Contour-based 3D Face Modeling from a Monocular Video

Himaanshu Gupta[*]
ObjectVideo
Reston VA 20191
hgupta@objectvideo.com

Amit K. RoyChowdhury[*]
Dept. of Electrical Engineering
University of California
Riverside CA 92521
amitrc@ee.ucr.edu

Rama Chellappa
Dept. of Electrical and Comp. Engineering
University of Maryland
College Park MD 20740
rama@cfar.umd.edu

Abstract

In this paper, we present a novel 3D face modeling approach from a monocular video captured using a conventional camera. The proposed algorithm relies on matching a generic 3D face model to the outer contours of the face to be modeled and a few of its internal features. At the first stage of the method, we estimate the head pose by comparing the edges extracted from video frames, with the contours extracted from a generic face model. Next, the generic face model is adapted to the actual 3D face by global and local deformations. An affine model is used for global deformation. The 3D model is locally deformed by computing the optimal perturbations of a sparse set of control points using a stochastic search optimization method. The deformations are integrated over a set of poses in the video sequence, leading to an accurate 3D model.

1 Introduction

Constructing 3D face models from a video is one of the challenging problems in computer vision. Successful solution of this problem has applications in multimedia, computer graphics, and face recognition. By incorporating 3D models into face recognition systems, the problems arising due to pose, illumination, and expression variations can be effectively addressed. Most current commercial systems address special cases of the 3D face reconstruction problem that are well-constrained by additional information in the form of depth estimates available from multiple cameras, projecting laser or other patterns on the face, decorating the face with special textures to make inter-frame correspondences simple, or using structured light to reveal the contours of the face. All these constraints and the special hardware needed reduce the operational flexility of the system.

One of the common approaches to solve the problem of 3D reconstruction from a monocular video is Structure from Motion (SfM). Numerous SfM algorithms [5] that can reconstruct a 3D scene from two or more images exist in the literature. The basic idea behind most SfM algorithms is to recover the structure of 3D points on a rigid object from 2D point correspondences across images, or recover a dense depth map from optical flow.

[*]The authors were at the Center for Automation Research, University of Maryland, College Park, MD when this work was performed.

These methods have been adapted for face modeling by a number of authors [4] [10] [9]. Romdhani, Blanz and Vetter [8] came up with an impressive appearance-based approach where they showed that it is possible to recover the shape and texture parameters of a 3D morphable model from a single image. Shape from Contours [1] is another promising approach for 3D reconstruction. One of the strongest cues for the 3D information contained in a 2D image is the outline of an object in the image. The occluding contour (extreme boundary) in a 2D image directly reflects the 3D shape. Shape from Silhouette techniques have been used to reconstruct 3D shapes from multiple silhouette images of an object without assuming any previous knowledge of the object to be reconstructed [6]. However, it is impossible to recover the concavities in the shape of the object from the silhouettes or contours. But if we assume prior knowledge of the object being reconstructed, and use a generic model, contour information can be exploited as an important constraint for the exact shape of the object. Moghaddam et al [7] have developed a system to recover the 3D shape of a human face from a sequence of silhouette images. They use a downhill simplex method to estimate the model parameters, which are the coefficients of the eigenhead basis functions.

We propose a contour-based 3D face modeling method capable of working in an unconstrained environment, without any special illumination or sensors. Our system reconstructs the 3D face model from a monocular video captured using a conventional camera. We begin by estimating the pose of the head in each frame. A generic 3D face model (which is used to initialize the reconstruction algorithm) is globally and locally deformed to adapt itself to the outer contours of the face to be modeled and a few of its internal features. The process is integrated over an entire video sequence. Figure 1 shows a block diagram of our proposed algorithm. A detailed description of each block is presented in the paper. This paper is organized as follows. In Section 2, we describe our contour-based pose estimation algorithm. The proposed algorithm for contour-based 3D face reconstruction is described in Section 3. 3D face modeling results obtained from a real test video sequence are presented in Section 4. Conclusions drawn from our work are discussed in Section 5.

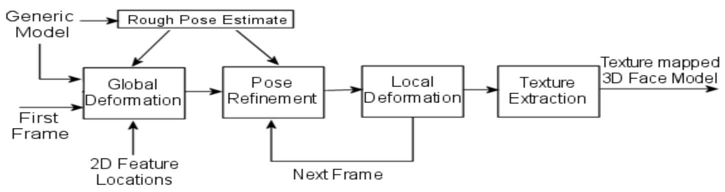

Figure 1: 3D face modeling algorithm

2 Pose Estimation

A contour-based pose estimation method is proposed to estimate the head pose for a video frame. To estimate the head pose without any prior knowledge about the 3D structure of the face, we use a generic 3D face model. Human shape variability is highly limited by both genetic and environmental constraints, and is characterized by a high degree of symmetry and approximate invariance of body lengths and ratios. Since the facial anthropometric measurements of the generic face are close to average, the algorithm can estimate the pose robustly, unless the person whose head pose is being estimated has a hugely deformed face. The problem of pose estimation along the azimuth angle is most commonly encountered in video sequences, and we limit ourselves to this case in the paper. The algorithm described in this section can be easily extended to incorporate more complex head motion (including roll and elevation). The addition of each degree of freedom of the head motion causes the search space for the face pose to grow exponentially, thus slowing down the 3D reconstruction algorithm at the rough pose estimation and pose refinement stages.

The frames extracted from the video are sub-sampled such that successive frames have a distinct pose variation. A simple image-difference method is used to detect the background pixels in the video. All edges in the background are removed to make sure they do not adversely affect the pose estimation algorithm. Since we assume that the head motion is only along the azimuth angle, the top and bottom pixels of the face region are marked out in the first frame, and are assumed to be constant across all video frames. All the edge maps extracted from the average 3D model are resized according to this scale. Also, only the edges in the region between the marked out top and bottom pixels of the face region are retained, thus removing the hair and shoulder edges. The pose estimation algorithm requires the coordinates of the nose tip in the image, because the edge maps obtained from the 2D projection of the 3D model and from the video frames are aligned at the nose tip. We use the Kanade-Lucas tracker [12] to automatically track the nose tip across multiple frames. In case of inaccurate tracking, the nose tip coordinates are manually refined.

For pose estimation, an average human texture is mapped onto the generic 3D face model. We obtained an average 3D face shape and texture data from [8]. The texture mapped generic face model is rotated along the azimuth angle, and edges are extracted using the Canny edge detector [3]. The projection of the average 3D model also has edges which result from the boundaries of the 3D model (e.g. top of forehead, bottom of neck). These edges are not the result of natural contours of the human face, and therefore should be removed. This is done using a modified average texture, the details of which are not presented here due to lack of space. The edge maps are computed for 3D model rotation along the azimuth angle from -90° to $+90^\circ$ in increments of 5°. These edge maps are computed only once, and stored offline in an image array to make the procedure fast.

To estimate the head pose in a given video frame, we extract the edges of the image using the Canny edge detector. Each of the scaled 3D model edge maps is compared to this frame edge map to determine which pose results in the best overlap of the edge maps. To compute the disparity between these edge maps, the Euclidean Distance Transform (DT) of the current video frame edge map is computed. For each pixel in the binary edge map, the distance transform assigns a number that is the distance between that pixel and the nearest nonzero pixel of the edge map. Figure 2 shows the binary edge map of a video frame, and the corresponding distance transform.

Each of the 3D model edge maps is aligned at the nose tip in the video frame, and the value of the cost function, F, is computed. The cost function, F, which measures the disparity between the 3D model edge map and the edges of the current video frame is of the form:

$$F = \frac{\sum_{(i,j)\in A_{EM}} DT(i,j)}{N} \tag{1}$$

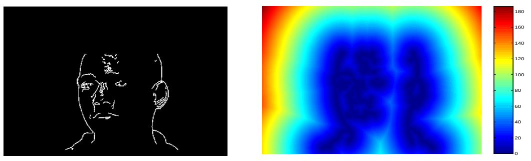

Figure 2: Left: Edge Map. Right: Distance Transform

where $A_{EM} \triangleq \{(i,j) : EM(i,j) = 1\}$ and N is the cardinality of set A_{EM} (total number of nonzero pixels in the 3D model edge map EM). F is the average distance transform value at the nonzero pixels of the binary 3D model edge map. The pose for which the corresponding 3D model edge map results in the lowest value of F is the estimated head pose for the current video frame. Figure 3 shows the head pose estimation results for a few video frames of a subject.

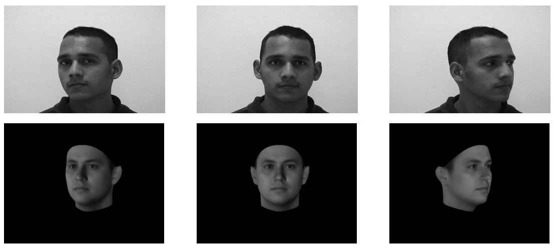

Figure 3: Top Row: Video Frames. Bottom Row: Generic model at estimated head pose

3 3D Face Model Reconstruction

In this section, we describe the contour-based algorithm for 3D face reconstruction from a monocular video. A generic 3D face model is assumed to be the initial estimate of the true 3D face model. The generic face model is globally and locally deformed to adapt itself to the actual 3D face of the person.

3.1 Registration and Global Deformation

Once the pose is estimated using the method described in Section 2, the next step is to perform a global deformation and register the 3D model to the 2D image. We use a scaled orthographic projection model for the camera, and an affine deformation model for the global deformation of the generic face model. The coordinates of four feature points (left eye, right eye, nose tip, and mouth center) are used to determine a solution for the affine parameters. Since these feature point locations need to be known for just the first frame, currently we mark these manually. The 3D coordinates of the corresponding feature points on the generic face model are available beforehand.

The generic 3D model is globally deformed only for the first frame. The following affine model is used for global deformation:

$$\begin{bmatrix} X_{\text{gb}} \\ Y_{\text{gb}} \\ Z_{\text{gb}} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & \frac{a_{11}+a_{22}}{2} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ 0 \end{bmatrix} \tag{2}$$

where (X, Y, Z) are the 3D coordinates of the vertices of the generic model, and subscript "gb" denotes global deformation. The affine model appropriately stretches/shrinks the 3D model along the the X, and Y axes and also takes into account the shearing in the X-Y plane. Considering the basic symmetry of human faces, the affine parameters contributing to shearing (a_{12}, a_{21}) are very small, and can often be neglected. Since we use an orthographic projection model, we can not have an independent affine deformation parameter for the Z-coordinate. The affine deformation parameters are obtained by minimizing the reprojection error of the 3D feature points on the rotated deformed generic model, and their corresponding 2D locations in the current frame. The 2D projection (x_f, y_f) of the 3D feature points (X_f, Y_f, Z_f) on the deformed generic face model is given by,

$$\begin{bmatrix} x_f \\ y_f \end{bmatrix} = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \end{bmatrix}}_{R_{12}} \begin{bmatrix} a_{11}X_f + a_{12}Y_f + b_1 \\ a_{21}X_f + a_{22}Y_f + b_2 \\ (\frac{a_{11}+a_{22}}{2})Z_f \end{bmatrix} \tag{3}$$

where R_{12} is the matrix containing the top two rows of the rotation matrix corresponding to the estimated head pose for the first frame. Using the coordinates of the four feature points, (3) can be reformulated into a linear system of equations. The affine deformation parameters $\mathbf{P} = [a_{11}, a_{12}, a_{21}, a_{22}, b_1, b_2]^T$ can be determined by obtaining a Least-Squares (LS) solution of the system of equations. The generic mesh is globally deformed according to these parameters. This process ensures that the 3D face model matches the approximate shape of the face and the significant internal features are properly aligned.

3.2 Local Deformation

To adapt the 3D model to a particular individual's face more accurately from the video sequence, we introduce local deformations in the globally deformed model. The algorithm for local deformation of the face model is described in this section.

3.2.1 Control Points

The globally deformed dense face mesh is sampled at a small number of points to obtain a sparse mesh shown in Figure 4. Each of the vertices of this sparse mesh is a control point in the optimization procedure. Each control point is imparted a random perturbation in the X, Y, and Z direction. The perturbation for each of the vertices of the dense face mesh is computed from the random perturbations of control points using triangle-based linear interpolation. The computed perturbations are imparted to the vertices of the face mesh to obtain a locally-deformed mesh. The outer contours obtained from the locally-deformed face mesh are compared with the edges from the current video frame. The optimum local perturbations of the control points are determined using a Direct Random Search method.

3.2.2 Direct Random Search

Direct Random Search methods [11] are based on exploring the domain D in a random manner to find a point that minimizes the cost function L. They are "direct" in the sense that the algorithms use minimal information about the cost function $L = L(\theta)$. The minimal information is essentially only the input-output data of the following form: input $= \theta$, output $= L(\theta)$. In the Global Random Search method, we repeatedly sample over D

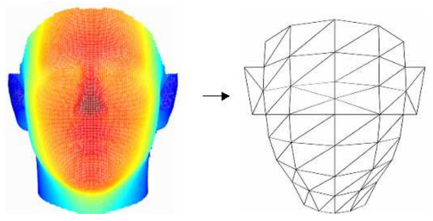
Figure 4: Sparse mesh of control points

such that the current sampling for θ does not take into account the previous samples. The domain D is a hypercube (a p-fold Cartesian product of intervals on the real line, where p is the problem dimension), and we use uniformly distributed samples. There exists a convergence proof [11] which shows that the two-step (after initialization) algorithm described below converges almost surely to the global minimum θ^*.

Global Random Search Algorithm:
Step 0 (initialization): Generate an initial value of θ, say $\hat{\theta}_0 \in D$, according to the uniform probability distribution on the domain D. Calculate $L(\hat{\theta}_0)$. Set $k = 0$.
Step 1: Generate a new independent value of $\theta \in D$, say $\theta_{new}(k+1)$, according to the uniform probability distribution. If $L(\theta_{new}(k+1)) < L(\hat{\theta}_k)$, set $\hat{\theta}_{k+1} = \theta_{new}(k+1)$. Else take $\hat{\theta}_{k+1} = \hat{\theta}_k$.
Step 2: Stop if the maximum number of L evaluations has been reached; else return to Step 1 with $k = k+1$.

In our application of the Global Random Search algorithm to achieve the optimum local deformation of the face model, the cost function L is the disparity between the outer contours obtained from the 3D face model after applying the local deformation (θ) and the edges from the current video frame. The stochastic optimization algorithm determines the value of θ that minimizes the cost function L. The parameter vector θ whose optimum value is being sought is of the form:

$$\theta = [\Delta X_c \ \Delta Y_c \ \Delta Z_c]_{P\times 3} \tag{4}$$

where ΔX_c, ΔY_c, and ΔZ_c are the perturbations in the X, Y, and Z directions of the P control points. The perturbations for all vertices of the dense face mesh, $[\Delta X_{\text{vt}} \ \Delta Y_{\text{vt}} \ \Delta Z_{\text{vt}}]$, are computed from the perturbations of these control points using a triangle-based linear interpolation method. The coordinates of the vertices of the locally deformed face model are given by,

$$\begin{bmatrix} X_{\text{lc}} \\ Y_{\text{lc}} \\ Z_{\text{lc}} \end{bmatrix} = \begin{bmatrix} X_{\text{gb}} \\ Y_{\text{gb}} \\ Z_{\text{gb}} \end{bmatrix} + \begin{bmatrix} \Delta X_{\text{vt}} \\ \Delta Y_{\text{vt}} \\ \Delta Z_{\text{vt}} \end{bmatrix} \tag{5}$$

where the subscript “lc” denotes local deformation. Let EM_θ be the binary edge map (outer contours) of the 2D projection of the 3D model after applying local deformation θ. The unwanted edges due to the boundaries of the 3D model are removed by the method described earlier in Section 2. Let DT be the distance transform of the edge map of the current video frame. The cost function, L, is of the form:

$$L(\theta) = \frac{\sum_{(i,j)\in A_{EM_\theta}} DT(i,j)}{N} \tag{6}$$

where $A_{EM_\theta} \triangleq \{(i,j) : EM_\theta(i,j) = 1\}$ and N is the cardinality of set A_{EM_θ}. The structure of this cost function is similar to the one used in (1). In Section 2, we compared the edge maps to determine the head pose, but here we compare them to determine the optimal local deformation for the globally deformed generic face model.

3.2.3 Multi-Resolution Search

The random perturbations are applied to the face mesh at two different resolutions. Initially, M iterations of large random perturbation are performed to scan a large search area at a coarse resolution. The final estimate at the end of M coarse-level iterations is denoted by $\hat{\theta}_M^c$. Once we have a coarsely deformed mesh which is close to the global minimum, we use this coarsely deformed mesh as the starting estimate for the fine resolution iterations, i.e. $\hat{\theta}_M^c = \hat{\theta}_0^f$. N small random perturbations are applied to the coarsely deformed mesh to get even closer to the actual solution. The final estimate at the end of N fine-level iterations is denoted by $\hat{\theta}_N^f$. This coarse to fine resolution search helps the algorithm converge to the solution much faster, compared to a search strategy where we sample the search space at a fixed fine resolution. As we increase the number of iterations performed for the coarse and fine resolution searches, the performance of the algorithm gets better, but, at the cost of computational time. We observed that the algorithm converges to a good solution in 300 iterations of large (coarse) random perturbations, and 100 iterations of small (fine) random perturbations.

As a final step to fine tune the control point perturbations estimate, we exhaustively search for the optimal perturbation $\hat{\theta}_N^{sf}(i)$ for each control point i, individually, in a small neighborhood of the solution from fine resolution iterations $\hat{\theta}_N^f(i)$, while fixing the perturbations of all other control points to the value determined by $\hat{\theta}_N^f$. The perturbation that results in the minimum value of the cost function L, is chosen to be the optimal perturbation $\hat{\theta}_N^{sf}(i)$ for that particular control point. The superscript "*sf*" denotes super-fine estimate. Ideally, we would like to perform this exhaustive search in a combinatorial manner (instead of individual) over the entire search space, but the computational complexity for this strategy would be extremely prohibitive.

3.2.4 Constraints

We impose a few constraints on the values of perturbations that are imparted to each of the control points. Without any constraints, the algorithm will also search in the domain of unrealistic faces, thus unnecessarily slowing down the algorithm. The following constraints are imposed on the perturbations of the control points:

- Symmetry along the vertical axis passing through nose tip, based on the fact that most human faces are symmetric about this axis.
- The maximum possible perturbation for the control points is defined.
- The perturbation of a few control points is dependent on the perturbations of their neighboring control points.

- Only control points whose movement might alter the contours of the 3D model (and hence change the cost function) are perturbed.

3.3 Pose Refinement and Model Adaptation across time

Once we have an adapted 3D model from a particular frame, we refine the rough pose estimate (obtained using a generic model) of the next frame to obtain a more accurate head pose estimate. The method we use for this pose refinement is similar to the method described in Section 2, except that we extract the contours to be compared to the edges of the current frame from the adapted 3D model upto that stage, instead of the generic face model. For the first video frame, the adapted 3D model used for pose refinement is just the globally deformed generic model. But, for later frames, we use the globally and locally adapted 3D model for pose refinement. This pose refinement step is critical to the shape estimation procedure because our contour-based algorithm is very sensitive to the head pose estimate. If our pose estimate is not accurate, the 3D model (at the wrong pose) will adapt itself in an inappropriate manner so that its contours conform with the edges extracted from the video frame. To refine the head pose estimate, we obtain the contours from the adapted 3D face model rotated about azimuth angles in a small neighborhood of the rough pose estimate (obtained using the method in Section 2). The angle which results in 3D model contours closest to the edges extracted from the current frame is chosen as the refined pose estimate. The similarity criterion used to compare the edge maps is the Distance Transform based measure described in Equation (1).

Once we have the refined pose estimate and the adapted (global and local) 3D face model from the first video frame, we recursively apply the algorithm for pose refinement and local deformation described earlier to all the subsequent frames to improve the quality of the reconstructed 3D face model. Integrating the algorithm across several frames makes the system robust to any noise in the contours extracted from a particular frame. The adapted 3D model from the previous frame is used as the starting estimate for the next frame. As we get more frames with varying head poses, we also get valuable cues to model certain aspects of the face more accurately (e.g. the side view models the structure of the nose better than the front view).

3.4 Texture Extraction

Once the pose refinement and the 3D model adaptation has been done for all the frames, as a final step, we need to extract the texture of the person's face and map it onto the adapted 3D model for visualization. A single frame is used for texture extraction, because we found from our experiments that texture sampling across multiple frames smears the texture slightly. The smearing occurs because of the fact that the registration (using estimated pose) across multiple frames is not exact. Now the question arises which frame to choose for texture extraction? Based on studies that face recognition systems perform best on 3/4 profile view [2], we believe that this view is most representative of the person. Hence, we choose the frame closest to 3/4 profile view ($\pm 45^\circ$) for texture extraction. In a 3/4 profile view, part of the face would be occluded for one half of the face. To get the texture for this half of face (part of which is occluded), we assume symmetry of face texture about the vertical axis passing through the nose tip.

4 Experimental Results

In this section, we present experimental results of our proposed contour-based 3D face modeling algorithm. The current implementation of the algorithm in MATLAB took approximately 1 hour to reconstruct the 3D face model using 9 video frames of size 720x480 on a 1.66 GHz Pentium 4 machine. No effort has been made yet to optimize the algorithm. Figure 5 shows a few video frames of two subjects whose face modeling results have been presented. Figure 6 shows the plot of perturbation errors (value of cost function L) at each stage of the multi-resolution search across all frames of the video sequence for subject A. It can be observed from the plot that, within each frame, the minimum perturbation error decreases as the search resolution gets finer. Figure 7 shows the reconstructed texture mapped 3D face model of the subjects from different viewpoints. Extensive experimentation on a number of subjects has been conducted, with good results.

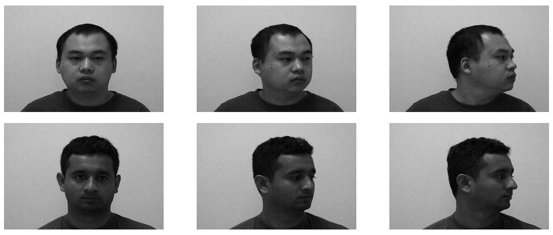

Figure 5: Video Frames. Top Row: Subject A. Bottom Row: Subject B.

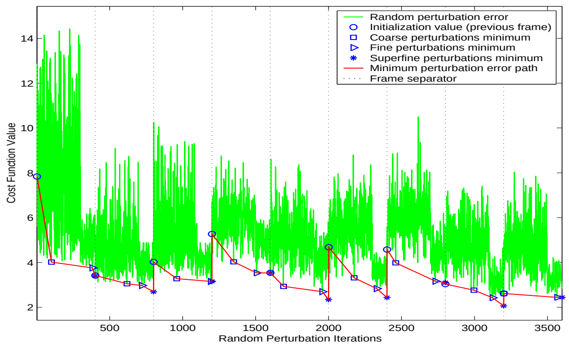

Figure 6: Perturbation errors at each stage of the multi-resolution search for subject A

5 Conclusions

In this paper, we have addressed the problem of 3D face modeling from a monocular video. Since our algorithm for pose estimation and 3D face reconstruction relies solely on contours, we do not require knowledge of rendering parameters (e.g light direction, intensity, etc.) which might otherwise be needed. Using contours separates the geometric subtleties of the human head from the variations in shading and texture. Hence, we believe our system is much more robust to noise and illumination changes compared

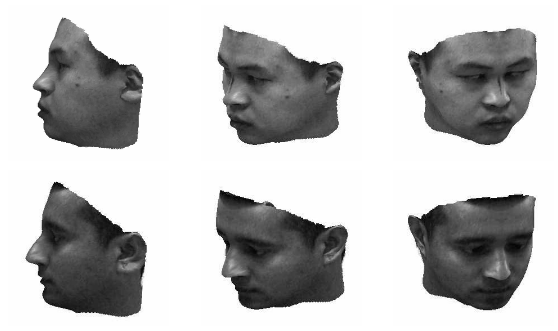

Figure 7: Reconstructed 3D face model. Top Row: Subject A. Bottom Row: Subject B.

to most structure from motion algorithms that rely on finding accurate point correspondences across frames. Our algorithm for 3D face modeling can be applied to build face recognition systems invariant to pose changes and significantly robust to changes in illumination.

References

[1] M. Brady and A. Yuille. An extremum principle for shape from contour. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3):288–301, May 1984.

[2] V. Bruce, T. Valentine, and A. Baddeley. The basis of the 3/4 view advantage in face recognition. *Applied Cognitive Psychology*, 1:109–120, 1987.

[3] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8:679–698, November 1986.

[4] P. Fua. Regularized bundle-adjustment to model heads from image sequences without calibration data. *International Journal of Computer Vision*, 38(2):153–171, July 2000.

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[6] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-based visual hulls. In *Computer Graphics Proceedings, Siggraph 2000, New Orleans, LA*, pages 369–374, July 2000.

[7] B. Moghaddam, J. Lee, H. Pfister, and R. Machiraju. Model-based 3-D face capture with shape-from-silhouettes. In *IEEE International Workshop on Analysis and Modeling of Faces and Gestures, Nice, France*, pages 20–27, October 2003.

[8] S. Romdhani, V. Blanz, and T. Vetter. Face identification by fitting a 3D morphable model using linear shape and texture error functions. In *European Conference on Computer Vision, Copenhagen, Denmark*, pages 3–19, May 2002.

[9] A. Roy Chowdhury and R. Chellappa. Face reconstruction from video using uncertainty analysis and a generic model. *Computer Vision and Image Understanding*, 91(1-2):188–213, July-August 2003.

[10] Y. Shan, Z. Liu, and Z. Zhang. Model-based bundle adjustment with application to face modeling. In *International Conference on Computer Vision, Vancouver, BC*, pages 644–651, July 2001.

[11] J. Spall. *Introduction to Stochastic Search and Optimization*. Wiley, 2000.

[12] C. Tomasi and J. Shi. Good features to track. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recogition, Jerusalem, Israel*, pages 593–600, October 1994.