3D FACE RECONSTRUCTION FROM VIDEO USING A GENERIC MODEL

A.Roy Chowdhury[*]*, R.Chellappa*

Center for Automation Research and
Department of ECE
University of Maryland
College Park, MD 20742
{amitrc,rama}@cfar.umd.edu

S.Krishnamurthy[†]

CACC, Dept. of ECE
North Carolina State University,
Raliegh, NC
shkrishn@unity.ncsu.edu

T.Vo[‡]

Department of CS
California State University,
Fullerton, CA
taivip@yahoo.com

ABSTRACT

Reconstructing a 3D model of a human face from a 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 common method of overcoming this problem is to use a generic model of a face. 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. A 3D estimate is obtained purely from the video sequence using SfM algorithms without use of the generic model. 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 local regions in the two models. The optimization is done using a Markov Chain Monte Carlo (MCMC) sampling strategy. 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. The evolution of the 3D model through the various stages of the algorithm is presented.

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], the quality of such reconstructions is often poor. 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 [2]. One particularly interesting application of 3D reconstruction from 2D images is in the area of modeling a human face from video. The successful solution of this problem has immense potential for applications in face recognition, surveillance, multimedia etc.

A few algorithms exist which attempt to solve this problem using a generic model of a face [3, 4]. Their typical approach is to initialize 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. This method might 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 very different from those being reconstructed, the solution from this approach may be highly erroneous.

We propose an alternative way of reconstructing a 3D model of a face. Our method also incorporates a generic model; however, we do so *after* obtaining the estimate from the SfM algorithm. The SfM algorithm reconstructs purely from the video data. This reconstruction is fused with the generic model in an energy function minimization framework [5]. The 3D estimate obtained from the 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 [6]) depending upon whether or not it is part of a depth boundary. The regions which are inside these boundaries are smoothed. 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 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 [7, 8, 9]. 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. Moreover, any errors in the reconstruction are corrected in the energy function minimization process by comparison with the generic model.

[*]Partially supported by NSF grant #0086075

[†]The author worked on this problem during his stay at Maryland in Fall 2001.

[‡]The author developed a part of the code during his summer internship at Maryland in 2001

2. INCORPORATING THE GENERIC MODEL

2.1. The Optimization Function

In this section, we will explain our method of incorporating the generic model after obtaining the 3D estimate from the video sequence using any standard SfM algorithm [10]. 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. Let $\{d_{g_i}, i = 1, ..., N\}$ be the set of depth values of the generic model for each of these N vertices of the triangles of the mesh. Let $\{d_{s_i}, i = 1, ..., N\}$ be the corresponding depth values from the SfM estimate. We wish to obtain a set of values $\{f_i, i = 1, ..., N\}$ which are a smoothed version of the SfM estimate, after correcting the errors on the basis of the generic model.

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 3D 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 significantly different from previous ones [3, 4], since we do not intend to fuse the depth in the two models but correct the errors based on the local 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 model, since it is free from errors. For each of the N vertices, we assign a binary number indicating whether it is part of the line process or not. This concept of the line process is borrowed from the seminal work of Geman and Geman [6] on stochastic relaxation algorithms in image restoration.

The optimization function we propose is

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

where $l_i = 1$ if the i^{th} 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 i^{th} vertex. $\mathbf{1}_{d_s \neq d_g}$ represents the indicator function which is 1 is $d_s \neq d_g$, else 0. In order to understand the importance of (1), consider the third term. When $l_i = 1$, the i^{th} 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 model will be to the generic model and the 3D estimate. When $l_i = 0$, indicating that the i^{th} vertex is not part of a line process, its final value in the smoothed model 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 model can be either d_s or d_g and this is taken care of in the indicator function in the third term in (1).

In order to solve the optimization problem in (1), we use the technique of simulated annealing built upon the Markov Chain Monte Carlo (MCMC) framework [9, 8, 11]. The MCMC optimizer is essentially a Monte Carlo integration procedure in which the random samples are produced by evolving a Markov chain. Let $T_1 > T_2 > ... > T_k > ...$ be a sequence of monotone decreasing temperatures in which T_1 is reasonably large and $\lim_{T_k \to \infty} = 0$. At each T_k, we run N_k iterations of a Metropolis-Hastings (M-H) sampler with the target distribution $\pi_k(f) \propto \exp\{-E(f)/T_k\}$ [7, 12]. 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 $E(f)$ is equivalent to maximizing $\pi(f)$, 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 and initial temperature level T_1.
- For each k, run N_k steps of MCMC iterations with $\pi_k(f)$ as the target distribution. Pass the final configuration of x to the next iteration.
- Increase k to $k + 1$.

2.2. Model 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 manually (as in [3, 4]). This is then used to obtain a registration between the two models. Thereafter, using proper interpolation techniques (standard MATLAB functions), 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.

2.3. Algorithm

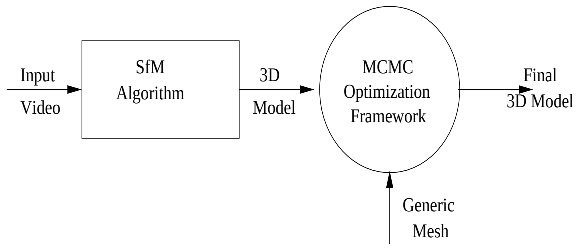

Fig. 1. A block diagram representation of the 3D modeling algorithm.

Figure (1) represents in a block diagram the 3D modeling algorithm, whose main steps are as follows.
1. Obtain the 3D estimate from the given video sequence using any standard SfM algorithm.
2. Register this 3D model with the generic model and obtain the depth estimate whose vertices are in one-to-one correspondence with the vertices of the generic model.
3. Compute the line processes and to each vertex i assign a binary value l_i.
4. Obtain the smoothed model f_i from the optimization function

in (1).
5. Map the texture onto f from the video sequence.

3. 3D MODELING RESULTS

3.1. SfM Algorithm

Figure 2 depicts two images from the video sequence which is the input to the SfM algorithm. We use a two frame algorithm that computes the structure from the optical flow [13] using two consecutive frames and then integrate over the video sequence using robust estimation techniques [14]. The output of the multi-frame SfM (MFSfM) algorithm is shown in Figure 5(b), where the model is represented using a triangular mesh. The model shown is obtained after the registration process explained in Section 2.[1] The 3D reconstruction from the SfM algorithm using 30 frames and after texture mapping is shown in Figure 3(b). The decrease in the distortion in the 3D estimate with increasing number of images [14] is shown in Figure 3(a).

3.2. The Line Process and Neighborhood Set

Figure 5(a) represents the generic model, on the basis of which the line processes were calculated. In Figure 4 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 (1)). 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 (1). 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.

3.3. The Optimization Procedure

The combinatorial optimization function in (1) 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 suggested by Geman and Geman [6] (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. We used a value of $\mu = 0.7$ in (1). The final smoothed model is shown in Figure 5(c).

[1]The ear region is stitched on from the generic model in order to provide an easier comparison between the different models.

3.4. Texture Mapping

Next, we need to map the texture onto the smoothed model in Figure 5(c). Direct mapping of the texture from the video sequence 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 Figure 5(d). Different views of the 3D model after the texture mapping are shown in Figure 6.

The main limitation of this work is our inability to compare against other methods. The reason for that is the difficulty of obtaining 3D ground truth and the implementation of other proposed algorithms. Hence we have no other option but to rely on our visual judgement for evaluating the quality of the reconstruction.

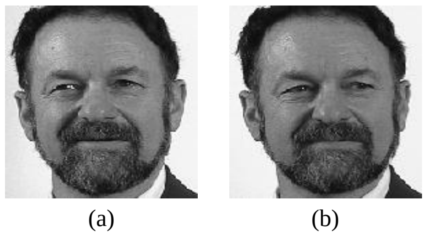

Fig. 2. Two views from the original video sequence which is the input to the SfM reconstruction algorithm.

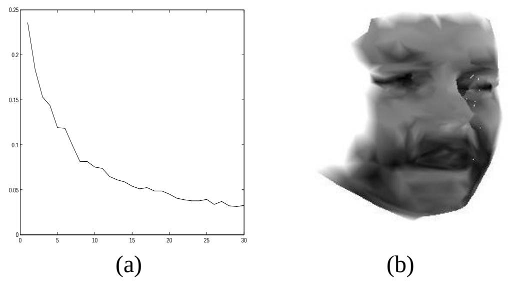

Fig. 3. (a) represents the change in the distortion of the 3D estimate from the SfM algorithm with the number of images; (b) depicts one view from the reconstructed model at this stage of the algorithm.

4. CONCLUSION

In this paper we have presented a novel method of 3D modeling of a face from a video sequence using an SfM algorithm and a generic face model. In previous approaches, the generic model was used to initialize the SfM algorithm. The problem with this approach was that 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. The main contribution of our

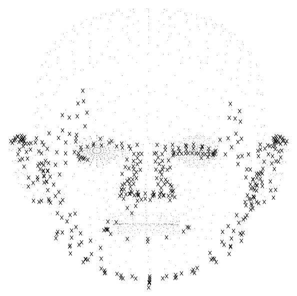

Fig. 4. The vertices which form part of the line processes indicating a change in depth values are indicated with black 'x's.

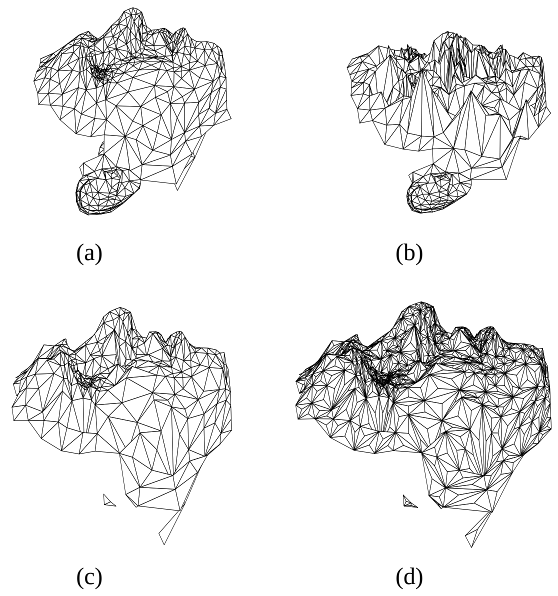

Fig. 5. Mesh representations of the 3d models obtained at different stages of the algorithm. (a) represents the generic mesh, (b) the model obtained from the SfM algorithm (the ear region is stitched on from the generic model in order to provide an easier comparison between the different models), (c) the smoothed mesh obtained after the optimization procedure and (d) a finer version of the smoothed mesh for the purpose of texture mapping.

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. Also, instead of combining the depth values of the two models, we used an optimization framework whereby the local *trends* in the 3D structure between the two models are compared and errors in the specific model are corrected for. In order to combine the generic model with this 3D estimate, we used an energy function minimization procedure. The optimization was done in a MCMC framework using a Metropolis-Hastings sampling strategy. The results of our method at different stages of the algorithm were presented.

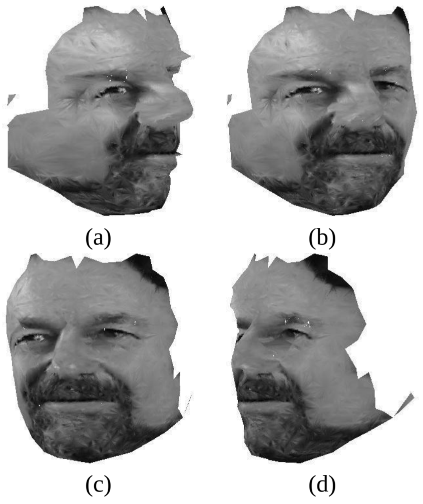

Fig. 6. Different views of the 3d model after texture mapping.

5. REFERENCES

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

[2] J. Oliensis, "A critique of structure from motion algorithms," Tech. Rep. http://www.neci.nj.nec.com/homepages/oliensis/, NECI, 2000.

[3] P. Fua, "Regularized bundle-adjustment to model heads from image sequences without calibration data," *International Journal of Computer Vision*, vol. 38, no. 2, pp. 153–171, July 2000.

[4] Y. Shan, Z. Liu, and Z. Zhang, "Model-based bundle adjustment with application to face modeling," in *International Conference on Computer Vision*, 2001, pp. 644–651.

[5] J.J. Clark and A.L. Yuille, *Data Fusion for Sensory Information Processing Systems*, Kluwer, 1990.

[6] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, November 1984.

[7] A. Doucet, "On sequential simulation based methods for bayesian filtering," *Statistics and Computing*, vol. 10(3), pp. 197–208, 1998.

[8] S. Kirpatrick, C.D. Gelatt, Jr., and M.P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[9] Jun S. Liu, "Markov chain Monte Carlo and related topics," Tech. Rep., Stanford University, 1999.

[10] Z. Zhang and O. Faugeras, *3D Dynamic Scene Analysis*, Springer-Verlag, 1992.

[11] J. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93(443), 1998.

[12] R.M. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," Tech. Rep. CRG-TR-93-1, University of Toronto, 1993.

[13] S. Srinivasan, "Extracting structure from optical flow using fast error search technique," *International Journal of Computer Vision*, vol. 37, pp. 203–230, 2000.

[14] A. Roy Chowdhury and R. Chellappa, "Stochastic approximation and rate-distortion analysis for robust structure and motion estimation," Tech. Rep., CS-TR-4261, University of Maryland, College Park, 2001.