Dynamic 3D Shape From Multi-viewpoint Images Using Deformable Mesh Model

Shohei Nobuhara Takashi Matsuyama
Graduate School of Informatics, Kyoto University
Sakyo, Kyoto, 606-8501, Japan
{nob, tm}@vision.kuee.kyoto-u.ac.jp

Abstract

This paper presents a framework for dynamic 3D shape reconstruction from multi-viewpoint images using a deformable mesh model. With our method, we can obtain 3D shape and 3D motion of the object simultaneously. We represent the shape by a surface mesh model and the motion by translations of its vertices, i.e., deformation. Thus global and local topological structure of the mesh are preserved from frame to frame. This helps us to analyse the motion of the object, to compress the 3D data, and so on. Our model deforms its shape so as to satisfy several constraints. This constraint-based deformation provides a computational framework to integrate several reconstruction cues such as surface texture, silhouette, and motion flow observed in multi-viewpoint images.

1 Introduction

The problem we consider in this paper is how we can reconstruct dynamic 3D shape from multi-viewpoint images, which are attracting many researchers.

3D shape reconstruction is a classic problem in computer vision. In recent methods, multi-viewpoint object images are employed to improve the robustness and accuracy. 3D shape reconstruction methods can be categorized based on 1) the representation of object shape: Volume-based or surface-based and 2) reconstruction cues:stereo-based or silhouette-based. To accomplish more stability and accuracy, recent methods propose frameworks to combine multiple reconstruction cues. For example, Fua[4] represented object shape by 2.5D triangular mesh model and deformed it based on photometric stereo and silhouette constraint. Cross[3] carved visual hull, a set of voxels given by silhouettes, using photometric property.

On the other hand, 3D motion recovery with known object shape is also a classic problem. For example, Heap[5] proposed human hand tracking from camera images using deformable hand model. Bottino[1] tracked 3D human action from multi-viewpoint silhouettes. Vedula[9] introduced a framework to compute dense 3D motion flow from optical flows with/without object shape.

The problem we consider in this paper is *dynamic* 3D shape from multi-viewpoint images, which reconstructs object shape and motion simultaneously. A naive method for this problem would be: Step 1. reconstruct 3D shape for each frame, Step 2. estimate 3D motion by establishing correspondences between a pair 3D shapes at frame t and next frame $t+1$. However, this approach consists of heterogeneous processings. We believe that homogeneous processings, i.e., simultaneous recovery is better than heterogeneous two step method. Toward simultaneous recovery, for example, Vedula[10] showed an algorithm to recover shapes represented by voxels in 2 frames and per-voxel-correspondence between them simultaneously.

1.1 Approach

In this paper, we present a framework for dynamic 3D shape reconstruction from multi-viewpoint images using a deformable mesh model. With our method, we can obtain 3D shape and 3D motion of the object simultaneously.

We represent the shape by a surface mesh model and the motion by translations of its vertices, i.e., deformation. Thus global and local topological structure of the mesh are preserved from frame to frame. This helps us to analyze the motion of the object, to compress the 3D data, and so on. Our model deforms its shape so as to satisfy several constraints, 1) “photo-consistency” constraint, 2) silhouette constraint, 3) smoothness constraint, 4) 3D motion flow constraint, and 5) inertia constraint. We show this constraint-based deformation provides a computational framework to integrate several reconstruction cues such as surface texture, silhouette, and motion flow observed in multi-viewpoint images.

In our 3D deformable mesh model, we introduce two types of deformation: intra-frame deformation and inter-frame deformation. In the intra-frame deformation, our model uses visual hull, a result of volumetric intersection, as initial shape and changes its shape so as to satisfy the constraint 1), 2) and 3) described above. The volumetric intersection estimates rough but stable shape using geometric information, i.e., silhouettes, and the deformable model refines the shape using photometric information. In the inter-frame deformation, our model changes its shape frame by

Figure 1. Input images and visual hull

frame, under all constraints 1), ..., 5). This deforming process enables us to obtain the *topologically consistent* shape in the next frame and *per-vertex*-correspondence information, i.e., motion simultaneously.

2 Input Images And Visual Hull

Figure 1 illustrates input images captured by cameras circumnavigating the object (dancing woman) and visual hull reconstructed by our volumetric intersection method proposed in [11] followed by the discrete marching cubes method[7].

3 Deformable 3D Mesh Model For Intra-frame Deformation

Using our deformable mesh model for the intra-frame deformation, we can employ geometric and photometric constraints of the object surface into its shape reconstruction process, which are not used in the volume intersection method, stereo, or the space carving method[8].

Our algorithm consists of the following steps:

step 1 Convert the visual hull of the object: the voxel representation into the triangle mesh model by the discrete marching cubes algorithm and use it as an initial shape.

step 2 Deform the model iteratively:

- **step 2.1** Compute force working at each vertex respectively.
- **step 2.2** Move each vertex according to the force.
- **step 2.3** Terminate if the vertex motions are small enough. Otherwise go back to 2.1 .

To realize the shape deformation like SNAKES[6], we can use either energy function based or force based methods. As described above, we employed a force based method. This is firstly, from a computational point of view, because we have too many vertices (for example, the mesh model shown in Figure 1 has about 12,000 vertices) to solve energy function and secondly, from an analytical point of view, because one of the constraints used to control the deformation cannot be represented as any analytical energy function (see below).

We employed the following three types of constraints to control the intra-frame deformation:

1. **Photometric constraint**: a patch in the mesh model should be placed so that its texture, which is computed by projecting the patch onto a captured image, should be consistent irrespectively of onto which image it is projected.
2. **Silhouette constraint**: when the mesh model is projected onto an image plane, its 2D silhouette should be coincide with the observed object silhouette on that image plane.
3. **Smoothness constraint**: the 3D mesh should be locally smooth and should not intersect with itself.

These constraints define a *frame and skin model* to represent 3D object shape:

- Suppose we want to model the object in Figure 2 (a).
- First, the silhouette constraint defines a set of frames of the object (Figure 2 (b)).
- Then the smoothness constraint defines a rubber sheet skin to cover the frames (Figure 2 (c)).
- Finally, the photometric constraint defines supporting points on the skin that have prominent textures (Figure 2 (d)).

In what follows, we describe the forces at each vertex generated to satisfy the constraints.

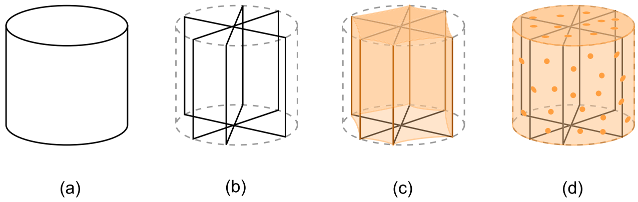


Figure 2. Frame and skin model

3.1 Forces at each Vertex

We denote a vertex, its 3D position, and the set of cameras which can observe that vertex by v, $\boldsymbol{q}_v$, and C_v respectively. For example, $C_v = \{\mathsf{CAM}_2, \mathsf{CAM}_3\}$ in Figure 3.

We introduce the following three forces at v to move its position so that the above mentioned three constraints should be satisfied:

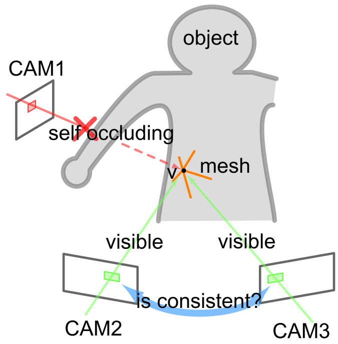


Figure 3. Photometric consistency and visibility

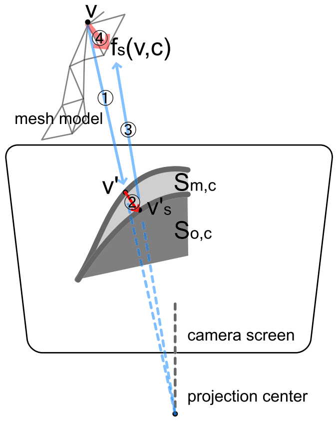


Figure 4. Silhouette preserving force

External Force: $\boldsymbol{F}_e(v)$
First, we define external force $\boldsymbol{F}_e(v)$ to deform the mesh to satisfy the photometric constraint.

$$\boldsymbol{F}_e(v) \equiv \nabla E_e(\boldsymbol{q}_v), \tag{1}$$

where $E_e(\boldsymbol{q}_v)$ denotes the correlation of textures to be mapped around v (Figure 3) :

$$E_e(\boldsymbol{q}_v) \equiv \frac{1}{N(C_v)} \sum_{c \in C_v} \left\| p_{v,c} - \overline{p_v} \right\|^2, \tag{2}$$

where c denotes a camera in C_v, $N(C_v)$ the number of cameras in C_v, $p_{v,c}$ the texture corresponding to v on the image captured by c, and $\overline{p_v}$ the average of $p_{v,c}$. $\boldsymbol{F}_e(v)$ moves v so that its corresponding image textures observed by the cameras in C_v become mutually consistent.
Internal Force: $\boldsymbol{F}_i(v)$
Since $\boldsymbol{F}_e(v)$ may destroy smoothness of the mesh or incur self-intersection, we introduce the internal force $\boldsymbol{F}_i(v)$ at v:

$$\boldsymbol{F}_i(v) \equiv \frac{\sum_j^n \boldsymbol{q}_{v_j} - \boldsymbol{q}_v}{n}, \tag{3}$$

where $\boldsymbol{q}_{v_j}$ denotes neighboring vertices of v and n the number of such vertices. $\boldsymbol{F}_i(v)$ works like tension between vertices and keeps them locally smooth.

Note that this simple internal force tends to shrink the mesh model. This is suitable for the intra-frame deformation because it starts with visual hull which encages the real object shape. However, in the inter-frame deformation, we redefine internal force as combination of gravity and repulsion between linked vertices as $\boldsymbol{F}_i(v)$, and add *diffusion* step just after step 2.1.
Silhouette Preserving Force: $\boldsymbol{F}_s(v)$
To satisfy the silhouette constraint described before, we introduce the silhouette preserving force $\boldsymbol{F}_s(v)$. This is the most distinguishing characteristics of our deformable model and involves nonlinear selective operation based on the global shape of the mesh, which cannot be analytically represented by any energy function.

Figure 4 explains how this force at v is computed, where $S_{o,c}$ denotes the object silhouette observed by camera c, $S_{m,c}$ the 2D projection of the 3D mesh on the image plane of camera c, and v' the 2D projection of v on the image plane of camera c.

1. For each c in C_v, compute the partial silhouette preserving force $\boldsymbol{f}_s(v,c)$ by the following method.
2. If
   (a) v' is located out of $S_{o,c}$ or
   (b) v' is located in $S_{o,c}$ and on the contour of $S_{m,c}$,

   then compute the 2D shortest vector from v' to $S_{o,c}$ (Figure 4 ②) and set its corresponding 3D vector = $\boldsymbol{f}_s(v,c)$ (Figure 4 ④).
3. Otherwise, $\boldsymbol{f}_s(v,c) = 0$.

The overall silhouette preserving force at v is computed by summing up $\boldsymbol{f}_s(v,c)$:

$$\boldsymbol{F}_s(v) \equiv \sum_{c \in C_v} \boldsymbol{f}_s(v,c). \tag{4}$$

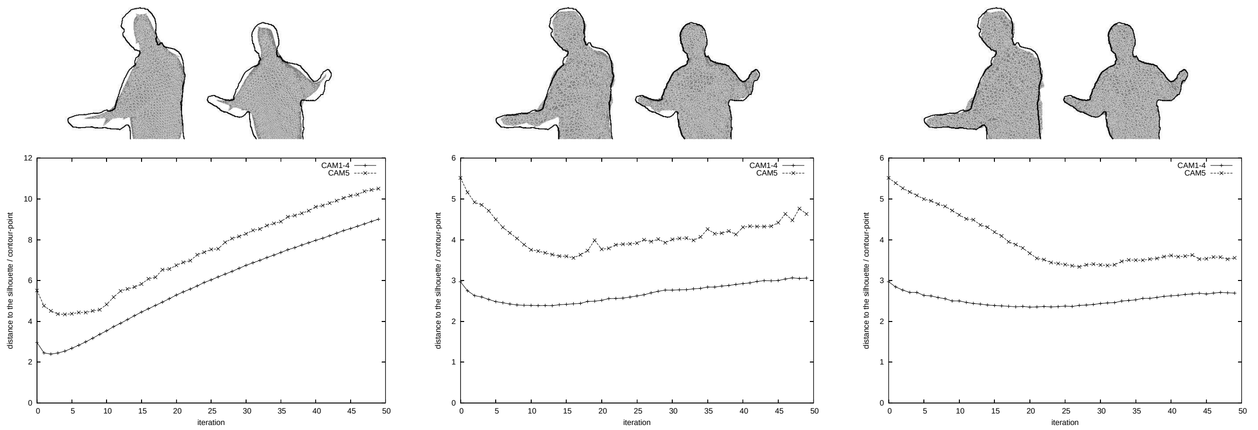
Note that $\boldsymbol{F}_s(v)$ works only at those vertices that are located around the object contour generator[3], which is defined based on the global 3D shape of the object as well as locations of image planes of the cameras.
Overall Vertex Force: $\boldsymbol{F}(v)$
Finally we define vertex force $\boldsymbol{F}(v)$ with coefficients α, β, γ as follows:

$$\boldsymbol{F}(v) \equiv \alpha \boldsymbol{F}_i(v) + \beta \boldsymbol{F}_e(v) + \gamma \boldsymbol{F}_s(v). \tag{5}$$

$\boldsymbol{F}_e(v)$ and $\boldsymbol{F}_s(v)$ work to reconstruct the accurate object shape and $\boldsymbol{F}_i(v)$ to smooth and interpolate the shape. Note that there may be some vertices where $C_v = \{\}$ and hence $\boldsymbol{F}_e(v) = \boldsymbol{F}_s(v) = 0$.

3.2 Performance Evaluation

Figure 5 illustrates the camera arrangement for experiments, where we use $\mathrm{CAM}_1, \ldots, \mathrm{CAM}_4$ for the shape reconstruction and CAM_5 for the performance evaluation. That



(a) (b) (c)

Figure 7. Experimental results. (a)$\boldsymbol{F}_i(v)$ alone ($\alpha = 1.0, \beta = 0.0, \gamma = 0.0$), (b)$\boldsymbol{F}_i(v) + \boldsymbol{F}_s(v)$ ($\alpha = 0.5, \beta = 0.0, \gamma = 0.5$) (c)$\boldsymbol{F}_i(v) + \boldsymbol{F}_e(v) + \boldsymbol{F}_s(v)$ ($\alpha = 0.3, \beta = 0.4, \gamma = 0.3$)

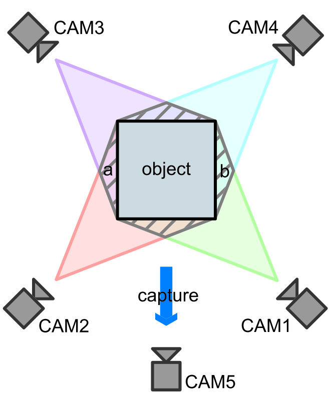
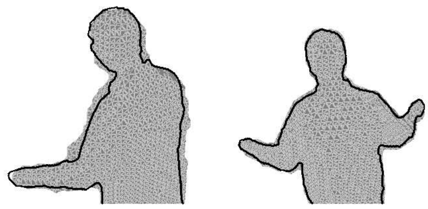


Figure 5. Camera arrangement



(a) CAM5 (b) CAM1

Figure 6. Initial shape. (a) is viewed from CAM$_5$ **in Figure 5, (b) from** CAM$_1$

is, we compare the 2D silhouette of the reconstructed shape viewed from CAM$_5$ position with the really observed one by CAM$_5$.

Figure 6 shows the initial object shape by the volume intersection using the images captured by CAM$_1$, ..., CAM$_4$. The shape is viewed from CAM$_5$ and CAM$_1$ positions, i.e. the shaded regions show $S_{m,5}$ and $S_{m,1}$, respectively. Bold lines in the figures highlight the contours of $S_{o,5}$ and $S_{o,1}$. We can observe some differences between $S_{o,5}$ and $S_{m,5}$ while not between $S_{o,1}$ and $S_{m,1}$. This is because the image captured by CAM$_5$ is used for the reconstruction.

In the experiments, we evaluated our algorithm with the following conditions : (a) $\boldsymbol{F}(v) = \boldsymbol{F}_i(v)$, (b) $\boldsymbol{F}(v) = \boldsymbol{F}_i(v) + \boldsymbol{F}_s(v)$, (c) $\boldsymbol{F}(v) = \boldsymbol{F}_i(v) + \boldsymbol{F}_e(v) + \boldsymbol{F}_s(v)$. The first row of Figure 7 illustrates $S_{m,5}$ (left) and $S_{m,1}$ (right) for each condition associated with the bold lines denoting the corresponding observed object silhouette contours: $S_{o,5}$ and $S_{o,1}$. The graphs on the second row show how the average error between $S_{m,c}$ and $S_{o,c}$ $c = 1, 5$ changes with the iterative shape deformation.

From these results we can get the following observations:

- With $\boldsymbol{F}_i(v)$ alone (Figure 7(a)), the mesh model shrinks and its 2D silhouette on each image plane becomes far apart from the observed one.

- With $\boldsymbol{F}_i(v)$ and $\boldsymbol{F}_s(v)$, while $S_{m,c}, c = \{1 \ldots 4\}$ well match with $S_{m,c}$, $S_{m,5}$, whose corresponding image is not used for the reconstruction, does not deform well (Figure 7(b)).

- With $\boldsymbol{F}_i(v)$, $\boldsymbol{F}_e(v)$, and $\boldsymbol{F}_s(v)$, $S_{m,5}$ matches well with $S_{o,5}$ (Figure 7(c)). This proves the effectiveness of $\boldsymbol{F}_e(v)$.

We also evaluated the mesh model with synthetic objects defined by super quadric functions. Although these objects had concave portions, the mesh model could reconstruct such concavity.

Compared with the Space-Carving method[8], which employs photometric consistency as its main reconstruction cue, our approach additionally employs geometric continuity and silhouette constraint. Such rich constraints make our approach more stable and accurate. Moreover, our deformable mesh model can be extended to dynamic inter-frame deformation, which will enable us to analyze dynamic object motion and realize highly efficient data compression. The next section describes this inter-frame deformation algorithm.

4 Dynamic Shape Recovery Using Deformable 3D Mesh Model

If a model at t deform its shape to satisfy the constraints at $t+1$, we can obtain shape at $t+1$ and motion from t to $t+1$ simultaneously.

Clearly, the constraints used in the intra-frame deformation should be satisfied and would be enough if we had rich texture information all over the object surface. To make the dynamic deformation process more stable, we first modify the photometric constraint:

1. **Photometric constraint(modified)**: textures of each patch in the multi-viewpoint images should be consistent in the frames at both t and $t+1$.

and then introduce additional constraints:

4. **3D Motion flow constraint**: a mesh vertex should drift in the direction of the motion flow of its vicinity.
5. **Inertia constraint**: the motion of a vertex should be smooth and continuous.

Drift Force: $\boldsymbol{F}_d(v_t)$
As described in section 3, we assume that we have silhouette images and a visual hull at each frame. With these visual hulls, i.e., sets of voxels, we can compute rough correspondences between them by the point-set-deformation algorithm[2]. This algorithm gives us the voxel-wise correspondence flow from V_t, voxel set at t, to V_{t+1}, voxel set at $t+1$. We can represent this flow by a set of correspondence lines:

$$L_t = \{l_i |\ i = 1, \ldots, N(V_t)\}, \tag{6}$$

where l_i denotes the correspondence line starting from ith voxel in V_t and $N(V_t)$ the number of voxels in V_t. Whereas visual hulls do not represent accurate object shapes, we can use this correspondence as roughly estimated motion flow.

Once the motion flow is obtained, we define the potential field $E_d(v_t)$ with this flow. First, let v_t denotes a vertex of the mesh at t, $\boldsymbol{q}_{v_t}$ the position of v_t, l_{v_t} the closest correspondence line in L_t from a vertex v_t, $\boldsymbol{p}_{l_{v_t},v_t}$ the closest point on l_{v_t} from v_t, and $\boldsymbol{s}_{l_{v_t}}$ the stating point of the correspondence line l_{v_t}. Then, we define the potential field as the function of the distance from v_t to l_{v_t} and the distance from $\boldsymbol{s}_{l_{v_t}}$ to $\boldsymbol{p}_{l_{v_t},v_t}$:

$$E_d(\boldsymbol{q}_{v_t}) \equiv \|\boldsymbol{s}_{l_{v_t}} - \boldsymbol{p}_{l_{v_t},v_t}\|^2 - \|\boldsymbol{q}_{v_t} - \boldsymbol{p}_{l_{v_t},v_t}\|^2. \tag{7}$$

Finally, we define the drift force $\boldsymbol{F}_d(v_t)$ at vertex v_t was the gradient vector of $E_d(\boldsymbol{q}_{v_t})$:

$$\boldsymbol{F}_d(v_t) \equiv \nabla E_d(\boldsymbol{q}_{v_t}). \tag{8}$$

Inertia Force: $\boldsymbol{F}_n(v_t)$
If we can assume that the interval between successive frames is short enough, we can expect the continuity and the smoothness of the object motion. This assumption tells us that we can predict a vertex location at $t+1$ from its motion history.

We can represent such predictions as a set of prediction lines connecting $\boldsymbol{q}_{v_t}$ and $\hat{\boldsymbol{q}}_{v_{t+1}}$, where $\hat{\boldsymbol{q}}_{v_{t+1}}$ denotes the predicted location of v_{t+1}. Then we can define the inertia force $\boldsymbol{F}_n(v_t)$ just in the same way as the drift force $\boldsymbol{F}_d(v_t)$:

$$\boldsymbol{F}_n(v_t) \equiv \nabla E_n(\boldsymbol{q}_{v_t}), \tag{9}$$

where $E_n(\boldsymbol{q}_{v_t})$ denotes potential field defined based on the set of prediction lines.

Overall Vertex Force: $\boldsymbol{F}(v_t)$
Finally we define vertex force $\boldsymbol{F}(v_t)$ with coefficients $\alpha, \beta, \gamma, \delta, \epsilon$ as follows:

$$\begin{aligned}\boldsymbol{F}(v_t) \equiv\ & \alpha \boldsymbol{F}_i(v_t) + \beta \boldsymbol{F}_e(v_t) + \gamma \boldsymbol{F}_s(v_t) \\ & + \delta \boldsymbol{F}_d(v_t) + \epsilon \boldsymbol{F}_n(v_t).\end{aligned} \tag{10}$$

Each vertex of the mesh is moved from $\boldsymbol{q}_{v_t}$ according to $\boldsymbol{F}(v_t)$ to get $\boldsymbol{q}_{v_{t+1}}$.

4.1 Experimental Results

Figure 8 illustrates the deformation result through 3 successive frames. Left, center and right columns of this figure illustrate real images, visual hulls generated by the discrete marching cubes method at each frame, and mesh models deformed frame by frame respectively. Note that we used the visual hull at frame t as the initial shape of the mesh model, and did not apply the intra-frame deformation in order to evaluate the inter-frame deformation exclusively.

From this result, we can observe:

- Our mesh model deforms non-rigidly.
- From topological point of view, although results of the marching cubes (center column) are globally (as entire mesh) consistent but locally (for each vertex) inconsistent, our mesh model (right column) preserves both global and local topological structure.

5 Conclusion

In this paper, we proposed a computational framework using a deformable mesh model to reconstruct dynamic 3D shape, i.e., full 3D shapes and motions simultaneously. We believe that simultaneous recovery approach, i.e., homogeneous processing scheme is better than the heterogeneous approach which recovers shapes first and then recovers motions from the shapes.

Our constraint-based deformable mesh model scheme realizes 1) integration of several reconstruction cues as deformation constraints and 2) seamless extension from the intra-frame deformation to the inter-frame deformation by

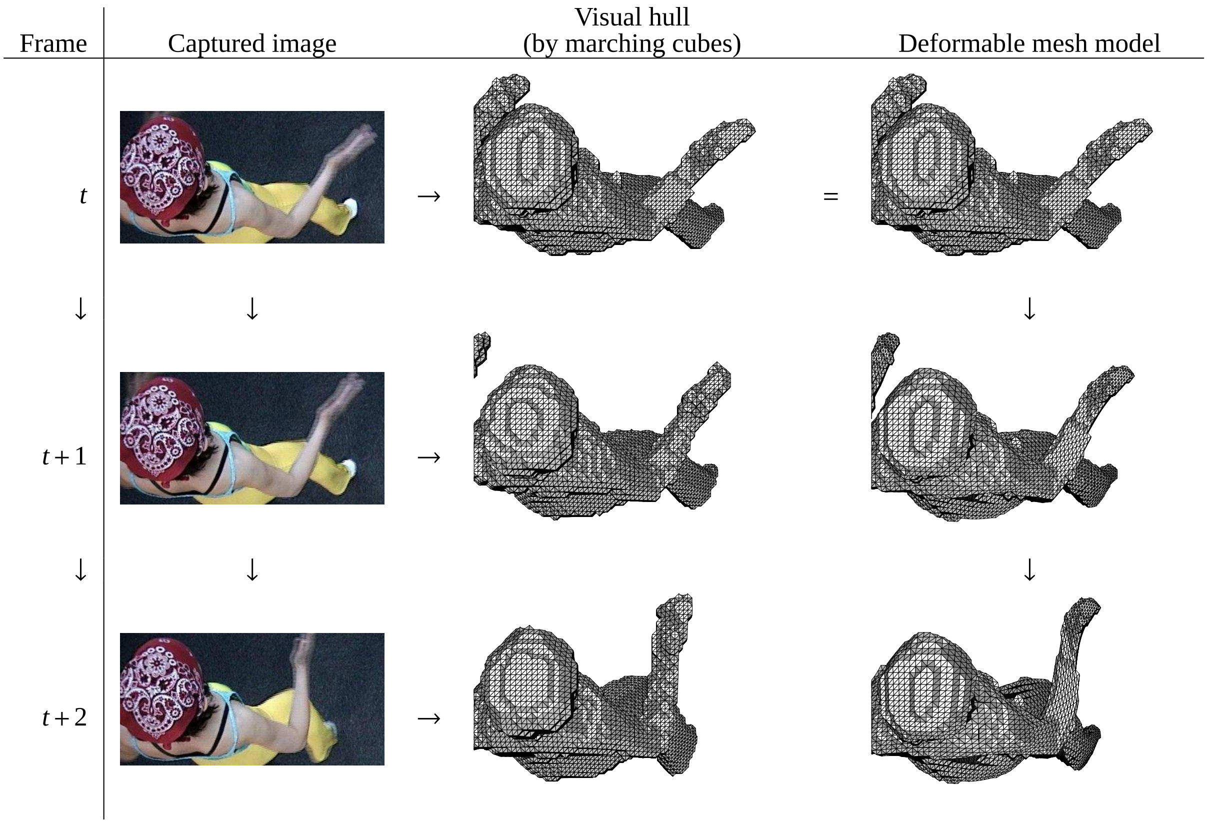


Figure 8. Successive deformation results

adding inter-frame specific constraints. We presented the design of our deformable mesh model and evaluated it in the intra-frame and the inter-frame deformation.

Using our computational framework, we will be able to develop more effective 3D motion analysis, 3D data compression, and so on.

Acknowledgements

This work was supported by the grant-in-aid for scientific research (A) 13308017. We are grateful to Real World Computing Partnership, Japan for allowing us to use their multi-viewpoint video data.

References

[1] A. Bottino and A. Laurentini. A silhouette-based technique for the reconstruction of human movement. *Computer Graphics and Image Processing*, 83:79–95, 2001.

[2] D. Burr. A dynamic model for image registration. *Computer Graphics and Image Processing*, 15:102–112, 1981.

[3] G. Cross and A. Zisserman. Surface reconstruction from multiple views using apparent contours and surface texture, 2000.

[4] P. Fua and Y. G. Leclerc. Using 3-dimensional meshes to combine image-based and geometry-based constraints. In *ECCV (2)*, pages 281–291, 1994.

[5] T. Heap and D. Hogg. Towards 3d hand tracking using a deformable model. In *Proc. of 2nd International Conference on Automatic Face and Gesture Recognition*, pages 140–145, 1996.

[6] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[7] Y. Kenmochi, K. Kotani, and A. Imiya. Marching cubes method with connectivity. In *Proc. of 1999 International Conference on Image Processing*, pages 361–365, Kobe, Japan, Oct. 1999.

[8] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *Proc. of International Conference on Computer Vision*, pages 307–314, Kerkyra, Greece, Sept. 1999.

[9] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *Proceedings of the 7th International Conference on Computer Vision*, volume 2, pages 722 – 729, Sept. 1999.

[10] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6d. In *Computer Vision and Pattern Recognition (CVPR)*, June 2000.

[11] X. Wu and T. Matsuyama. Real-time active 3d shape reconstruction for 3d video. In *this proccdings*, Roma, Italy, Sept. 2003.