Time Varying Surface Reconstruction from Multiview Video

S. Cihan Bilir*   Yücel Yemez†

Multimedia, Vision and Graphics Laboratory, Koç University, Istanbul, Turkey

Abstract

We present a fast deformation-based method for building time-varying surface models of dynamic objects from multiview video streams. Starting from an initial mesh representation, the surface of a dynamic object is tracked over time, both in geometry and connectivity, based on multiview silhouette information via a mesh-based deformation technique. The resulting smooth time-varying surface is then represented as a mesh sequence that can efficiently be encoded in terms of mesh restructuring operations and small-scale vertex displacements along with the initial model. Another advantage of the proposed method is the ability to deal with dynamic objects that may undergo a nonrigid transformation. We demonstrate the performance of the presented method on a synthetic human body model sequence.

Keywords: Time-varying surface reconstruction, surface deformation, surface tracking.

Index Terms: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Boundary representations; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Tracking

1 Introduction

3D time-varying modeling of dynamic real scenes is an emerging research topic with applications in various domains such as 3D television, virtual reality, and 3D animation. The major challenges for dynamic scene modeling concern the resulting representation load and the computational efficiency of the reconstruction method. A time-varying scene sampled at a standard rate of 30 frames per second would yield enormous 3D model data for representation and a considerable amount of time for reconstruction if no particular care is shown to exploit redundancies between consecutive time frames. In this respect, time-varying mesh representations with a connectivity as fixed as possible, but with changing vertex positions, would certainly provide efficiency both for storage and processing, and also for visualization. In this work, we present a fast deformation-based reconstruction method for modeling dynamic objects based on multiview silhouette information. Starting from an initial mesh representation, the surface of a dynamic object is tracked over time, both in geometry and connectivity, via a mesh-based deformation technique. The mesh representation of each frame is obtained by deforming the mesh representation of the previous frame towards the optimal isosurface, using mesh restructuring operations and vertex displacements. These mesh deformation operations and small-scale displacements along with the initial mesh representation yield a compact and smooth representation of the whole time-varying surface.

There exist several works in the literature, that address 3D modeling of time-varying scenes from multiview video sequences [4, 6, 1]. All these methods assume that the object of interest moving in the scene is a human actor. The method presented in [4] is based on volumetric level-set technique and builds a spatially and temporally smooth surface model from multiview color consistency and silhouette information. The proposed method is however computationally very expensive, necessitating a parallel implementation of the algorithm and does not address the problem of time-consistent mesh representation. In [6], the mesh representation of the time-varying surface is reconstructed from scratch, separately for each time instance. The vertices of the resulting meshes are then matched so as to obtain a time-consistent representation. However since there is no space-time analysis involved during reconstruction, the resulting time-varying surface is not necessarily smooth over time, and moreover, the time-consistency can only be achieved for very short time intervals. On the other hand, the methods based on multi-camera motion capture, such as the one in [1], are concerned only with the motion of the object and applicable only to objects with rigid motion, for which a generic 3D model is available a priori.

The time-varying reconstruction technique that we propose in this paper is based on surface deformation. Deformation models have widely been used in various modeling problems of 3D computer graphics and vision such as shape recovery, animation, surface editing, tracking and segmentation [5]. The main motivation behind employing deformable models is that they in general yield smooth, robust representations that can successfully capture and preserve semantics of the data with well established mathematical foundations. They can easily adapt changes occurring in the geometry of the objects under investigation and can therefore be applied

*e-mail: sbilir@ku.edu.tr
†e-mail: yyemez@ku.edu.tr

IEEE International Conference on Shape
Modeling and Applications 2008
4 - 6 June, Stony Brook, New York, USA
978-1-4244-2261-6/08/$25.00 ©2008 IEEE

to modeling time-varying characteristics of dynamic scenes. The deformation model that we use in this paper follows the Lagrangian approach, hence it is basically a mesh-based deformation scheme.

2 System Overview

The block diagram of the overall reconstruction scheme is given in Figure 1. The input to the reconstruction scheme is the multiview video sequence of the dynamic scene captured with a calibrated multicamera system. In this work, since we work with synthetic models, the silhouette images and the calibration parameters are given a priori. The initial surface model of the first frame is reconstructed prior to the surface tracking/deformation task by using any shape from silhouette technique that produces a topologically correct shape model which is eligible for further deformation. The overall time-varying surface representation of a dynamic scene is then reconstructed by successively estimating the surface representation of each time frame by deforming the mesh representation of the previous frame based on the multiview silhouette information. This produces a sequence of meshes, $M^{(0)}, M^{(1)}, ..., M^{(t)}, ...$, representing the time-varying geometry. For the surface evolution to successfully converge to the desired mesh representation, at each frame we first roughly estimate the global rigid motion of the object from its silhouettes and then register the 3D pose of the starting mesh with reference to the target. This initial pose registration does not only improve the chances of the surface evolution to successfully converge to the desired surface, but it also speeds up the deformation process.

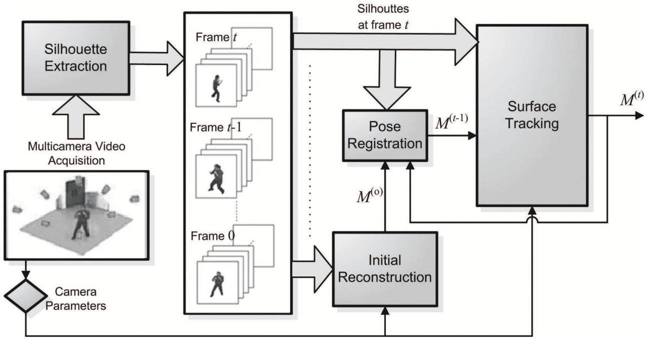

Figure 1: Block diagram of the overall reconstruction system.

3 Deformation Scheme

Our deformation technique is based on the iterative use of an appropriate transformation T that deforms, at each frame t, an initial triangle mesh $M_0^{(t)}$ towards the object surface $S^{(t)}$ through the following surface evolution equation:

$$M_{k+1}^{(t)} = T(M_k^{(t)}). \tag{1}$$

The deformable model $M_k^{(t)}$ is required to remain as a smooth topologically correct mesh representation free of geometrical distortions during its evolution and to converge to an optimal mesh $M_{k^*}^{(t)}$ that faithfully represents the object surface $S^{(t)}$ at the equilibrium state,

$$M_{k^*}^{(t)} = T(M_{k^*}^{(t)}). \tag{2}$$

We define T as the composition of three transformations: $T = T_\mathrm{d} \circ T_\mathrm{s} \circ T_\mathrm{r}$, which we will refer to as displacement, smoothing and restructuring operators, respectively. The displacement operator pushes the deformable mesh towards the object surface while the smoothing operator regularizes the effect of this displacement and the restructuring operator modifies the mesh connectivity to eliminate any geometrical distortions that may appear during surface evolution.

Displacement operator: The distance between the deformable mesh M_k and the object surface S, dropping the index t, can be approximated by the average distance from the vertex set of M_k to the surface:

$$\varepsilon(M_k, S) = \frac{1}{N} \sum_{i=1}^{N_k} d(\mathbf{v}_{i,k}, S), \tag{3}$$

where $\mathbf{v}_{i,k}$ is the position vector of the ith vertex, and $d(\mathbf{v}_{i,k}, S)$ is the Euclidean distance of the vertex to the surface S. To reduce the distance $\varepsilon(M_k, S)$, the operator $T_\mathrm{d}(M_k)$ maps the deformable mesh M_k to M'_k by moving each vertex $\mathbf{v}_{i,k}$ with a displacement $\mathbf{d}(\mathbf{v}_{i,k})$:

$$\mathbf{v}'_{i,k} = \mathbf{v}_{i,k} + \mathbf{d}(\mathbf{v}_{i,k}), \tag{4}$$

where $\{\mathbf{v}'_{i,k}\}$ is the vertex set of the transformed mesh M'_k which has the same connectivity as M_k. The displacement is set to be in the direction of the surface normal $\mathbf{N}(\mathbf{v}_{i,k})$, inwards or outwards, depending on positioning of the vertex with respect to the target surface S:

$$\mathbf{d}(\mathbf{v}_{i,k}) = \delta(\mathbf{v}_{i,k}) \cdot \mathbf{N}(\mathbf{v}_{i,k}). \tag{5}$$

The displacement scalar $\delta(\mathbf{v}_{i,k})$ is computed based on the signed distance from the vertex $\mathbf{v}_{i,k}$ to the target surface S, as will later be explained in Section 4. Also note that the magnitude of the displacement has to be bounded above for a stable surface evolution.

Smoothing operator: The smoothing operator, T_s, is necessary for a robust mesh evolution that is free of topological errors and to have eventually a visually pleasant fair surface representation. It should be easy to compute, yet must not yield any geometrical shrinkage and bias in the final surface estimate. To achieve this, we employ a combination of the classical Laplacian smoothing and Taubin's surface fairing technique [8].

The operator $T_\mathrm{s}(M)$ maps the deformable mesh M to M' by moving each vertex $\mathbf{v}$ to $\mathbf{v}'$ (dropping the vertex index i and the iteration index k):

$$\mathbf{v}' = \mathbf{v} + \Delta\mathbf{v}_\mathrm{T} + \Delta\mathbf{v}_\mathrm{N}, \tag{6}$$

where the displacements $\Delta\mathbf{v}_\mathrm{T}$ and $\Delta\mathbf{v}_\mathrm{N}$ correspond to smoothing along tangential and normal directions of the surface, respectively. We obtain the tangential component, $\Delta\mathbf{v}_\mathrm{T}$, by tangential Laplacian smoothing [9]:

$$\Delta\mathbf{v}_\mathrm{T} = \mathbf{L}(\mathbf{v}) - (\mathbf{L}(\mathbf{v}) \cdot \mathbf{N})\mathbf{N}, \tag{7}$$

where $\mathbf{L}(\mathbf{v})$ denotes the Laplacian displacement that moves the vertex $\mathbf{v}$ to the centroid of the vertices in its one-ring neighborhood. The component $\Delta\mathbf{v}_\mathrm{N}$, on the other hand, is obtained by fairing the surface along its normal direction:

$$\Delta\mathbf{v}_\mathrm{N} = (\mathbf{F}(\mathbf{v}) \cdot \mathbf{N})\mathbf{N}, \tag{8}$$

where $\mathbf{F}(\mathbf{v})$ denotes the displacement created by the non-shrinking surface fairing algorithm described in [8].

Restructuring operator: The restructuring operator, T_r, is the composition of three operators: $T_\mathrm{r} = T_\mathrm{split} \circ T_\mathrm{col} \circ T_\mathrm{flip}$, which are edge split, edge collapse and edge flip transformations introduced in [2] for mesh optimization. We use these elementary transformations in the way [3] uses them for mesh editing. At the end of each iteration of the surface evolution, the operator T_split first splits all edges longer than ε_max at their midpoints. Then, the operator T_col successively eliminates all edges shorter than ε_min by edge collapses. Finally, the flip operator T_flip is applied to reduce the number of irregular vertices possibly created by the previous collapse and split operations. For the split operation to be compatible with the collapse operation, the threshold ε_max has to be chosen such that $\varepsilon_\mathrm{max} \geq 2\varepsilon_\mathrm{min}$ since otherwise split operations would create edges with length smaller than ε_min. We set $\varepsilon_\mathrm{max} = 2\varepsilon_\mathrm{min}$ to have uniformly sized triangles with small aspect ratios. Thanks to the restructuring operation applied at each iteration of the surface evolution, the deformable mesh can adapt its shape to the object surface, avoiding geometrical distortions such as degenerate triangles and irregular vertices.

4 Surface Tracking Algorithm

The mesh representation of the object surface at each frame t, $M_{k^*}^{(t)}$, is reconstructed by deforming the surface reconstructed at the previous frame, $M_0^{(t)} = M_{k^*}^{(t-1)}$. The displacement, $\mathbf{d}(\mathbf{v}_{i,k})$, at each vertex i of the mesh and at each iteration k of the surface evolution, is computed based on the time-varying silhouette information. The magnitude of the displacement, $\delta(\mathbf{v}_{i,k})$, is based on how far and in which direction (inside or outside) the vertex $\mathbf{v}$ is with respect to the silhouettes at that time instant. Thus the displacement scalar δ, which may take negative values as well, is computed by projecting $\mathbf{v}$ onto the image planes and thereby estimating an isolevel value $f(\mathbf{v})$ via bilinear interpolation:

$$\delta(\mathbf{v}) = \varepsilon_\mathrm{min} f(\mathbf{v}) = \varepsilon_\mathrm{min} \min\{G[\mathrm{Proj}_{I_n}(\mathbf{v})] - 0.5\}, \tag{9}$$

where $\mathrm{Proj}_{I_n}(\mathbf{v})$ is the projection of the vertex $\mathbf{v}$ to I_n, the n'th binary silhouette image (0 for outside, 1 for inside) in the sequence. The function G, taking values between 0 and 1, is the bilinear interpolation of the sub-pixelic projection of the vertex $\mathbf{v}$. Thus, the isolevel function $f(\mathbf{v})$ takes on values between -0.5 and 0.5, and the zero crossing of this function reveals the isosurface. The isovalue of the vertex $\mathbf{v}$ is provided by the image of the silhouette that is farthest away from the point, or in other words, where the interpolation function G assumes its minimum value.

We distinguish the vertices of the deformable mesh under three categories with respect to their isovalues. A vertex $\mathbf{v}$ is labeled as IN if $f(\mathbf{v})$ is 0.5, OUT if -0.5 and ON if in-between. According to this definition, ON vertices are those positioned within a narrow band around the boundary surface. By Eq. 9, the displacement at each ON vertex varies within the interval $(-\varepsilon_{min}/2, \varepsilon_{min}/2)$. The vertices which are out of this band are labeled as IN or OUT, depending on whether they are located inside or outside the silhouettes and they have displacement scalars $\varepsilon_{min}/2$ or $-\varepsilon_{min}/2$.

We now introduce the overall algorithm to obtain the sequence of meshes representing the time-varying geometry. The inputs are the multi-viewpoint time-varying silhouette images $\{I_n^{(t)}\}$, the mesh model of the first time frame $M^{(0)}$, and the projection parameters of the multi-camera system:

```
Iterate on $t$
  Set $M_0^{(t)} = M^{(t-1)}$;
  Pose register $M_0^{(t)}$ using the silhouettes $\{I_n^{(t)}\}$;
  Iterate on $k$
    Displace $M_k^{(t)}$ by $T_\mathrm{d}(M_k^{(t)})$;
    Smooth $M_k^{(t)}$ by $T_\mathrm{s}(M_k^{(t)})$;
    Restructure $M_k^{(t)}$ by $T_\mathrm{r}(M_k^{(t)})$;
  Till convergence
  Set $M^{(t)} = M_{k^*}^{(t)}$ as mesh representation of frame $t$;
Till end of scene
```

The resulting mesh sequence, $M^{(0)}, M^{(1)}, ..., M^{(t)}, ...$, representing the time-varying geometry, can be encoded in terms of the small-scale vertex displacements and the restructuring operations along with the initial model and the pose registration parameters of each frame. Since the restructuring operations are applied to edges, an edge index is normally sufficient to encode each operation. On the other hand, the displacements applied to vertices are always significantly smaller in size in comparison to the vertex positions in space. The exact order of the size is dependent on the maximum size of the object of interest, the speed of the motion and the required precision. In Section 5, we will compare the encoding efficiency of representing the mesh sequence by these data instead of using a separate mesh representation at each time instance on our test sequence.

5 Experimental Results

We demonstrate our method on 150 frames of the synthetic "jumping man" mesh sequence [7], which was originally reconstructed from a real scene. Hence this sequence exhibits the realistic motion of the jumping act of a human actor. We have created the time-varying multiview silhouette images

$\varepsilon_{\max}$	Time (sec)	Iterations (#)	Split (#)	Collapse (#)	Flip (#)	Vertices (#)	ε-error
0.021	26	15	601	599	1116	9792	0,0209
0.033	9	11	180	181	345	3750	0,0209
0.045	4	12	101	103	160	1963	0,0208

Table 1: Average value for reconstruction time, numbers of iterations, split operations, collapse operations, flip operations, vertices and the average reconstruction error over the whole sequence with varying resolution $\varepsilon_{\max}$.

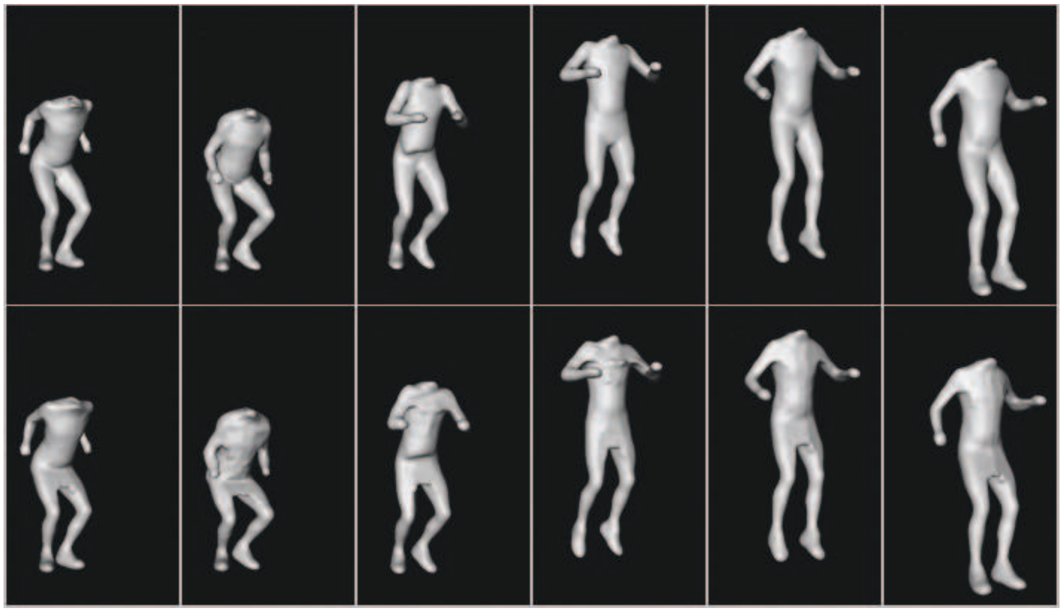

Figure 2: Top row: Frames 60, 65, 70, 75 and 80 of the original jumping man mesh sequence. Bottom row: Corresponding reconstructions.

from the 3D models of this sequence, using an artificial circular camera configuration consisting of 16 cameras modeled with perspective projection. Since we have the ground-truth of the time-varying scene to be reconstructed, we are able to assess quantitatively the error performance of our reconstruction algorithm.

Our reconstruction algorithm based on surface deformation successfully tracks the time-varying geometry through the whole test sequence. In Fig. 2, we display samples from the mesh sequence reconstructed with the proposed technique in comparison to the original mesh sequence. We observe on the reconstructed sequence some discrepancies from the original geometry, which are mainly due to the well-known limitations of the shape from silhouette technique, but the geometry is recovered as smoothly and faithfully as possible from the available information thanks to our smooth deformation framework. In Fig. 5 we plot the normalized error for each of the reconstructed frames. The distance of each reconstructed mesh representation to the original surface, ε, is displayed as normalized to the unit sphere that bounds the geometry of the whole sequence. The observed errors are mostly due to the missing concavities in the silhouette geometry.

In Fig. 4 we illustrate the surface evolution from frame 80 to 81. As the iterations proceed, we observe that the IN and OUT vertices, which are not fit to the surface, tend to vanish. Fig. 3 illustrates the same surface evolution zoomed onto the arm of the body in wireframe overlay. We observe that the deformable mesh maintains a high quality in terms of the aspect ratio of the triangles and smoothness during surface evolution.

To be able to assess the efficiency of the proposed technique, both in reconstruction time and storage, we plot in Fig. 5, the time and the number of restructuring operations needed at each frame of the reconstruction for three different values of the maximum edge length constraint, $\varepsilon_{\max}$. We note that the common peaks at the plotted curves refer to the frames where the local motion of the object is faster. Table 1 provides the averages of these values over the whole sequence along with the average reconstruction error and the average number of vertices. We observe that the efficiency of the technique, both in reconstruction time and representation load, decreases significantly as the value of $\varepsilon_{\max}$ decreases, i.e., as the resolution of the reconstruction increases. We also observe that for these three different values of $\varepsilon_{\max}$, the reconstruction error does not exhibit significant variation. We see that, for $\varepsilon_{\max} = 0.045$, which seems as the optimal resolution for the jumping man, the average reconstruction time

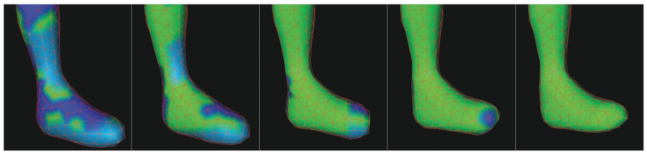

Figure 3: Zoom on the foot at frame 71. Rendered after iterations 1, 6, 12, and 15 (ON, IN and OUT vertices colored green, dark blue and light blue, respectively).

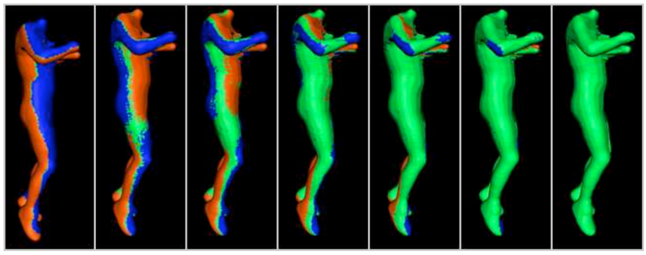

Figure 4: Surface evolution from frame 80 to 81. From left to right: before pose registration, after pose registration, after iterations 2, 7, 13 and 15 (ON, IN and OUT vertices are colored in green, blue and red, respectively).

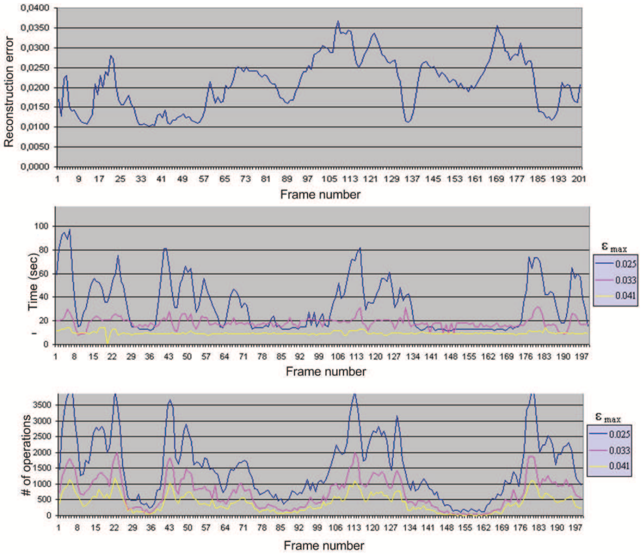

Figure 5: From top to bottom: Reconstruction error, reconstruction time, and number of restructuring operations for each frame with varying resolution $\varepsilon_{\max}$.

is 4 seconds per frame whereas the reconstruction of a single frame from scratch by using the same deformation scheme, starting from the bounding sphere, would take about 90 seconds.

In Table 1, we also observe that the average number of restructuring operations, for $\varepsilon_{\max} = 0.045$, is about 360. Thus the whole time-varying geometry can be encoded on average in terms of 360 restructuring operations and the small-scale displacements of the 1963 vertices per frame, excluding the cost of the initial mesh and the pose registration parameters. In case 12-bit geometrical precision is sufficient, we have observed that each vertex displacement can be encoded using 6 bits for each x, y and z directions. This means that the time-varying mesh representation can be encoded on average approximately by 4.8KB of data per frame, assuming that each restructuring operation can be represented by the index of an edge, that is, with 12 bits. Note that there is no statistical compression involved in computation of this representation load. If each mesh of the sequence were to be independently encoded using a vertex-edge list without employing any compression, then the resulting load would be approximately 12KB per frame, which is about three times of the former.

6 Conclusion

We have presented a computationally efficient surface reconstruction technique that can successfully track the time-varying geometry of a moving object, such as the jumping man, from its multiview silhouettes, using mesh deformation. The resulting representation is both spatially and temporally smooth and space efficient. Since both the connectivity and the geometry of the object can be tracked, our method is applicable also to objects with non-rigid motion. The current drawback of the proposed algorithm is that it may fail to track the surface in case the local motion is too fast on small shape details that can be confused by the silhouette information. For the algorithm to work successfully, there is generally a compromise between the frame rate, the speed of the motion and the size of the shape details. We plan to overcome this problem by incorporating multistereo information to our algorithm, which would not only increase its robustness, but would also increase its efficiency since in this way a faster convergence would be possible with less number of iterations.

Acknowledgements

This work has been supported by TUBITAK under the project EEEAG-105E143 and by the European FP6 Network of Excellence 3DTV (www.3dtv-research.org).

References

[1] J. Carranza, C. Theobalt, M. A. Magnor, and H. Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 22(3):569–577, 2003.

[2] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *Proc. SIGGRAPH'93*, pages 19–26, 1993.

[3] L. P. Kobbelt, T. Bareuther, and H. Seidel. Multiresolution shape deformations for meshes with dynamic vertex connectivity. *Computer Graphics Forum (Eurographics'00)*, 19, 2000.

[4] M. A. Magnor and B. Goldlcke. Spacetime-coherent geometry reconstruction from multiple video streams. *Int. Symp. 3DPVT*, pages 365–372, 2004.

[5] J. Montagnat, H. Delingette, and N. Ayache. A review of deformable surfaces: topology, geometry and deformation. *Image and Vision Computing*, 19:1023–1040, 2001.

[6] K. Mueller, A. Smolic, P. Merkle, M. Kautzner, and T. Wiegand. Coding of 3d meshes and video textures for 3d video objects. *Proc. Picture Coding Symposium*, 2004.

[7] P. Sand, L. McMillan, and J. Popovic. Continuous capture of skin deformation. *Int. Conf. on Computer Graphics and Interactive Techniques*, 2003.

[8] G. Taubin. A signal processing approach to fair surface design. *Proc. SIGGRAPH'95*, pages 315–358, 1995.

[9] Z. J. Wood, P. Schröder, D. Breen, and M. Desbrun. Semi-regular mesh extraction from volumes. *Proc. Visualization'00*, pages 275–282, 2000.