of the procedure outlined above is regular bundle adjustment [11]. If the correspondences were perfect, this would be sufficient to retrieve the motion parameters. However the point correspondences can be expected to be noisy and to include mismatches. To increase the robustness of our algorithm, we augment the standard procedure in two ways:

1. **Iterative reweighted least squares** Because some of the point matches may be spurious, we use a variant of the Iterative Reweighted Least Squares [19] technique. We first run the bundle adjustment algorithm with all the weights w_i^j of Equation 2 being equal to 1. We then recompute these weights so that they are inversely proportional to the final residual errors. More specifically, for each tie point, we compute the average residual error ϵ_i:

$$\epsilon_i = \frac{\sum_j \delta_i^j (Pr_u^j(x_i, y_i, z_i) - u_i^j)^2 + (Pr_v^j(x_i, y_i, z_i) - v_i^j)^2}{\sum_j \delta_i^j} \ .$$

We take w_i to be

$$w_i = exp(\frac{-\epsilon_i}{\overline{\epsilon_i}}) \ ,$$

where $\overline{\epsilon_i}$ is the median value of the ϵ_i for $1 \leq i \leq n$. In effect, we use $\overline{\epsilon_i}$ as an estimate of the noise variance and we discount the influence of points that are more than a few standard deviations away.

2. **Regularization** The tie points are the vertices of our bundle-adjustment triangulation and represent a surface that is known to be smooth. We want to prevent excessive deformation by adding a regularization term $\mathcal{E}_D$ to the objective function $\mathcal{E}$ of Equation 2.
To compute this term, we first rewrite our observation equations as:

$$\begin{aligned} Pr_u^j(x_i + dx_i, y_i + dy_i, z_i + dz_i) &= u_i^j \\ Pr_v^j(x_i + dx_i, y_i + dy_i, z_i + dz_i) &= v_i^j \end{aligned} \quad (5)$$

where x_i, y_i, z_i are the vertex coordinates that are now fixed and dx_i, dy_i, dz_i are displacements that become the actual optimization variables.
If the surface was continuous, we could take $\mathcal{E}_D$ to be the sum of the square of derivatives of the dx_i,dy_i and dz_i across the surface. Because out bundle-adjustment triangulation is a triangulated surface, we can treat its facets as C^0 finite elements and evaluate $\mathcal{E}_D$ as follows: We introduce a stiffness matrix K such that

$$\mathcal{E}_D = 1/2(dX^t K dX + dY^t K dY + dZ^t K dZ) \ , \quad (6)$$

approximates the sum of the square of the derivatives of displacements across the triangulated surface when dX,dY and dZ are the vectors of the dx_i,dy_i and dz_i.

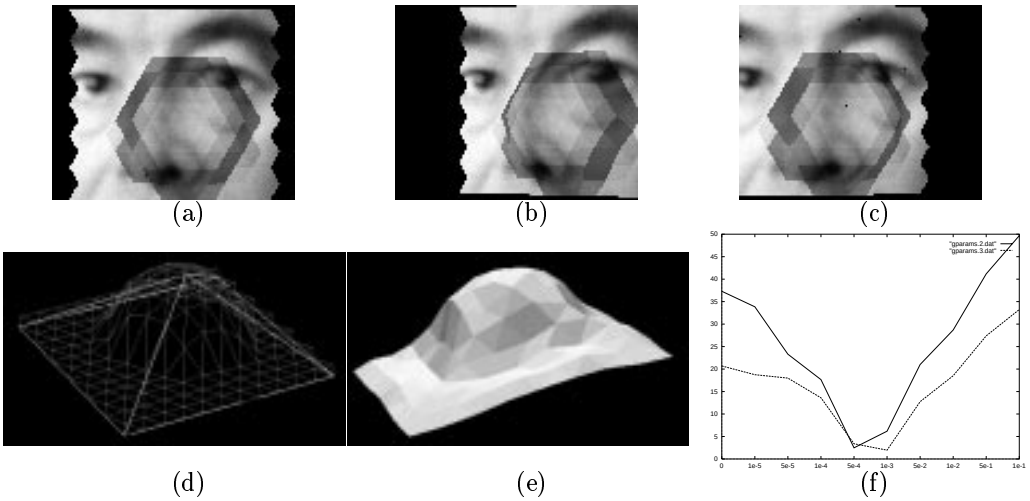


Fig. 4. Synthetic sphere images: (a,b,c) Three synthetic images of a textured half-sphere. (d) The "real" shape of the half sphere and the pyramid-shaped bundle-adjustment triangulation used in this experiment. (e) Recovered shape of the bundle-adjustment triangulation after recovery of the camera parameters for a give value of the regularization parameter λ. (f) Distance of the recovered position the camera centers to their true positions as a function of λ.

We can now enforce smoothness by minimizing the regularized objective function $\mathcal{E}_T$:

$$\begin{aligned} \mathcal{E}_T &= \mathcal{E} + \lambda \mathcal{E}_D \qquad (7) \\ \mathcal{E} &= \sum_i w_i \sum_j \delta_i^j ((Pr_u^j(x_i + dx_i, y_i + dy_i, z_i + dz_i) - u_i^j)^2 \\ &\quad + (Pr_v^j(x_i + dx_i, y_i + dy_i, z_i + dz_i) - v_i^j)^2) \ , \end{aligned}$$

where λ is a smoothing coefficient.

To illustrate the algorithm's behavior and to quantify the influence of the λ regularization parameter of Equation 7, we have used the three synthetic images shown in Figure 4. They were generated by texture mapping the image of a face on a half sphere seen from several known viewpoints. The underlying half-sphere is shown in Figure 4(d) along with the pyramid shaped bundle-adjustment triangulation we have used in this example. We fixed the position of the first camera, used a regular sampling of the bundle-adjustment triangulation as our tie points and ran our algorithm for the following range of values of λ: 0.0, 1e-5, 5e-5, 1e-4, 5e-4, 1e-4, 1e-3, 5e-3, 5e-2, 1e-2, 5e-1 and 1e-1. In Figure 4(e) we show the shape of the bundle-adjustment surface at the end of the optimization for λ =5e-4. In Figure 4(f) we plot, for each camera, the distance of the recovered position of the optical center to the real one as a function of λ. For λ too small, there is not enough regularization and the algorithm tends to overfit the noisy data resulting in large errors. For λ too large, the regularization term prevents the bundle-adjustment surface to deform adequately, also resulting in a poor result. There is however a wide range of values of λ— approximately between 1e-4 and 5e-4— that yield minimal error.

These results were produced using the correct internal parameters—focal-length and principal points—to instantiate the projection matrix of Equation 3. Using the same data, we have also verified that when we perform the computation using internal parameters that differ from the real ones, the recovered shape is related to the original one by an affine transformation. The position of the principal points has little influence and, for values of the focal length that are within 20% of the real value, the affine transform is close to being a rotation translation.

We now turn to the video sequence of Figure 2, we recover the camera positions shown in Figure 5(a). Figure 5(b,c) depicts the corresponding bundle-adjustment triangulation's shape. To check the quality of the result, we have used a Minolta^{tm} laser scanner to acquire the model of the same head shown in Figure 5(d,e). In theory, the bundle-adjustment triangulation and the output of the laser scanner—that we assume to be a fairly precise model of the real face—should have the same shape up to an affine transform. We have therefore computed the affine transform that brings the bundle-adjustment triangulation closest to the laser-scanner model. As shown in Figure 5(f,g,h), the superposition is good and the deformation introduced by the affine transform is not very severe. We have performed the same experiment on the images of Figure 7 with similar results.

IV. Model Fitting

Given the camera models computed above, we can now recover additional information about the surface by using a simple correlation-based algorithm [17] to compute a disparity map for each pair of consecutive images in the video sequences and by turning each valid disparity value into a 3–D point. 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.

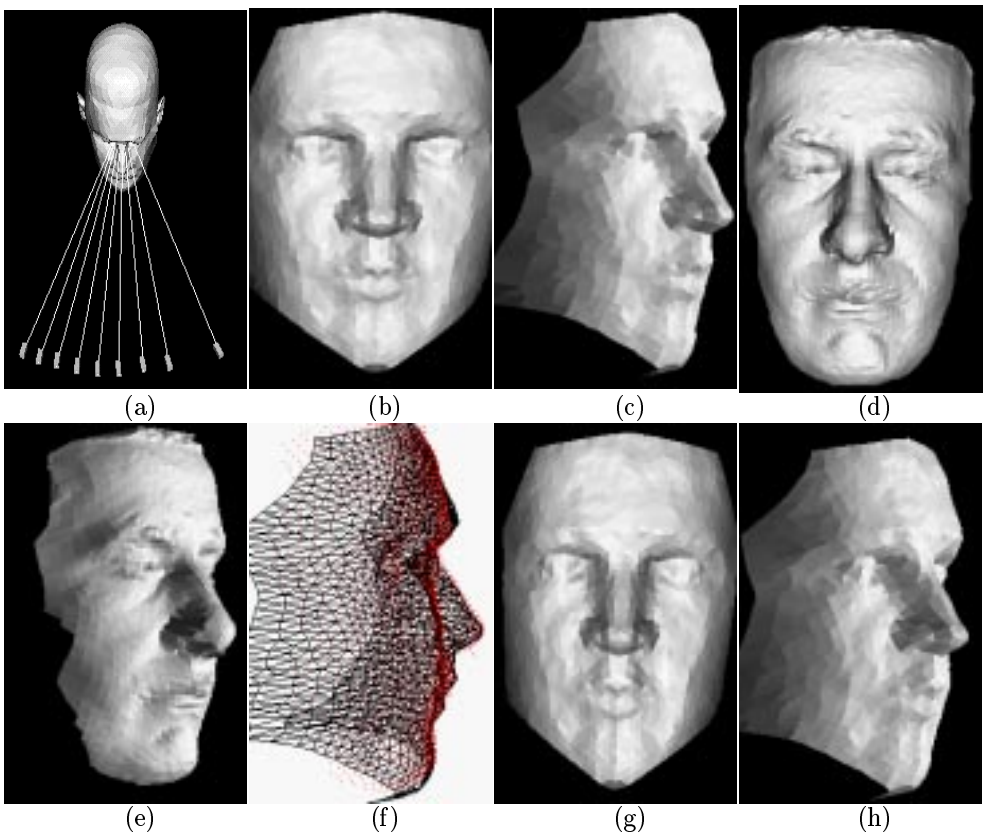


Fig. 5. Comparison with laser scanner data: (a) Recovered relative camera positions. (b,c) A shaded representation of the bundle-adjustment triangulation. (d,e) The laser scanner data. (f) The control triangulation, shown as wireframe, after an affine transform that projects it to the laser data, shown as red points. Note that they are well superposed (g,h) A shaded representation of the affine transformed bundle-adjustment triangulation.

For additional details, we refer the interested reader to an earlier publication [20].

Our goal then is to deform the generic mask so that it conforms to the cloud of points, that is to treat each patch as an attractor and to minimize its distance to the final mask. In our implementation, this is achieved by computing the orthogonal distance d_i^a of each attractor to the closest facet as a function of the x,y, and z coordinates of its vertices and minimizing the sum of the squares of these distances

$$E_A = \sum_i w_i (d_i^a)^2 \tag{8}$$

where w_i is a weight associated to each attractor.

Finding this "closest facet" is computationally expensive if we exhaustively search the list of facets. 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.

A. Control Triangulation

In theory we could optimize with respect to the state vector P 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.

Instead of directly modifying the vertex positions during the minimization, we introduce *control triangulations* such as the ones shown in Figure 1(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

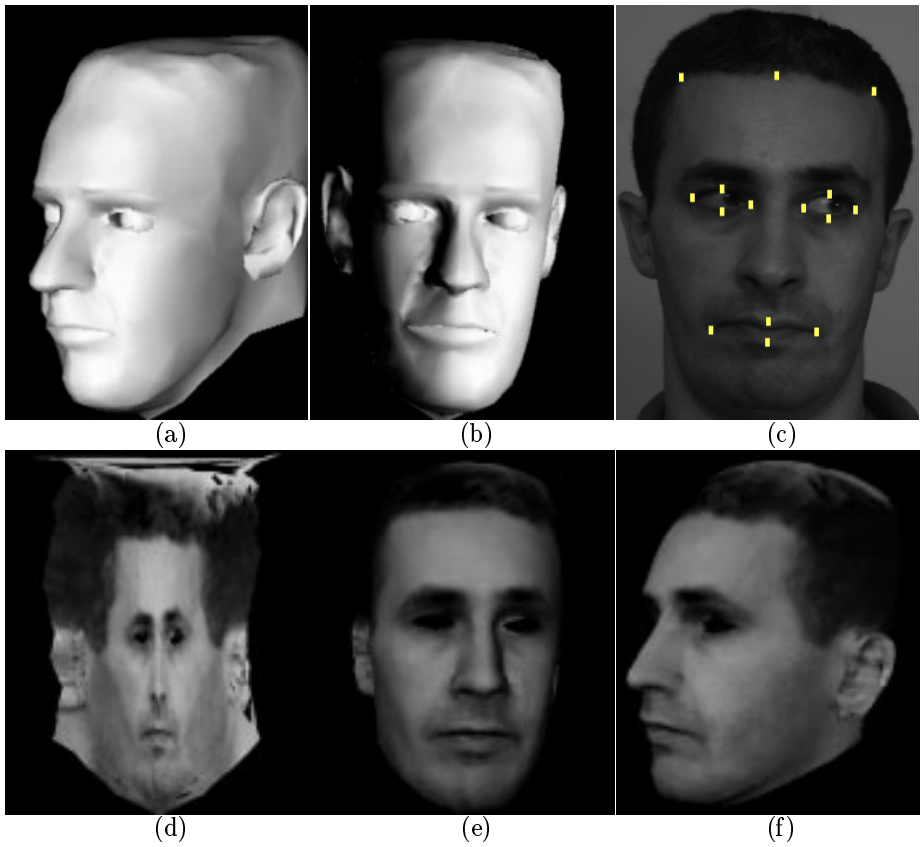


Fig. 6. Recovered head: (a,b) Shaded model of the head of Figure 2. (c) 2–D feature points used to constrain the reconstruction. (d) Computed texture map. (e,f) Texture mapped representations of the head.

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

$$P_i = P_i^0 + l_1^i \delta_{j1} + l_2^i \delta_{j2} + l_3^i \delta_{j3} \quad , \tag{9}$$

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 state vector to be optimized 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 a precomputed set of increasingly refined control triangulations, we implement a hierarchical fitting scheme that has proved very useful when dealing with noisy data.

B. 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 δ_j of the control vertices located where there is little or no data are consistent with their neighbors. As discussed in Section III-C, because our control triangulation is discrete, we can again treat its facets as C^0 finite elements and write our stiffness term as

$$E_S = \Delta_x^t K \Delta_x + \Delta_y^t K \Delta_y + \Delta_z^t K \Delta_z \tag{10}$$

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 objective function we actually optimize becomes

$$E = E_A + \lambda_S E_S \quad , \tag{11}$$

where λ_S is a small positive constant and E_A is the sum of the square distances to the attractors of Equation 8.

C. Weighing the Observations

We recompute the facet closest to each attractor at each stage of the hierarchical fitting scheme of Section IV-A, that is each time we introduce a new control triangulation.

Because the stereo data is noisy and may contain errors, we use again the Iterative Reweighted Least Squares

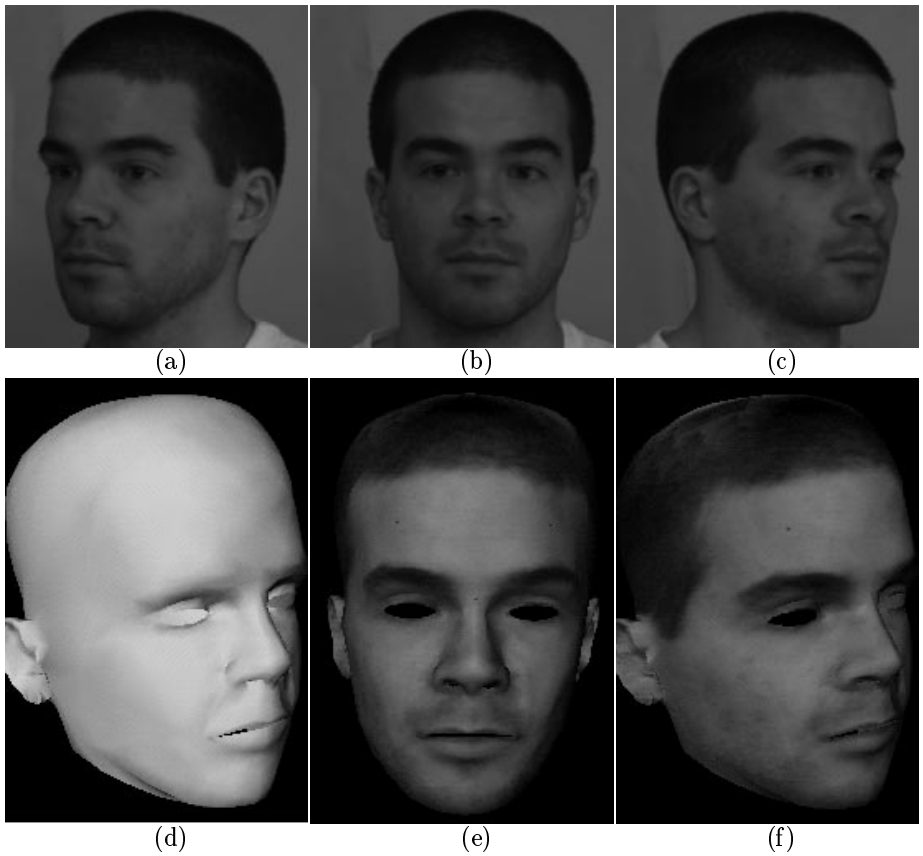


Fig. 7. Another video sequence: (a,b,c) Three images of a sequence of 9. (d) Shaded model of the head (e,f) Texture mapped representation.

technique introduced in section III-C. Each time we recompute the attachments, we also recompute the weight w_i associated with attractor i and take it to be inversely proportional to the initial distance d_i of the data point to the surface triangulation using the same formula as before. That is, we take w_i to be

$$w_i = exp(\frac{-d_i^a}{\overline{d^a}}) \tag{12}$$

where $\overline{d^a}$ is the median value of the d_i.

D. Shape and Texture

This fitting procedure has been used to model the face of the heads shown in Figures 6(a,b), 7(d) and 8(d) using the control triangulation of Figure 1(d). In both cases, to ensure that some of the key elements of the face—corners of the eyes, mouth and hairline—project at the right places we have manually supplied the location of the projection in one image of a few feature points such as the ones shown in Figure 6(c). We have then added a term to the objective function of Equation 11 that forces the projection of the corresponding vertices of the generic mask model to be close to those locations [21]. Note that the five manually supplied points used to initialize the bundle-adjustment procedure of Section III-B form a subset of these features points. To produce these face models, the manual intervention required therefore reduces to supplying these few points by clicking on their approximate locations in one single image, which can be done quickly.

The shape of the top of the head has been recovered by semi-automatically delineating in each image of the video sequence the boundary between the hair and the background and treating it as a silhouette that constrains the shape [21].

Given the final model of the head, we can compute a texture map by:

1. Generating a cylindrical projection of the head model.
2. For each projected point, finding the images in which it is visible and averaging the corresponding gray-levels.

This yields texture maps such as the one shown in Figure 6(d) and allows textured reconstruction such as the ones of Figures 6(e,f) and 7(e,f). If we use color images instead of black and white ones, we can repeat the process on each of the color bands and produce color models such as the one of Figure 8.

V. Conclusion

We have presented a technique that allows us to fit a complex animation model to uncalibrated video sequences with very limited manual intervention. As a result, these models can be produced cheaply and fast using an entirely passive sensor.

This has direct applications in the field of video com-

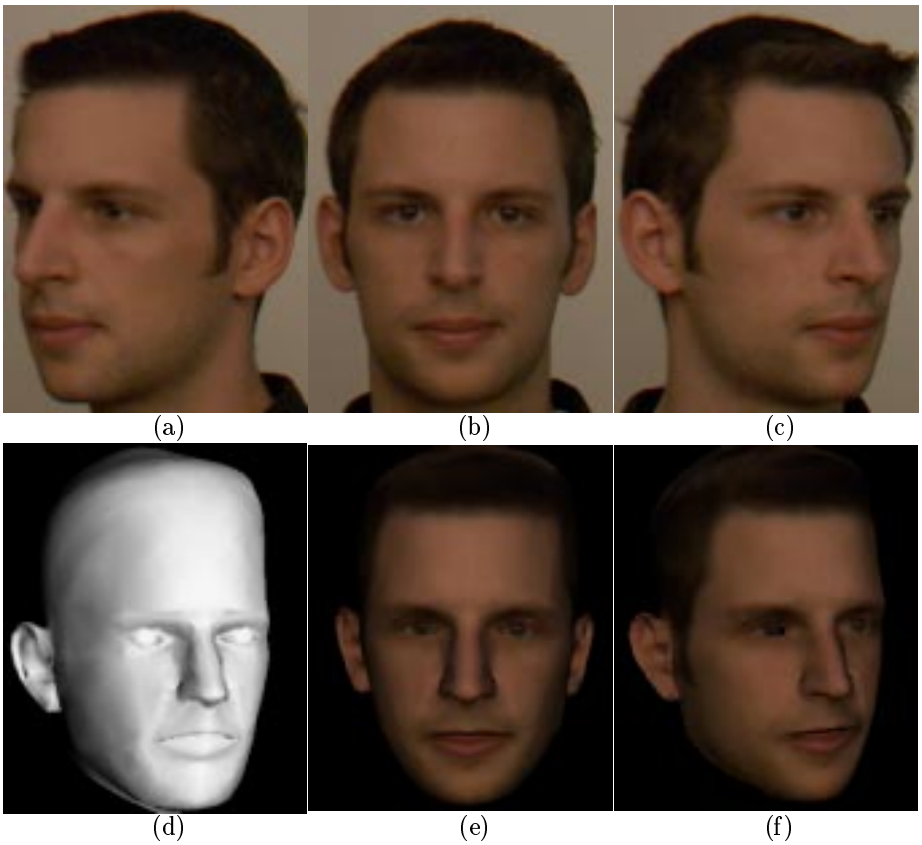


Fig. 8. A color video sequence: (a,b,c) Three color images (d) Shaded model of the head (e,f) Texture mapped representation.

munication and entertainment and will allow the the fast generation of realistic avatars from widely available data. 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. Christian Heipke for the many illuminating discussions about bundle-adjustment we have had when we began the work reported here. We also wish to thank Prof. Nadia Magnenat Thalmann and Prof. Daniel Thalmann for having made their facial animation model available to us. We are also indebted to Mirco Sartor who spent many hours implementing the algorithm that generates the texture maps shown in this article.

References

[1] 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*, Seattle, WA, June 1994, pp. 208–213.

[2] 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.

[3] 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.

[4] L. Tang and T.S. Huang, "Analysis-based facial expression synthesis," *ICIP-III*, vol. 94, pp. 98–102.

[5] 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*, 1996, pp. 231–238.

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

[7] W.S. Lee and N. Magnenat Thalmann, "From Real Faces To Virtual Faces: Problems and Solutions," in *3IA*, Limoges, France, 1998.

[8] P.E. Gill, W. Murray, and M.H. Wright, *Practical Optimization*, Academic Press, London a.o., 1981.

[9] P. Fua and C. Brechbühler, "Imposing Hard Constraints on Deformable Models Through Optimization in Orthogonal Subspaces," *Computer Vision and Image Understanding*, vol. 24, no. 1, pp. 19–35, February 1997.

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

[11] A.Gruen and H. Beyer, "System Calibration through Self-Calibration," in *Calibration and Orientation of Cameras in Computer Vision*, Washington D.C., August 1992.

[12] O.D. Faugeras, Q.-T. Luong, and S.J. Maybank, "Camera self-calibration: theory and experiments," in *European Conference on Computer Vision*, Santa-Margerita, Italy, 1992, pp. 321–334.

[13] B. Triggs, "Autocalibration and the Absolute Quadric," in *Conference on Computer Vision and Pattern Recognition*, 1997, pp. 609–614.

[14] 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.

[15] Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong, "A Robust

Technique for Matching two Uncalibrated Images through the Recovery of the Unknown Epipolar Geometry," *Artificial Intelligence*, vol. 78, pp. 87–119, 1995.

[16] A.W. Fitzgibbon and A. Zisserman, "Automatic Camera Recovery for Closed or Open Image Sequences," in *European Conference on Computer Vision*, Freiburg, Germany, June 1998, pp. 311–326.

[17] P. Fua, "A Parallel Stereo Algorithm that Produces Dense Depth Maps and Preserves Image Features," *Machine Vision and Applications*, vol. 6, no. 1, pp. 35–49, Winter 1993.

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

[20] P. Fua, "From Multiple Stereo Views to Multiple 3–D Surfaces," *International Journal of Computer Vision*, vol. 24, no. 1, pp. 19–35, 1997.

[21] P. Fua and C. Miccio, "From Regular Images to Animated Heads: A Least Squares Approach," in *European Conference on Computer Vision*, Freiburg, Germany, June 1998.