ACADEMIC PRESS

Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Computer Vision and Image Understanding 91 (2003) 188–213

Computer Vision and Image Understanding

www.elsevier.com/locate/cviu

Face reconstruction from monocular video using uncertainty analysis and a generic model☆

Amit K. Roy Chowdhury and Rama Chellappa*

Department of Electrical and Computer Engineering and Center for Automation Research, University of Maryland, College Park, MD 20742, USA

Abstract

Reconstructing a 3D model of a human face from a monocular video sequence is an important problem in computer vision, with applications to recognition, surveillance, multimedia, etc. However, the quality of 3D reconstructions using structure from motion (SfM) algorithms is often not satisfactory. One of the reasons is the poor quality of the input video data. Hence, it is important that 3D face reconstruction algorithms take into account the statistics representing the quality of the video. Also, because of the structural similarities of most faces, it is natural that the performance of these algorithms can be improved by using a generic model of a face. Most of the existing work using this approach initializes the reconstruction algorithm with this generic model. The problem with this approach is that the algorithm can converge to a solution very close to this initial value, resulting in a reconstruction which resembles the generic model rather than the particular face in the video which needs to be modeled. In this paper, we propose a method of 3D reconstruction of a human face from video in which the 3D reconstruction algorithm and the generic model are handled separately. We show that it is possible to obtain a reasonably good 3D SfM estimate purely from the video sequence, provided the quality of the input video is statistically assessed and incorporated into the algorithm. The final 3D model is obtained after combining the SfM estimate and the generic model using an energy function that corrects for the errors in the estimate by comparing the local regions in the two models. The main advantage of our algorithm over others is that it is able to retain the specific features of the face in the video sequence even when these features are different from those of the generic model and it does so even as the quality of the input video varies. The evolution of the 3D model through the various stages of the algorithm and an analysis of its accuracy are presented.

☆ This work was partially supported by NSF Grant 086075 and DARPA/ONR Grant N00014-00-1-0908.

* Corresponding author. Fax: 1-301-314-9115.
E-mail addresses: amitrc@cfar.umd.edu (A.K. Roy Chowdhury), rama@cfar.umd.edu (R. Chellappa).
URLs: http://www.cfar.umd.edu/~amitrc, http://www.cfar.umd.edu/~rama.

1077-3142/$ - see front matter © 2003 Published by Elsevier Inc.
doi:10.1016/S1077-3142(03)00079-1

Keywords: Structure from motion; Face modeling; Uncertainty analysis; Stochastic approximation; Generic model; Energy function minimization

1. Introduction

Reconstructing 3D models from video sequences is an important problem in computer vision with applications to recognition, medical imaging, video communications, etc. Though numerous algorithms exist which can reconstruct a 3D scene from two or more images using structure from motion (SfM) [1,2], the quality of such reconstructions is often unsatisfactory. The main reason for this is the poor quality of the input images and a lack of robustness in the reconstruction algorithms to deal with it [3]. One particularly interesting application of 3D reconstruction from 2D images is in the area of modeling a human face from video. Successful solution of this problem has applications in multimedia, computer graphics, and surveillance. In multimedia, 3D face models can be used in video conferencing applications for efficient transmission. In computer graphics applications, 3D face models form the basic building block upon which facial movements and expressions can be added. Being able to build these models automatically from video data would greatly simplify such animation tasks where models are now painstakingly built with significant human intervention. In surveillance applications, 3D models can be used for recognition across wide changes in viewing angles.

Various researchers have addressed the issue of 3D face modeling. In [4], the authors used an extended Kalman filter to recover the 3D structure of a face which was then used for tracking. A method for recovering non-rigid 3D shapes as a linear combination of a set of basis shapes was proposed in [5]. A factorization based method for recovering non-rigid 3D structure and motion from video was presented in [6]. In [7], the author proposes a method for self-calibration in the presence of varying internal camera parameters and reconstructs metric 3D structure. Romdhani et al. [8] have shown that it is possible to recover the shape and texture parameters of a 3D morphable model from a single image. They used their 3D model for identification of faces under different pose and illumination conditions. Our work is along the lines of [9,10], where the authors proposed solving the problem of 3D face modeling using a generic model. Their method of bundle-adjustment works by initializing the reconstruction algorithm with this generic model. The difficulty with this approach is that the algorithm often converges to a solution very near this initial value, resulting in a reconstruction which has the characteristics of the generic model, rather than that of the particular face in the video which needs to be modeled. This method may give very good results when the generic model has significant similarities with the particular face being reconstructed. However, if the features of the generic model are different from those of the face being reconstructed, the solution obtained using this approach may be unsatisfactory.

In this paper we deal with face modeling from monocular video, with special emphasis on the incorporation of the generic face model.

Reconstructing from a monocular video. This is particularly important in unregulated surveillance applications where the training data (from which the 3D model needs to be estimated) may contain a few images or a video from one view (e.g., frontal), but the probe may be another view of the person (e.g., profile). Since the motion between pairs of frames in a monocular video is usually small, we adopt the optical flow paradigm for 3D reconstruction [11]. Also, estimating the motion between the pairs of frames accurately may be a challenge in many situations because of differences in the quality of the input video. Our method learns certain statistical parameters of the incoming video data and incorporates them in the algorithm. Quality evaluation is done by estimating the statistical error covariance of the depth estimate analytically from the optical flow equations. The details of the statistical analysis can be found in [12,13].

Avoid biasing the reconstruction with the generic model. Mathematically speaking, the introduction of the generic model is similar to introducing constraints on the solution of the 3D estimation problem. In our method, we introduce the generic model *after* obtaining the estimate using the SfM algorithm. The SfM algorithm reconstructs purely from the video data after computing the optical flow. Instead of directly fusing the depth values in the 3D estimate and the generic model, we propose a cost function which identifies local regions where there are no sharp depth discontinuities, looks for deviations in the trend of the values of the 3D estimate in these regions and then corrects for the errors. The optimization of the cost function is done in a Markov Chain Monte Carlo (MCMC) framework using a Metropolis–Hastings sampler [14]. The advantage of this method is that the particular characteristics of the face that is being modeled are not lost since the SfM algorithm does not incorporate the generic model. However, most errors (especially those with large deviations from the average representation) in the reconstruction are corrected in the energy function minimization process by comparison with the generic model.

The issues of statistical analysis and reconstruction from monocular video have been dealt with in detail in [13], and we will only provide a brief overview of the method here. The main focus of this paper will be the use of the generic model in face reconstruction.

We will now briefly discuss about our method for incorporating a generic model in the 3D face modeling framework. For a detailed review of work on statistical error analysis in 3D modeling from video, the interested reader can refer to our paper [13].

1.1. Incorporating a generic model in an energy function minimization framework

Our 3D reconstruction framework based on the uncertainty calculations provides a depth estimate using the input video data only (explained in detail later). However, localized errors still remain which were not detected using the error correction strategy outlined above. The reason is that while the error covariance calculations can identify and correct for small errors, they are unable to correct the larger errors due to outliers. We use a generic face model in order to overcome such errors. A regularization approach to incorporating the generic model is proposed by imposing

smoothness constraints on the final 3D reconstruction. A pertinent question to ask here is: why do we need the error correction strategy (in the SfM algorithm) *and* the generic model? Is it not sufficient to have a simple multi-frame reconstruction algorithm without the error correction strategies, followed by the generic model to correct for all the errors? The answer is negative, because if we use the generic model to correct for all the errors, we run into the problem of over-smoothing the 3D structure estimate. This is similar to the situation when the generic model is incorporated at the beginning of the SfM algorithm, as explained before. The aim here is to obtain as precise a 3D model as possible from SfM and then use the generic model to correct for the errors which still persist.

The idea of using energy functions (also known as variational methods, regularization theory, and relaxation methods) to impose smoothness constraints has been very influential in computer vision [15]. Regularization theory works by minimizing a functional $E[f(x)]$ with respect to a function $f(x)$. It usually contains one term (a consistency or fidelity term) which ensures that the smoothed solution is close to the data, and a second term (a regularization term) which imposes smoothness constraints on the solution. In most implementations, where the data is specified on a regular lattice, the energy function is discretized as $E[f_i]$.

In our problem, the 3D estimate obtained from the multi-frame reconstruction algorithm needs to be smoothed in local regions where there are errors. These regions are identified with the help of the generic model. After the 3D depth estimate and the generic model have been aligned, the boundaries where there are sharp depth discontinuities are identified from the generic model. Each vertex of the triangular mesh representing the model is assigned a binary variable (defined as a line process, following the terminology of [16]) depending upon whether or not it is part of a depth boundary. Within each region inside the boundaries, the trend in the values of the 3D estimate is considered and any appreciable deviations are smoothed using an energy function minimization process. The energy function consists of two terms which determine the closeness of the final smoothed solution to either the generic model or the 3D depth estimate, and a third term which determines whether or not a particular vertex of the mesh should be smoothed based on the value of the binary variable representing the line process for that vertex. The combinatorial optimization problem is solved using simulated annealing and a Markov Chain Monte Carlo sampling strategy [14].

2. 3D face reconstruction from monocular video using SfM

In this section, we explain the first part of our face reconstruction algorithm, i.e., evaluating the quality of the video data and estimating the 3D structure using SfM.

2.1. Uncertainty analysis of 3D reconstruction

Since the motion between adjacent frames in a video sequence of a face is usually small, we will adopt the optical flow framework for reconstructing the structure [11].

It is assumed that the coordinate frame is attached rigidly to the camera with the origin at the center of perspective projection and the z-axis perpendicular to the image plane. The camera is moving with respect to the face being modeled (which is assumed rigid) with translational velocity $\mathbf{V} = [v_x, v_y, v_z]$ and rotational velocity $\mathbf{\Omega} = [\omega_x, \omega_y, \omega_z]$ (this can be reversed by simply changing the sign of the velocity vector). Using the small-motion approximation to the perspective projection model for motion field analysis, and denoting by $p(x,y)$ and $q(x,y)$, the horizontal and vertical velocity fields of a point (x,y) in the image plane, we can write the equations relating the object motion and scene depth by [11]

$$\begin{aligned} p(x,y) &= (x - fx_f)h(x,y) + \frac{1}{f}xy\omega_x - \left(f + \frac{1}{f}x^2\right)\omega_y + y\omega_z, \\ q(x,y) &= (y - fy_f)h(x,y) + \left(f + \frac{1}{f}y^2\right)\omega_x - \frac{1}{f}xy\omega_y - x\omega_z, \end{aligned} \tag{1}$$

where f is the focal length of the camera, $(x_f, y_f) = (v_x/v_z, v_y/v_z)$ is known as the *focus of expansion* (FOE), and $h(x,y) = v_z/(z(x,y))$ is the scaled inverse scene depth. We will assume that the FOE is known over a few frames of the video sequence. Under the assumption that the motion between adjacent frames in a video is small, we compute the FOE from the first two or three frames and then keep it constant over the next few frames [17]. For N corresponding points, using subscript i to represent the above-defined quantities at the ith point, we define (similar to [17])

$$\begin{aligned} \mathbf{h} &= (h_1, h_2, \ldots, h_N)^{\mathrm{T}}_{N\times 1}, \\ \mathbf{u} &= (p_1, q_1, p_2, q_2, \ldots, p_N, q_N)^{\mathrm{T}}_{2N\times 1}, \\ \mathbf{r}_i &= (x_i y_i, -(1 + x_i^2), y_i)^{\mathrm{T}}_{3\times 1}, \\ \mathbf{s}_i &= (1 + y_i^2, -x_i y_i, -x_i)^{\mathrm{T}}_{3\times 1}, \\ \mathbf{\Omega} &= (w_x, w_y, w_z)^{\mathrm{T}}_{3\times 1}, \\ \mathbf{Q} &= [\, r_1 \quad s_1 \quad r_2 \quad s_2 \quad \ldots \quad r_N \quad s_N \,]^{\mathrm{T}}_{2N\times 3}, \\ \mathbf{P} &= \begin{bmatrix} x_1 - x_f & 0 & \cdots & 0 \\ y_1 - y_f & 0 & \cdots & 0 \\ 0 & x_2 - x_f & \cdots & 0 \\ 0 & y_2 - y_f & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_N - x_f \\ 0 & 0 & \cdots & y_N - y_f \end{bmatrix}_{2N\times N}, \\ \mathbf{A} &= [\mathbf{P}\ \mathbf{Q}]_{2N\times(N+3)}, \\ \mathbf{z} &= \begin{bmatrix} \mathbf{h} \\ \mathbf{\Omega} \end{bmatrix}_{(N+3)\times 1}. \end{aligned} \tag{2}$$

Then (1) can be written as

$$\mathbf{A}\mathbf{z} = \mathbf{u}. \tag{3}$$

Our aim is to compute $\mathbf{z}$ from $\mathbf{u}$ and to obtain a quantitative idea of the accuracy of the 3D reconstruction $\mathbf{z}$ as a function of the uncertainty in the motion estimates $\mathbf{u}$. Let us denote by $\mathbf{R_u}$ the covariance matrix of $\mathbf{u}$ and by C the cost function

$$C = \frac{1}{2}||\mathbf{A}\mathbf{z} - \mathbf{u}||^2 = \frac{1}{2}\sum_{i=1}^{n=2N} C_i^2(u_i, \mathbf{z}). \tag{4}$$

In [12,13], using the implicit function theorem [18], we proved the following result.

Theorem 2.1. *Define*

$$\begin{aligned}
\mathbf{A}_{\bar{i}p} &= [0\cdots 0 - (x_{\bar{i}} - x_f)0\cdots 0 - x_{\bar{i}}y_{\bar{i}}(1 + x_{\bar{i}}^2) - y_{\bar{i}}] \\
&= [-(x_{\bar{i}} - x_f)\mathbf{I}_{\bar{i}}(N)| - \mathbf{r}_{\bar{i}}] = [\mathbf{A}_{\bar{i}ph}|\mathbf{A}_{\bar{i}pm}], \\
\mathbf{A}_{\bar{i}q} &= [0\cdots 0 - (y_{\bar{i}} - y_f)0\cdots 0 - (1 + y_{\bar{i}}^2)x_{\bar{i}}y_{\bar{i}}(N)x_{\bar{i}}] \\
&= [-(y_{\bar{i}} - y_f)\mathbf{I}_{\bar{i}}(N)| - \mathbf{s}_{\bar{i}}] = [\mathbf{A}_{\bar{i}qh}|\mathbf{A}_{\bar{i}qm}],
\end{aligned} \tag{5}$$

where $\bar{i} = \lceil i/2 \rceil$ *is the ceiling of* i ($\bar{i}$ *will then represent the number of feature points* N *and* $i = 1, \ldots, n = 2N$) *and* $\mathbf{I}_n(N)$ *denotes a* 1 *in the* n*th position of the array of length* N *and zeros elsewhere. The subscript* p *in* $\mathbf{A}_{\bar{i}p}$ *and* q *in* $\mathbf{A}_{\bar{i}q}$ *denotes that the elements of the respective vectors are derived from the* p*th and* q*th components of the motion in* (1). *Then*

$$\mathbf{R_z} = \mathbf{H}^{-1}\left(\sum_i \frac{\partial C_i^{\mathrm{T}}}{\partial \mathbf{z}}\frac{\partial C_i}{\partial \mathbf{u}}\mathbf{R_u}\frac{\partial C_i^{\mathrm{T}}}{\partial \mathbf{u}}\frac{\partial C_i}{\partial \mathbf{z}}\right)\mathbf{H}^{-\mathrm{T}} \tag{6}$$

$$= \mathbf{H}^{-1}\left(\sum_{\bar{i}=1}^{N}\left(\mathbf{A}_{\bar{i}p}^{\mathrm{T}}\mathbf{A}_{\bar{i}p}R_{u\bar{i}p} + \mathbf{A}_{\bar{i}q}^{\mathrm{T}}\mathbf{A}_{\bar{i}q}R_{u\bar{i}q}\right)\right)\mathbf{H}^{-\mathrm{T}} \tag{7}$$

and

$$\mathbf{H} = \sum_{\bar{i}=1}^{N}\left(\mathbf{A}_{\bar{i}p}^{\mathrm{T}}\mathbf{A}_{\bar{i}p} + \mathbf{A}_{\bar{i}q}^{\mathrm{T}}\mathbf{A}_{\bar{i}q}\right), \tag{8}$$

where $\mathbf{R_u} = \mathrm{diag}[R_{u1p}, R_{u1q}, \ldots, R_{uNp}, R_{uNq}]$.

Because of the partitioning of $\mathbf{z}$ in (2), we can write

$$\mathbf{R_z} = \begin{bmatrix} \mathbf{R_h} & \mathbf{R_{hm}} \\ \mathbf{R_{hm}^{\mathrm{T}}} & \mathbf{R_m} \end{bmatrix}. \tag{9}$$

We can then show that for N points and M frames, the average distortion in the reconstruction is

$$D_{\mathrm{avg}}(M, N) = \frac{1}{MN^2}\sum_{j=1}^{M}\mathrm{trace}(\mathbf{R}_{\mathbf{h}}^j), \tag{10}$$

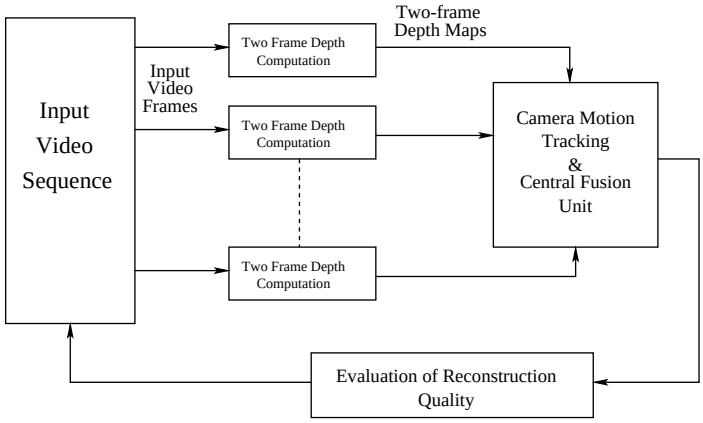

Fig. 1. Block diagram of the 3D reconstruction framework.

where the superscript in the index to the frame number. We will call (10) the multi-frame SfM (MFSfM) rate-distortion function [19], hereafter referred to as the video rate-distortion (VRD) function. Given a particular tolerable level of distortion, the VRD specifies the minimum number of frames necessary to achieve that level. In [20,34] we had proposed an alternative information theoretic criterion for evaluating the quality of a 3D reconstruction from video and had analyzed the comparative advantages and disadvantages. The above results do not require the standard assumptions of Gaussianity of observations and is thus an extension of the error covariance results presented in [21].

2.2. SfM algorithm for face reconstruction

Fig. 1 shows a block-diagram schematic of the complete 3D face reconstruction framework using SfM. The input is a monocular video sequence. We choose an appropriate two-frame depth reconstruction strategy [17]. The depth maps are aligned to a single frame of reference and the aligned depth maps are fused together using stochastic approximation.

Let $\mathbf{s}^i \in \mathbf{R}^3$ represent the structure,[1] computed for a particular point, from ith and $(i+1)$st frame, $i = 1, \ldots, K$, where the total number of frames is $K+1$.[2] Let the fused structure sub-estimate at the ith frame be denoted by $\mathbf{S}^i \in \mathbf{R}^3$. Let $\mathbf{\Omega}^i$ and $\mathbf{V}^i$ represent the rotation and translation of the camera between the ith and $(i+1)$st frames. Note that the camera motion estimates are valid for all the points in the object in that frame. The 3×3 rotation matrix $\mathbf{P}^i$ describes the change of coordinates between times i and $i+1$, and is orthonormal with positive determinant. When the rotational

[1] In our description, subscripts will refer to feature points and superscripts will refer to frame numbers. Thus x_i^j refers to the variable x for the ith feature point in the jth frame.

[2] For notational simplicity, we use i and $(i+1)$st frames to explain our algorithm. However, the method can be applied for any two frames provided the constraints of optical flow are not violated.

velocity $\mathbf{\Omega}$ is held constant between time samples, $\mathbf{P}$ is related to $\mathbf{\Omega}$ by $\mathbf{P} = e^{\hat{\mathbf{\Omega}}}$.[3] The fused sub-estimate $\mathbf{S}^i$ can now be transformed as $T^i(\mathbf{S}^i) = \mathbf{P}^i\mathbf{S}^i + \mathbf{V}^{i^{\mathrm{T}}}$. But in order to do this, we need to estimate the motion parameters $\mathbf{V}$ and $\mathbf{\Omega}$. Since we can determine only the direction of translational motion $(v_x/v_z, v_y/v_z)$, we will represent the motion components by the vector $\mathbf{m} = [v_x/v_z, v_y/v_z, \omega_x, \omega_y, \omega_z]$. Thus, the problems at stage $(i+1)$ will be to (i) reliably track the motion parameters obtained from the two-frame solutions and (ii) fuse $\mathbf{s}^{i+1}$ and $T^i(\mathbf{S}^i)$. If $\{l^i\}$ is the transformed sequence of inverse depth values with respect to a common frame of reference, then the optimal value of the depth at the point under consideration is obtained as

$$u^* = \arg\min_u \mathrm{median}_i \left(w_l^i (l^i - u)^2 \right), \tag{12}$$

where $w_l^i = (\mathbf{R}_{\mathbf{h}}^i(l))^{-1}$, with $\mathbf{R}_{\mathbf{h}}^i(l)$ representing the covariance of l^i (which can be obtained from (9)). However, since we will be using a recursive strategy, it is not necessary to align all the depth maps to a common frame of reference a priori. We will use a Robbins–Monro stochastic approximation (RMSA) algorithm (refer to Appendix A) where it is enough to align the fused sub-estimate and the two-frame depth for each pair of frames and proceed as more images become available.

For each feature point, we compute $X^i(u) = w_l^i (l^i - u)^2,\ u \in \mathcal{U}$. At each step of the RM recursion, the fused inverse depth, $\hat{\theta}^{k+1}$, is updated according to (see Appendix A for details)

$$\hat{\theta}^{k+1} = \hat{T}^k(\hat{\theta}^k) - a^k(p^k(\hat{\theta}^k) - 0.5), \tag{13}$$

where a^k is determined by (A.3), $p^k(\hat{\theta}^k) = \mathbf{I}_{[X^k \leqslant \hat{T}^k(\hat{\theta}^k)]}$, $\mathbf{I}$ represents the indicator function, and $\hat{T}^k$ is the estimate of the camera motion. When $k = K$, we obtain the fused inverse depth $\hat{\theta}^{K+1}$, from which we can get the fused depth value $\mathbf{S}^{K+1}$. The camera motion $\hat{T}$ is estimated using a tracking algorithm as described in [12].

The reconstruction algorithm. Assume that we have the fused 3D structure $\mathbf{S}^i$ obtained from i frames and the two-frame depth map $\mathbf{s}^{i+1}$ computed from the ith and $(i+1)$st frames. Fig. 2 shows a block diagram of the multi-frame fusion algorithm. The main steps of the algorithm are:

Track. Estimate the camera motion using the camera motion tracking algorithm [12].

Transform. Transform the previous model $\mathbf{S}^i$ to the new reference frame.

Update. Update the transformed model using $\mathbf{s}^{i+1}$ to obtain $\mathbf{S}^{i+1}$ from (13).

Evaluate reconstruction. Compute a performance measure for the fused reconstruction from (10).

Iterate. Decide whether to stop on the basis of the performance measure. If not, set $i = i + 1$ and go back to Track.

[3] For any vector $\mathbf{a} = [a_1, a_2, a_3]$, there exists a unique skew-symmetric matrix

$$\hat{\boldsymbol{a}} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{11}$$

The operator $\hat{\boldsymbol{a}}$ performs the vector product on $\mathbf{R}^3$: $\hat{\boldsymbol{a}}X = \mathbf{a} \times X\ \forall X \in \mathbf{R}^3$. With an abuse of notation, the same variable is used for the random variable and its realization.

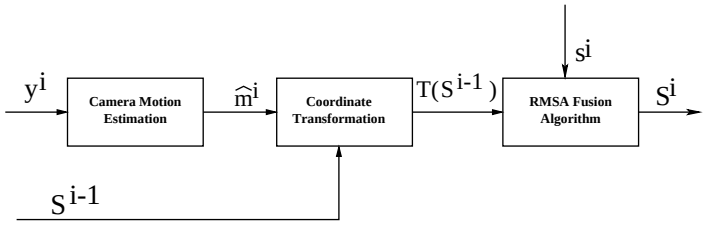

Fig. 2. Block diagram of the multi-frame fusion algorithm.

3. Incorporating the generic model

In this section, we present our method of incorporating the generic model after obtaining the 3D estimate from the video sequence using the SfM algorithm described in the previous section. We propose an optimization framework for combining the two models in such a way that the errors in the 3D estimate are corrected by comparison with the generic model.

The optimization function. Both the generic model and the 3D estimate have a triangular mesh representation with N vertices and the depth at each of these vertices is known. (We will explain how this can be obtained in a later section). Let $\{d_{g_i},\ i = 1,\ldots,N\}$ be the set of depth values of the generic mesh for each of the N vertices of the triangles of the mesh. Let $\{d_{s_i},\ i = 1,\ldots,N\}$ be the corresponding depth values from the SfM estimate. We wish to obtain a set of values $\{f_i,\ i = 1,\ldots,N\}$ which are a smoothed version of the SfM model, after correcting the errors on the basis of the generic mesh.

Since we want to retain the specific features of the face we are trying to model, our error correction strategy works by comparing local regions in the two models and smoothing those parts of the SfM estimate where the *trend* of the depth values is significantly different from that in the generic model, e.g., a sudden peak on the forehead will be detected as an outlier after the comparison and smoothed. This is where our work is different from previous work [9,10], since we do not intend to fuse the depth in the two models but to correct errors based on local geometric trends. Towards this goal, we introduce a line process on the depth values. The line process indicates the borders where the depth values have sudden changes, and is calculated on the basis of the generic mesh, since it is free from errors. For each of the N vertices, we assign a binary number indicating whether or not it is part of the line process. This concept of the line process is borrowed from the seminal work of Geman and Geman [16] on stochastic relaxation algorithms in image restoration.

The optimization function we propose is

$$E(f_i, l_i) = \sum_{i=1}^{N} (f_i - d_{s_i})^2 + (1-\mu)\sum_{i=1}^{N} (f_i - d_{g_i})^2 + \mu \sum_{i=1}^{N} (1 - l_i) \sum_{j \in \mathcal{N}_i} (f_i - f_j)^2 \mathbf{1}_{d_s \neq d_g}, \tag{14}$$

where $l_i = 1$ if the ith vertex is part of a line process and μ is a combining factor which controls the extent of the smoothing. $\mathcal{N}_i$ is the set of vertices which are neighbors of the ith vertex. $\mathbf{1}_{d_s \neq d_g}$ represents the indicator function which is 1 if $d_s \neq d_g$, else 0. In order to understand the importance of (14), consider the third term. When $l_i = 1$, the ith vertex is part of a line process and should not be smoothed on the basis of the values in $\mathcal{N}_i$; hence this term is switched off. Any errors in the value of this particular vertex will be corrected on the basis of the first two terms, which control how close the final smoothed mesh will be to the generic one and the SfM estimate. When $l_i = 0$, indicating that the ith vertex is not part of a line process, its final value in the smoothed mesh is determined by the neighbors as well as its corresponding values in the generic model and SfM estimate. The importance of each of these terms is controlled by the factor $0 < \mu < 1$. In the case (largely academic) where $d_s = d_g$, the smoothed mesh can be either d_s or d_g and this is taken care of in the indicator function in the third term in (14). The line process l_i has a value of 1 or 0, depending on whether there is a sudden change in the depth in the generic model at the particular vertex i. Obtaining such changes relies on derivative computations, which are known to be noise prone. Hence, in our optimization scheme, we allow the line process to be perturbed slightly around its nominal value computed from the generic model.

The design of the cost function follows conventional ideas of regularization theory [15], whereby the energy function usually consists of a data term requiring the solution to be close to the data and a regularizer which imposes a smoothness on the solution. Various other forms of the different terms in (14) could be considered. One interesting variation would be to impose a penalty term for the discontinuities. The graduated non-convexity algorithm of [22] is an appropriate method for solving such problems. It works by first finding the minimum of a convex approximation to the non-convex function, followed by minimization of a sequence of functions, ending with the true cost function. However, based on experimental analysis of the solution of our reconstruction algorithm, we decided that we could work with the simpler version of (14), which does not have the penalty term. This is because, during the optimization, the line process does not move very far from its nominal value, and hence, the penalty term does not have any significant contribution.

Eq. (14) can be optimized in various ways. If the l_i are fixed, this is equivalent to solving a sparse linear system of equations [23, Chapters 9 and 10]. In the parlance of classical deterministic optimization, we then assume that we have perfect information about the loss function and that this information is used to determine the search directions in a deterministic manner in every step of the algorithm. However, as explained before, one of the major sources of noise in (14) is the estimate of l_i. If the l_i are not known perfectly, we need to optimize over this variable also. The cost function can no longer be represented as a linear system of equations. More complicated optimization schemes need to be investigated for this purpose.

We use the technique of simulated annealing built upon the MCMC framework [14]. MCMC is a natural method for solving energy function minimization problems [15]. The MCMC optimizer is essentially a Monte Carlo integration procedure in which the random samples are produced by evolving a Markov chain. Let $T_1 > T_2 > \cdots > T_k > \cdots$ be a sequence of monotone decreasing temperatures in

which T_1 is reasonably large and $\lim_{T_k\to\infty} = 0$. At each such T_k, we run N_k iterations of a Metropolis–Hastings (M–H) sampler [14] with the target distribution represented as $\pi_k(f,l) \propto \exp\{-E(f,l)/T_k\}$. As k increases, π_k puts more and more of its probability mass (converging to 1) in the vicinity of the global maximum of E. Since minimizing $E(f,l)$ is equivalent to maximizing $\pi(f,l)$, we will almost surely be in the vicinity of the global optimum if the number of iterations N_k of the M–H sampler is sufficiently large. The steps of the algorithm are:

- Initialize at an arbitrary configuration f_0, l_0 and initial temperature level T_1. Set $k = 1$.
- For each k, run N_k steps of MCMC iterations with $\pi_k(f,l)$ as the target distribution. Consider the following update strategy. For the line process, consider all the vertices (say $L < N$) for which the nominal value, $l_{i,\text{nominal}} = 1$, and their individual neighborhood sets, $N_1,\ldots,N_L$. For each $l_i = 1$, consider the neighborhood set among $N_1,\ldots,N_L$ that it lies in, randomly choose a vertex in this neighborhood set whose value is not already set to 1, and switch the values of l_i and this chosen vertex. Starting from $l_i = l_{i,\text{nominal}}$, this process ensures that the values of l_i do not move too far from the nominal values. In fact, only the vertices lying in the neighborhood sets $N_1,\ldots,N_L$ can take a value of $l_i = 1$. Next, randomly determine a new value of f, using a suitable transition function [14]. With the new values, $f_{\text{new}}, l_{\text{new}}$ of f,l, compute $\delta = E(f_{\text{new}}, l_{\text{new}}) - E(f,l)$. If $\delta < 0$, i.e., the energy decreases with this new configuration, accept $f_{\text{new}}, l_{\text{new}}$; else, accept with a probability ρ. Pass the final configuration of f,l to the next iteration.
- Increase k to $k+1$.

Mesh registration. The optimization procedure described above requires a one-to-one mapping of the vertices $\{d_{s_i}\}$ and $\{d_{g_i}\}$. Once we obtain the estimate from the SfM algorithm, a set of corresponding points between this estimate and the generic mesh is identified. This can be done manually as in [9] or [10] or automatically as described next. This is then used to obtain a registration between the two models. Thereafter, using proper interpolation techniques, the depth values of the SfM estimate are generated corresponding to the (x,y) coordinates of the vertices of the triangles in the generic model. By this method, we obtain the meshes with the same set of N vertices, i.e., the same triangulation.

If we want to perform the registration automatically, we can follow a simple variant of our method for registering wide baseline images [24]. In that paper, we showed that it is possible to register two face images obtained from different viewing directions by considering the similarity of the shape of important facial features (e.g., eyes, nose, etc.) and compensating for the variability of the shape with viewing direction by considering prior information about it. Applying it to this problem is actually simpler because the two meshes are from the same viewing angle. We can consider the 2D projection of the generic mesh and identify the shape of some important facial features a priori. Once the 3D estimate from SfM is obtained, we can take its 2D projection from the same viewing direction (e.g., from the front view), automatically extract the shape of the important features (as described in the paper by using a corner-finder algorithm and k-means clustering [25]) and then register by computing the similarity of the set of two shapes.

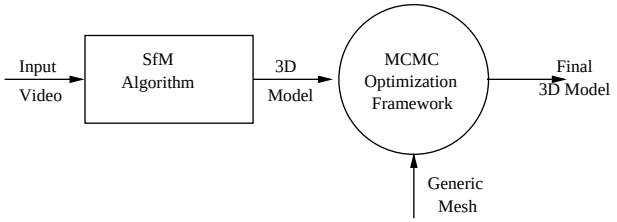

Fig. 3. A block diagram representation of the complete 3D modeling algorithm using the generic mesh and the SfM algorithm.

Choice of μ. There exists substantial literature on how to optimally choose the constant μ for energy functions similar to (14) [15, Chapter 3]. For our problem, we decided to choose the value based on a qualitative analysis of (14). When $l_i = 1$, the third term in the optimization is switched off. These are the points where there are sharp changes in the depth (see Fig. 7). These changes are important characteristics particular to a person's face and need to be retained. However, any errors in these regions should also be corrected. From these considerations, the value of μ is chosen to be between 0.7 and 0.8. When $l_i = 0$, the most important term is the third term of (14), which controls local smoothing. The errors should be corrected by comparing with neighboring values of the 3D estimate, rather than by fusing with the depth values of the vertices of the generic model. With this in mind, we again choose a value of μ between 0.7 and 0.8. This also ensures that the generic model is not given undue importance leading to over-smoothing.

The generic mesh algorithm. The main steps of the algorithm for incorporating the generic mesh are are as follows.

1. Obtain the 3D estimate from the given video sequence using SfM (output of the reconstruction algorithm of Section 2).
2. Register this 3D model with the generic mesh and obtain the depth estimate whose vertices are in one-to-one correspondence with the vertices of the generic mesh.
3. Compute the line processes and to each vertex i assign a binary value l_i.
4. Obtain the smoothed mesh f from the optimization function in (14).
5. Map the texture onto f from the video sequence.

The final 3D model: The complete 3D reconstruction paradigm is composed of a sequential application of the two algorithms (3D Reconstruction Algorithm and the Generic Mesh Algorithm) we have described in Sections 2 and 3. Fig. 3 represents in a block diagram the complete 3D modeling algorithm.

4. Experimental results

We first present the results of our 3D modeling algorithm on three different video sequences. We present the detailed results of our algorithm on one of these sequence.

Then we present a comparative evaluation of our method using examples where the ground truth (i.e., the true depth values) is available.

4.1. SfM algorithm

Figs. 4a and d show two images from the video sequence which is the input to the SfM algorithm. We use an algorithm that computes the structure from the optical flow [17] using two frames and then integrates the multiple two-frame reconstructions over the video sequence using robust estimation techniques (Section 2). The error covariance of the optical flow was estimated a priori over the first few frames of the video sequence, which were not used in the reconstruction. It was done over a sampled grid of points (rather than the dense flow) so as to simplify calculations. The technique used is similar to the gradient-based method of [26], except that, for more accurate results, it was repeated for each of these initial frames and the final estimate was obtained using bootstrapping techniques [27]. This is the stage where the quality of the video data is estimated and incorporated into the algorithm. Assuming that the statistics remain stationary over the frames used in the reconstruction, the error covariance of the 3D reconstruction, $\mathbf{R}_z$ in (7), was computed. The

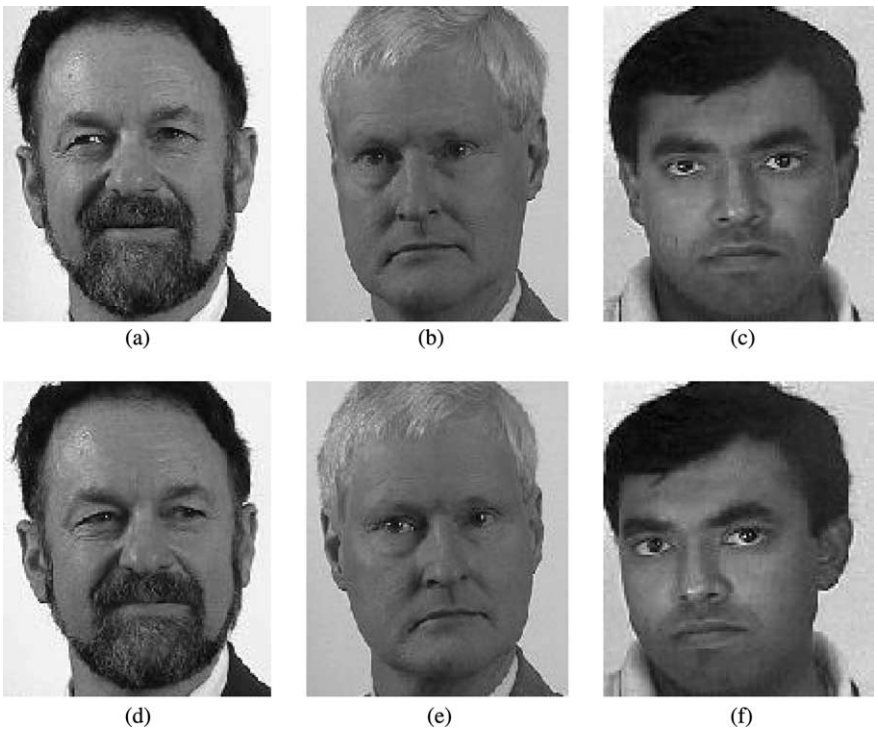

Fig. 4. Two frames from each of the three video sequences of subjects a, b, and c which are used in our experiments.

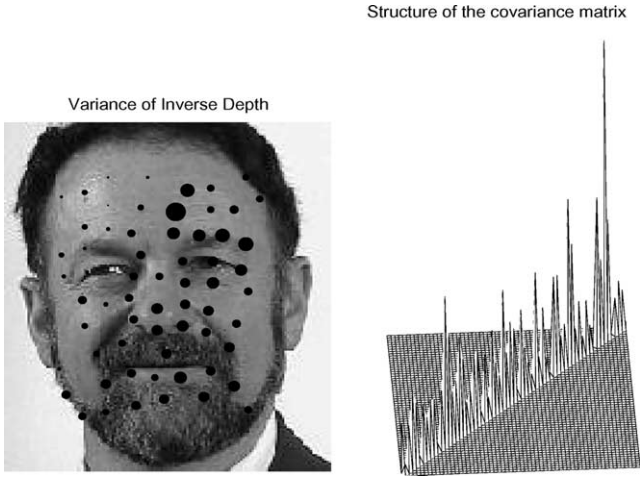

Fig. 5. Plot of the variance of the inverse depth for different features in a face sequence. The diameter of the circle at each feature point is proportional to the variance at that feature point. In the second plot, the diagonal elements of $\mathbf{R_h}$ are shown.

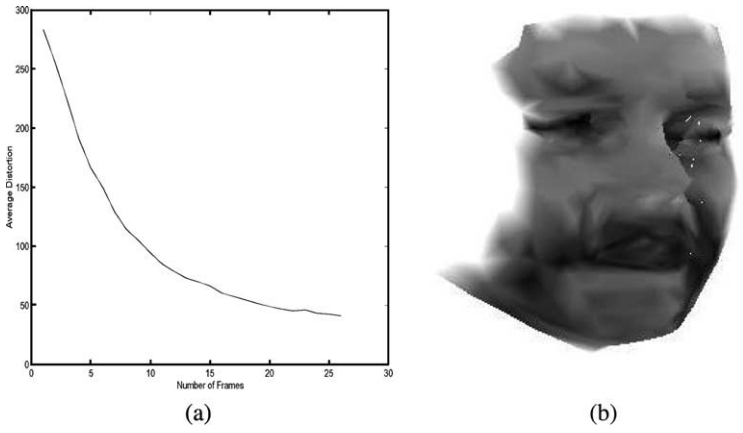

Fig. 6. (a) represents the VRD of the SfM algorithm, i.e., the change in the average distortion with the number of images; (b) depicts one view from the reconstructed model at this stage of the algorithm.

variance in the inverse depth computed using our theoretical analysis of Section 2 is shown in Fig. 5. The diameters of the circles indicate the variance in the inverse depth estimate for the points which were tracked across the video sequence.[4] A plot of the covariance matrix is also shown in the same figure so that it is possible to compare the relative magnitudes of the errors. The quality evaluation of the fusion

[4] For some points with relatively smooth texture, the variance is small, which is counter-intuitive. However, on close observation, it becomes clear that these regions have brighter illumination, and hence the points are tracked better. Also, extremely small variances are not reliable and are not used.

algorithm was done using the rate-distortion function of (10). Fig. 6a plots the VRD curve for 30 frames of the video sequence.

4.2. Reconstruction example without generic model

Fig. 6b shows one particular view from the reconstructed model obtained after completion of the 3D reconstruction algorithm using SfM but without the generic model. The output, without texture mapping, of the multi-frame SfM algorithm is also shown in Fig. 8b, where the model is represented using a triangular mesh. The model shown is obtained after the registration process which was explained in Section 3.[5] It is evident from these plots that the general characteristics of the face are represented; however, it is also clear that a pure SfM algorithm is not enough for a completely satisfactory reconstruction of the face. We now present the effect of the introduction of the generic model into the reconstruction strategy.

4.3. The line process and neighborhood set

Fig. 8a represents the generic model. The line process was calculated on the basis of this generic mesh. In Fig. 7, the vertices of the generic mesh that indicate the boundaries between regions with sharp changes in depth are marked with black 'x's. For these vertices, $l_i = 1$ (in (14)). The local directional derivatives were calculated at each of the vertices of the generic mesh. The vertices at which there was a sharp change in the magnitude of the depth were selected to indicate that they belong to a line process. Thus, the line processes form the boundaries between regions having different depth values and divide the set of vertices into different equivalence classes.

For each vertex, we need to identify a neighborhood set of vertices for the optimization function in (14). The vertices which are within a certain radial distance are identified as belonging to the neighborhood set of the central vertex. However, if a line process is encountered within this region, only those vertices which are in the same equivalence class as the central one are retained in the neighborhood. Since the entire process of determining the line processes and neighborhood sets is done on the generic mesh, it need not be done separately for each 3D model.

In order to optimize over the line process, we adopt the following procedure. For each l_i whose nominal value is 1, we consider the vertices in its neighborhood set. Some of these vertices are perturbed randomly as the optimization proceeds.

4.4. The optimization procedure

The combinatorial optimization function in (14) was implemented using the simulated annealing procedure based on a Metropolis–Hastings sampler. At each temperature we carried out 100 iterations and this was repeated for a decreasing sequence of 20 temperatures. Although this is much below the optimal annealing schedule

[5] The ear region was not obtained from the SfM algorithm but was later stitched on for easy comparison with the other models.

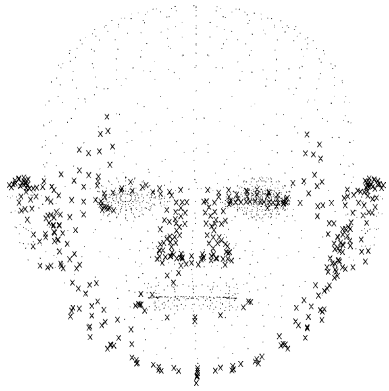

Fig. 7. The vertices which form part of the line processes indicating a change in depth values are indicated with black '×'s.

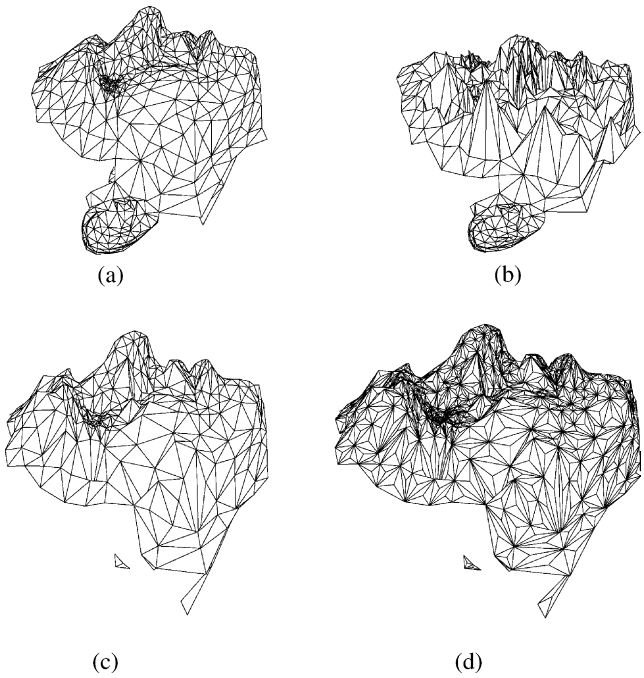

Fig. 8. Mesh representations of the 3D models obtained at different stages of the algorithm. (a) represents the generic mesh, (b) the model obtained from the SfM algorithm (the ear region is stitched on from the generic model in order to provide an easier comparison between the different models), (c) the smoothed mesh obtained after the optimization procedure, and (d) a finer version of the smoothed mesh for the purpose of texture mapping.

suggested by Geman and Geman [16] (whereby the temperature T_k should decrease sufficiently slowly as $\mathcal{O}(\log(\sum_{i=1}^{k} N_i)^{-1})$, N_i being the total number of iterations at temperature T_i), it does give a satisfactory result for our face modeling example. This

is because the optimization algorithm was initialized with the SfM reconstruction obtained from Section 2. A comparative analysis of the results is shown next. We used a value of $\mu = 0.7$ in (14). The final smoothed model is shown in Fig. 8c.

4.5. Texture mapping

Next, we need to map the texture onto the smoothed model in Fig. 8c. This is done using an image from the video sequence corresponding to the front view of the face. Direct mapping of the texture from the image is not possible since the large size of the triangles smears the texture over its entire surface. In order to overcome this problem, we split each of the triangles into smaller ones. This is done only at the final texture mapping stage. The initial number of triangles is enough to obtain a good estimate of the depth values, but not to obtain a good texture mapping. This splitting at the final stage helps us save a lot of computation time, since the depth at the vertices of the smaller triangles is obtained by interpolation, not by the optimization procedure. The fine mesh onto which the texture is mapped is shown in Fig. 8d. Different views of the 3D model after the texture mapping are shown in Fig. 9.

It should be noted that the number of vertices at which we compute the depth is only 638. This is enough to get a good 3D reconstruction largely because of the de-

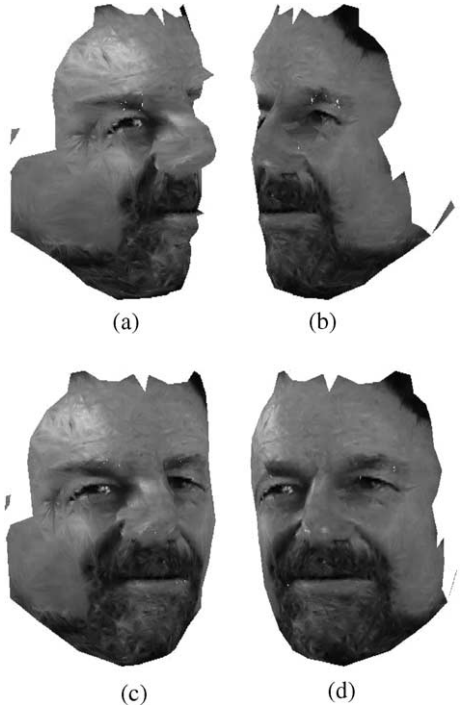

Fig. 9. Different views of the 3D model after texture mapping.

tailed statistical analysis. However, texture mapping with this relatively sparse set of points produces a smearing effect over the surfaces of the triangles in some parts of the face (e.g. in Fig. 4a, the part at the boundary of the face and the beard). Hence we require the sub-division into smaller triangles. Since it is done at the very last stage of the algorithm, the increase in computational cost is small. However, with some of the latest computer graphics software available, this step may not be necessary any more. However, we include it in the paper since it was part of our system when we developed it.

4.6. Computational load

The computational complexity of the entire system can be analyzed by its individual parts. For the two-frame algorithm [17], given a $N \times N$ flow field, the complexity is $\mathcal{O}(N^2 \log N)$. For M frames, the complexity of the fusion algorithm is $\mathcal{O}(N^2 M)$. If the statistics are computed at $N' < N^2$ points, it takes computational power of the order of $\mathcal{O}(N'^2)$. The computational time for the final optimization will be determined by the actual annealing schedule chosen.

We have a running demonstration of the face modeling software. The software runs on a 1.9 GHz P4 PC with 1 GB memory. A JVC DVL9800 video camera is attached to the PC to capture the video. Using a combination of C and MATLAB implementations, the entire reconstruction (from capturing the video sequence, preprocessing it to creating of a final 3D Graphics model) takes about 5 min. This can be substantially reduced by optimizing the code, converting it entirely to C and automating certain pre-processing stages (like identifying the relevant part of the input video sequence). We hope to build a completely automated end-to-end face modeling system in the near future.

4.7. Accuracy analysis of 3D reconstructions

We have applied our algorithm to several video sequences. We present here the results on three such sequences, example images from which are shown in Fig. 4. We will name the three people as subjects A, B, and C. The details, which have been presented for Subject A, are similar for the other examples too. Since the line processes and the neighborhood set are calculated from the generic model, the precomputed results from the previous model were used for this experiment also. Fig. 10 shows two views of the reconstruction for Subject B, two of the original images being depicted in Figs. 4b and e. Similarly, for Subject C (Figs. 4c and f), the reconstruction is shown in Fig. 11.

We computed the accuracy of the 3D reconstruction for these three cases by comparing the projections of the 3D model with the images from the original video sequence. Fig. 12 plots the root mean square (RMS) projection errors as a percentage of the actual values in the original images. In order to depict the change of the error with the viewing angle, the horizontal axis of the figure represents this angle, with 0 indicating the front view. We considered all the combinations between the three subjects, i.e., A–A, A–B, A–C, B–B, B–C, and C–C. From the plots, we see that

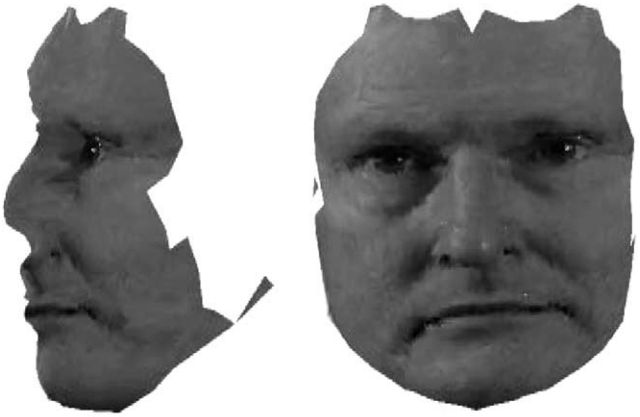

Fig. 10. Different views of the 3D model after texture mapping on the second video sequence.

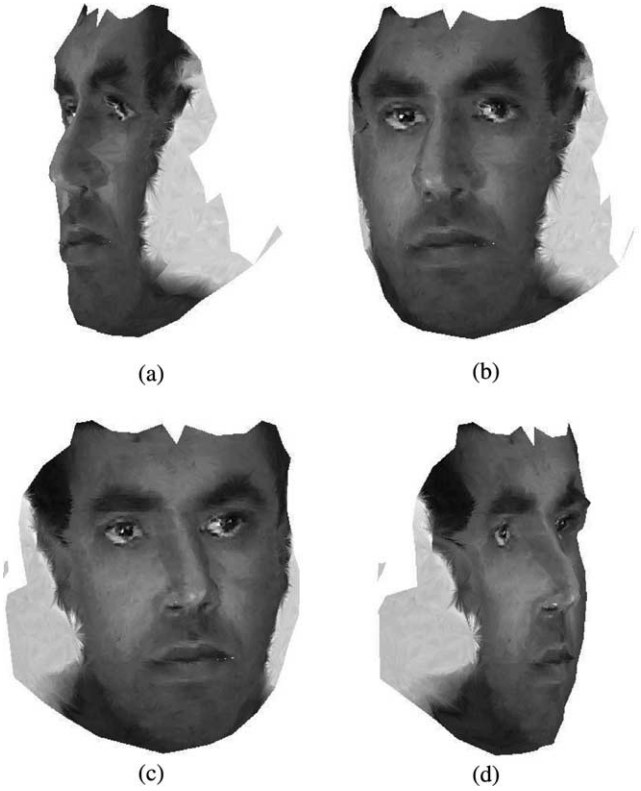

Fig. 11. Different views of the 3D model after texture mapping on the third video sequence.

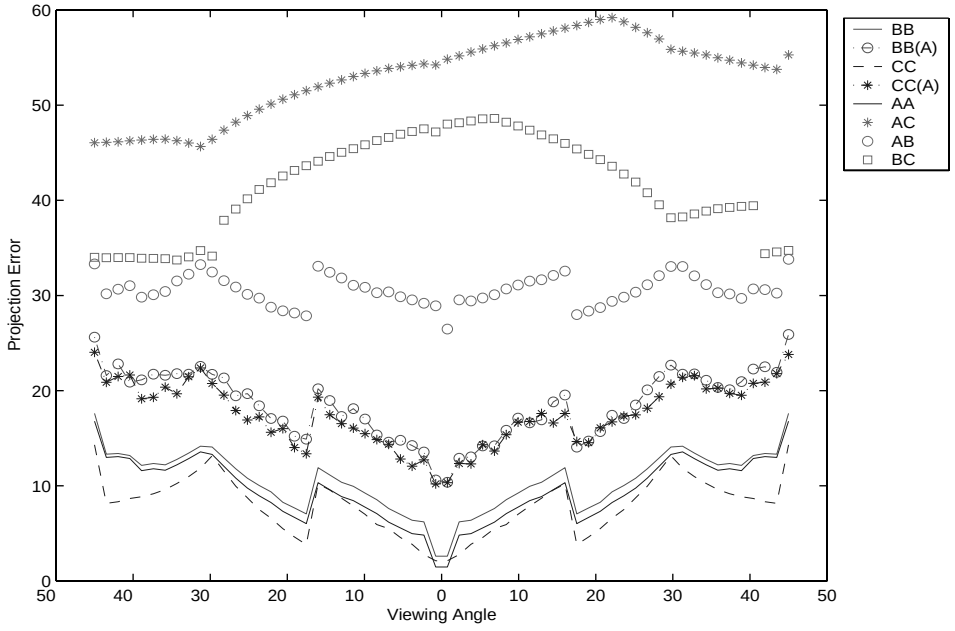

Fig. 12. Percentage errors between the projections of the 3D models of subjects A, B, and C and their images obtained from the video sequences, i.e., AA represents the error between the projections of 3D model of A and the images of A; similarly for AB, AC, BB, BC, and CC. BB(A) represents the projection error when B's texture is overlaid on A's model and the projections are compared to B's images; similarly for CC(A). The error is plotted as a function of the viewing angle which is represented on the x-axis.

the average error in the reconstruction at the front view is about 1%. The error increases with viewing direction, as would be expected. Also there are certain preferred viewing directions, a fact which has been reported before in the literature [28]. In order to obtain an understanding of the role the depth (as opposed to the texture) plays in projections, we considered the case where the texture of B and C are overlaid on the model of A and the projections are compared to the images of B and C, respectively. The experiment was also repeated for the models of B and C with similar results; however, for the sake of clarity, the plots are not shown in Fig. 12. The separation between the error curves for the different combinations holds out hope that 3D models can be used for recognition across pose variations. For example, given the model of Subject A with its proper texture, the projection errors, at any viewing angle, with other subjects is much more than it is with itself.

There are many issues that contribute to the error in face recognition across pose. Accuracy of the 3D model is only one of them. Others are issues of registration of the projections of the model with the image, changes of illumination, etc. This is a separate research problem by itself and is one of our future directions of work. For the experiments in this paper, most of these issues were taken care of manually so that the error values represented in Fig. 12 are mostly due to errors in 3D models (though errors due to other sources cannot be completely eliminated).

4.8. Comparative evaluation of algorithm

In order to analyze the accuracy of the 3D reconstruction directly (as opposed to comparing the 2D projections), we require the ground truth of the 3D models. We experimented with a publicly available database of 3D models obtained from a Minolta 700 range scanner. The data are available on the World Wide Web at http://sampl.eng.ohio-state.edu/sampl/data/3DDB/RID/minolta/faces-hands.1299/index.html. We will report numerical results from our algorithm on some of the data available here, though we will not publish the images or 3D models of the subjects.

In order to perform an accurate analysis of our methods, we require a video sequence of the person and the 3D depth values. This, however, is not available on this particular database or on any other that we know of. Thus we had to generate a sequence of images in order to apply our algorithm.[6] This was done using the 3D model and the texture map provided on the web-site. Given these images we performed the following experiments:

- Obtain the 3D reconstruction without using the generic model (Section 2).
- Introduce the generic model at the beginning of the 3D reconstruction algorithm, by initializing (12) with the the values at the vertices of the generic model.[7] Note that the statistical error analysis of the video data is still done.
- Apply the algorithm in this paper, which postpones the introduction of the generic model.

We considered the error in the 3D estimate in all these methods compared to the actual 3D values. Fig. 13 plots the percentage RMS errors (percentage taken with respect to the true value) as a function of the percentage difference of the specific 3D model (as obtained from the website) and the generic one. The percentages on the horizontal axis are calculated with respect to the generic model, while those on the vertical axis are computed with respect to the ground truth of the 3D model of the particular subject. The first five subjects on that website, referred to as "frame001"–"frame005," were considered. The percentage differences of the specific 3D models with the generic one are tabulated in Table 1. From the figure, it is clear that if the generic model is introduced at an early stage of the algorithm, the error in the reconstruction increases as the model of the subject deviates from the generic one. On the other hand, if the generic model is introduced later (as in our algorithm), the error in the reconstruction remains approximately constant. However, the reconstructions for the case where the generic model is introduced earlier (e.g., [9,10]) are visually very pleasing.

An idea of the progress of the optimization can be obtained from the following experiment, the results of which are shown in Fig. 14. The experiment was done

[6] The optical flow computed with the generated sequences may be more accurate than in a normal setting. However, for all the methods that are compared, the images used are the same. Hence, it is reasonable to assume that the effect of the image quality would be similar in all three cases. Hence the comparison of the 3D reconstruction accuracy, keeping all other factors constant, should still be useful.

[7] We cannot compare it directly with [9] or [10] because of substantial differences in the input data.

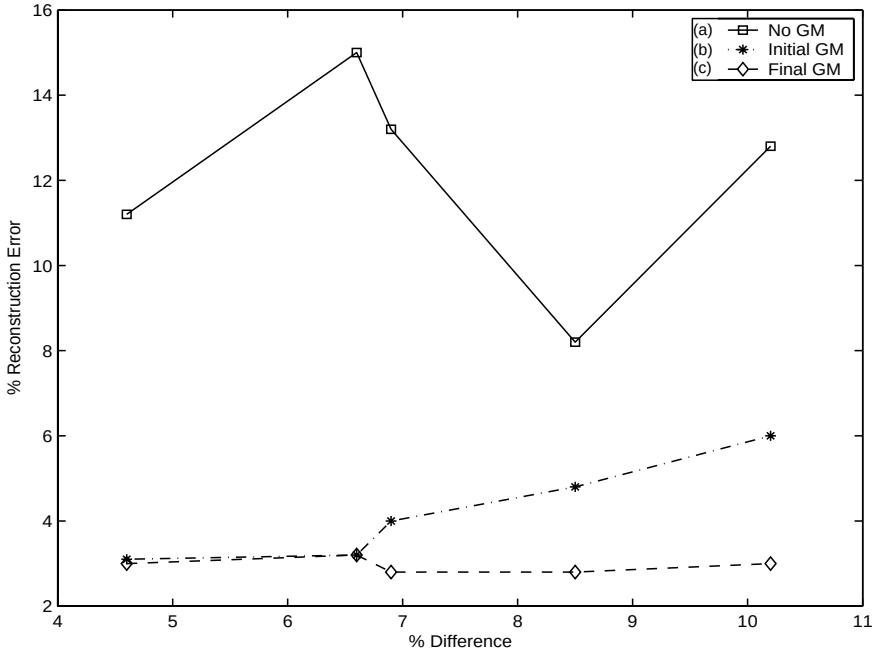

Fig. 13. The error in 3D reconstruction when (a) no generic model (GM) is used; (b) the generic model is used to initialize the reconstruction algorithm; (c) the generic model is used later as described in this paper. The error is plotted as a function of the difference of the specific 3D model with the generic one. Five subjects were considered in this experiment.

Table 1
Average percentage difference of the 3D models from the generic model

Subject index	Percentage difference
1 (frame 001)	10.2
2 (frame 002)	8.5
3 (frame 003)	4.6
4 (frame 004)	6.6
5 (frame 005)	6.9

on Subject 1 of Table 1, which is an interesting case since the face is very different from the generic one. We considered 10 significant points on the face (similar to fiducial points). They are the left eye, the bridge between the eyes, the right eye, the left extreme part of the nose, the tip of the nose, the right extreme part of the nose, the left and right ends of the lips, and the center of the left and right cheeks. We considered a window around each of these points and computed the average depth in each of them for the various reconstructions. Fig. 14 plots the average depth at each of these points for the following cases: true depth, depth with generic model introduced later (our algorithm), depth with generic model used as the initial value,

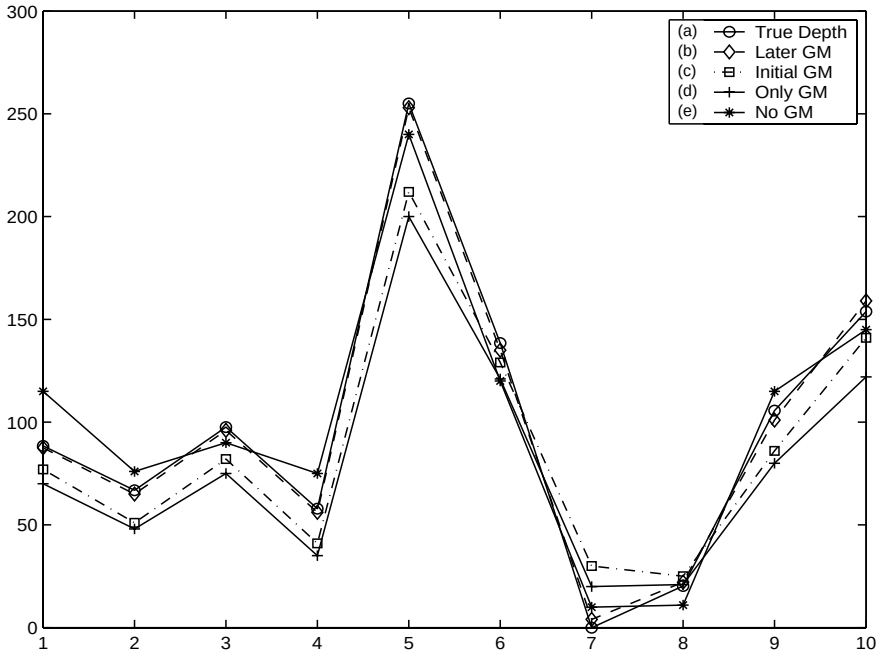

Fig. 14. Plots of (a) the true depth, (b) the depth with generic model introduced later (our algorithm), (c) the depth with generic model used as initial value, (d) the depth of the generic model, and (e) the SfM reconstruction with no generic model. The depths are computed in local neighborhoods around a set of fiducial points on the face for Subject 1.

depth of the generic model, and the SfM reconstruction with no generic model. The depth values are normalized between 0 and 255, as in a depth map image. The plots provide some very interesting insights. When the reconstruction is initialized with the generic mesh, the solution at these points do not move very far from the initial value. On the other hand, the SfM estimate with no generic mesh gives a solution not very far from the true value, which is improved further by considering the generic model. Moreover, we find that the average error of the final reconstruction in these fiducial regions is less than the overall average error of Fig. 13. This is very interesting since it shows that these significant regions of the face are reconstructed accurately.

5. Conclusion

In this paper we have presented a novel method of 3D modeling of a face from a monocular video sequence using an SfM algorithm and a generic face model. In previous approaches, the generic model was used to initialize the SfM algorithm. Using this method, the final solution often converged very close to the initial value, resulting in a reconstruction which had the characteristics of the generic model rather than

those of the particular face in the video which needs to be modeled. One of the main contributions of our work lies in the fact we incorporated the generic model *after* the SfM algorithm, which obtains the 3D estimate purely from the input video sequence. The other main contribution of this work was that the quality of the specific video data was analyzed and information about it incorporated into the 3D modeling algorithm. The 3D structure estimation process was based on fusing the estimates obtained from pairs of frames from the video sequence, after computing the uncertainties of the two-frame solutions. In order to combine the generic model with this 3D estimate, we used an energy function minimization procedure. The results of our method at different stages of the algorithm and a comparative study with other methods were presented. Extensions of the work would involve applications to face recognition across pose, facial animations from video, multimedia communications, etc.

Acknowledgments

The authors thank Prof. Pascal Fua of EPFL, Switzerland for many discussions on 3D face reconstruction during his visit to Maryland in the summer of 2001. They also acknowledge Tai Vo and Sandeep Krishnamurthy who contributed significantly in developing the code for the generic mesh algorithm.

Appendix A. Stochastic approximation

The method of stochastic approximation (SA) is useful for certain sequential parameter estimation problems [29]. Let $\{e(k)\}$ be a sequence of random variables with the same distribution indexed by a discrete time variable k. A function $Q(x, e(k))$ is given such that

$$E[Q(x, e(k))] = g(x) = 0, \tag{A.1}$$

where E denotes expectation over e. The distribution of $e(k)$ is not known; the exact form of the function $Q(x, e)$ may also be unknown, though its values are observed and it can be constructed for any chosen x. The problem is to determine the solution of $g(x) = 0$. Robbins and Monro (RM) [30] suggested the following scheme for solving (A.1) recursively as time evolves:

$$\hat{x}(k) = \hat{x}(k-1) + a_k Q(\hat{x}(k-1), e(k)), \tag{A.2}$$

where the gain sequence $\{a_k\}$ must satisfy the following conditions [29,31,32]:

$$a_k \geqslant 0, \quad a_k \to 0, \quad \sum_{k=1}^{\infty} a_k = \infty, \quad \sum_{k=1}^{\infty} a_k^2 < \infty. \tag{A.3}$$

A popular choice of the gain sequence, which was used in our experiments also, is $a_k = a/(k+1)^{0.501}$. It can be shown that the estimate obtained from SA is unbiased, consistent, and asymptotically normal, and in many cases, also efficient [31,33].

Using the above ideas, the recursion of (13) can be derived. Our aim is to compute the median (say θ) of $X^0,\ldots,X^K$, i.e., to obtain θ such that $g(\theta)=F_X(\theta)-0.5=0$, where $F_X(\theta)$ is the distribution function of θ. Define $Y^k(\hat{\theta}^k)=p^k(\hat{\theta}^k)-0.5$, where $p^k(\hat{\theta}^k)=\mathbf{I}_{[X^k\leqslant\hat{T}^k(\hat{\theta}^k)]}$ ($\mathbf{I}$ represents the indicator function, $\hat{T}^k$ is the estimate of the camera motion and $\hat{\theta}^k$ is the estimate obtained at the kth stage). Then

$$E[Y^k(\hat{\theta}^k)|\hat{\theta}^k]=E[p^k(\hat{\theta}^k)|\hat{\theta}^k]-0.5=E[\mathbf{I}_{[X^k\leqslant\hat{T}^k(\hat{\theta}^k)]}]-0.5=P(X^k\leqslant\hat{T}^k(\hat{\theta}^k))-0.5$$
$$=F_X(\hat{\theta}^k)-0.5=g(\hat{\theta}^k).$$

Hence Eq. (13) follows from (A.2).

References

[1] R.I. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, 2000.
[2] Z. Zhang, O. Faugeras, 3D Dynamic Scene Analysis, Springer-Verlag, Berlin, 1992.
[3] J. Oliensis, A critique of structure from motion algorithms, Techical Report. Available from http://www.neci.nj.nec.com/homepages/oliensis/, NECI, 2000.
[4] T. Jebara, A. Pentland, Parameterized structure from motion for 3d adaptive feedback tracking of faces, in: Conference on Computer Vision and Pattern Recognition, 1997, pp. 144–150.
[5] C. Bregler, A. Hertzmann, H. Biermann, Recovering non-rigid 3D shape from image streams, in: Conference on Computer Vision and Pattern Recognition, 2000, pp. II:690–696.
[6] M. Brand, Morphable 3D models from video, in: Conference on Computer Vision and Pattern Recognition, 2001, pp. II:456–463.
[7] M. Pollefeys, Self-calibration and metric 3D reconstruction from uncalibrated image sequences, PhD Thesis, ESAT-PSI, K.U. Leuven, 1999.
[8] S. Romdhani, V. Blanz, T. Vetter, Face identification by fitting a 3d morphable model using linear shape and texture error functions, in: European Conference on Computer Vision, 2002.
[9] P. Fua, Regularized bundle-adjustment to model heads from image sequences without calibration data, Internat. J. Comput. Vision 38 (2) (2000) 153–171.
[10] Y. Shan, Z. Liu, Z. Zhang, Model-based bundle adjustment with application to face modeling, in: International Conference on Computer Vision, 2001, pp. 644–651.
[11] V. Nalwa, A Guided Tour of Computer Vision, Addison-Wesley, Reading, MA, 1993.
[12] A. Roy Chowdhury, Statistical analysis of 3D modeling from monocular video streams, PhD Thesis, Univeristy of Maryland, College Park, 2002.
[13] A. Roy Chowdhury, R. Chellappa, Stochastic approximation and rate-distortion analysis for robust structure and motion estimation, Internat. J. Comput. Vision, in press.
[14] A. Doucet, On sequential simulation based methods for Bayesian filtering, Statist. Comput. 10 (3) (1998) 197–208.
[15] J. Clark, A. Yuille, Data Fusion for Sensory Information Processing Systems, Kluwer, Dordrecht, 1990.
[16] S. Geman, D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, IEEE Trans. Pattern Anal. Machine Intell. 6 (6) (1984) 721–741.
[17] S. Srinivasan, Extracting structure from optical flow using fast error search technique, Internat. J. Comput. Vision 37 (2000) 203–230.
[18] R. Walter, Principles of Mathematical Analysis, third ed., McGraw-Hill, New York, 1976.
[19] T. Cover, J. Thomas, Elements of Information Theory, John Wiley and Sons, New York, 1991.
[20] A. Roy Chowdhury, R. Chellappa, An information theoretic criterion for evaluating the quality of 3D reconstructions from video, IEEE Trans. Image Process. (under review).
[21] G. Young, R. Chellappa, Statistical analysis of inherent ambiguities in recovering 3D motion from a noisy flow field, IEEE Trans. Pattern Anal. Machine Intell. 14 (1992) 995–1013.

[22] A. Blake, A. Zisserman, Visual Reconstruction, MIT Press, Cambrdige, MA, 1987.
[23] G. Golub, C. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, MD, 1989.
[24] A. Roy Chowdhury, R. Chellappa, T. Keaton, Wide baseline image registration with application to 3D face modeling, IEEE Trans. Multimedia (accepted).
[25] R. Duda, P. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.
[26] Z. Sun, V. Ramesh, A. Tekalp, Error characterization of the factorization method, Comput. Vision Image Understanding 82 (2001) 110–137.
[27] B. Efron, R. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, London, 1993.
[28] W. Zhao, R. Chellappa, A. Rosenfeld, P. Phillips, Face recognition: a literature survey, Technical Report, CAR-TR-948, University of Maryland, College Park, 2000.
[29] A. Benveniste, M. Metivier, P. Priouret, Adaptive Algorithms and Stochastic Approximations, Springer-Verlag, Berlin, 1987.
[30] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Statist. 22 (1951) 400–407.
[31] L. Ljung, T. Soderstrom, Theory and Practice of Recursive Identification, MIT Press, Cambridge, MA, 1987.
[32] J. Spall, Introduction to Stochastic Search and Optimization, Wiley, New York, 2000.
[33] H. Poor, An Introduction to Signal Detection and Estimation, Springer-Verlag, Berlin, 1988.
[34] Amit K. Roy Chowdhury, R. Chellappa, Toward a criterion for evaluating the quality of 3D reconstructions, Proc. of Intl. Conf. on Acoustics, Speech and Signal Processing, Orlando, Florida, 2002.