4D Mesh Reconstruction from Time-Varying Voxelized Geometry through ARAP Tracking

L. Blache[1], M. Desbrun[2], C. Loscos[1], and L. Lucas[1]

[1]University of Reims Champagne-Ardenne
[2]Caltech

Abstract

We present a method to derive a time-evolving triangle mesh representation from a sequence of binary volumetric data representing an arbitrary motion. Multi-view reconstruction studios use a multiple camera set to turn an actor's performance into a time series of visual hulls. However, the reconstructed sequence lacks temporal coherence as each frame is generated independently, preventing easy post-production editing with off-the-shelf modeling tools. We propose an automated tracking approach to convert the raw input sequence into a single, animated mesh. An initial mesh is globally deformed via as-rigid-as-possible, detail-preserving transformations guided by a motion flow estimated from consecutive frames. Local optimization is added to better match the mesh surface to the current visual hull, leading to a robust 4D mesh reconstruction.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

1. Introduction

The entertainment industry increasingly relies on a mix of real pictures and computer-generated images. While advanced technologies such as motion capture or matte painting are widely used, new methods of *multi-view reconstruction* are now emerging. These methods use a set of multi-viewpoint cameras in an indoor studio to generate an animated 3D model of an actor's performance, without the traditional markers used in motion capture techniques. Most multi-view reconstruction studios use a *model-free* approach to generate a 3D object for each frame of the multi-view video sequence. The resulting series of static poses are not well-suited for subsequent editing as they are devoid of any temporal coherency. Our goal is to generate a time-evolving triangle mesh representing the motion of a single actor, for now, in the capture sequence. Our ultimate goal, however, is to be sufficiently robust to handle complex scenes, involving one or several actors wearing costumes and accessories, or even animals. This requirement prevents the use of most existing methods that assume rigidly articulated models.

Model-based multi-view reconstruction approaches use a template model representing an actor, typically, an articulated mesh of a generic human body [VBMP08, GSdA*09]. The multi-view reconstruction is then achieved by deforming this template in time according to a set of directives (optical flow or silhouette matching) extracted from the multi-viewpoint videos. De Aguiar *et al.* [dAST*08] use a similar approach with a tetrahedral mesh reconstructed from a 3D scan acquisition. The main advantage of this family of techniques is the temporally-coherent animation they produce. Nevertheless, the use of a template model restricts their generality. Cagniart *et al.* [CBI10] employ a deformable surface, initialized with the first frame of the sequence. Following the same kind of approach, we will use the first pose of the sequence as our initial mesh that we then deform based on the rest of the sequence. A mesh-tracking method [SH07, VZBH08, PLBF11] can match several meshes according to curvature or texture criteria, from which one can compute the motion flow describing the movements of an actor between two frames. In our case, however, the visual hull reconstruction usually create significant artifacts which prevent us from using such a mesh-tracking approach directly. Instead, we can use a volumetric approach to compute a motion flow based on a voxel matching algorithm applied to the input sequence [NM04, BLNL14]. Once the motion flow is determined, we animate the initial mesh of the sequence. *Skeleton-based* animation techniques have been shown especially efficient for motion capture. However, they can-

DOI: 10.2312/egsh.20151002

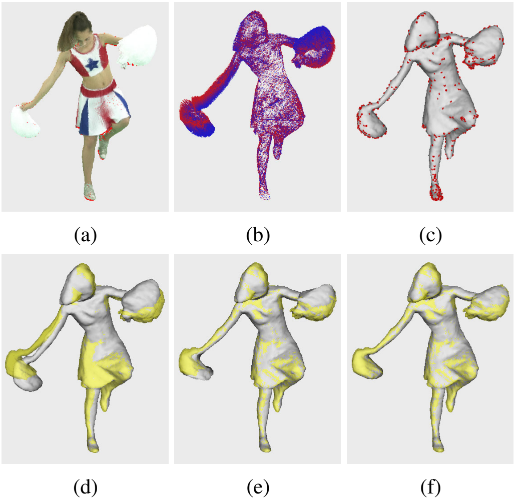


Figure 1: **Our approach at a glance.** Starting from a sequence of volumes (a), we first compute a motion flow (b); we then sample a set of anchor vertices (c) on the current mesh (in grey). To deform the current mesh towards the next visual hull in yellow (d), we perform a global deformation of the mesh based on the displacements of the anchors (e), before applying a local optimization on the mesh to finely match the target visual hull (f).

not handle complex, non-rigid motions. *As-rigid-as-possible* (ARAP) surface modeling [SA07] offer an interesting alternative allowing to globally deform a mesh with fixed connectivity with local geometric transformations that are as close to rigid as possible. A set of anchor points drives the global deformation of the mesh through a non-linear, but simple energy minimization.

The method we propose is based on a template mesh derived from the first frame of the sequence of captured volumes, leading to an automatic initialization. A motion flow is then computed through a voxel matching method (Section 2), more adapted to our noisy input data than a mesh-tracking approach. An anchor-guided ARAP deformation of the mesh is performed, complying as much as possible to the resulting motion flow (Section 3). This reliably finds a good match for the next pose even for an inaccurate estimation of the motion. Finally, a local optimization step adjusts the vertex positions of the mesh to capture *non-rigid shape variations* of the visual hull (Section 4) missed by the ARAP step. After this two-steps mesh animation the template mesh closely fits the visual hull, including details, like the clothes folds, which could not be properly recovered with a classical articulated model-based approach.

2. Motion extraction

Our input is a sequence of n digital volumes obtained by a silhouette-based reconstruction from multi-view video. The i-th volume of this sequence represents the actor at time t_i. The reconstructed volumes we use are 3D grids of binary voxels. Voxels straddling the surface are assigned a RGB color as well, based on the input video. We also compute a volumetric Euclidean distance transform (*EDT*) for each frame. In order to extract a motion flow from this set of volumetric images, we begin with the method described in [BLNL14] to compute voxel matching based both on local geometry and color between consecutive poses. We immerse the binary volume at t_0 in the EDT grid at t_1, so that the EDT value associated to each surface voxel at time t_0 represents its distance to the next pose at time t_1. This distance is used to automatically fix the search radius for the voxel matching step at time t_0. Then, each surface voxel at t_i uses the motion vector estimated between t_{i-1} and t_i to predict the position of the matching voxel in t_{i+1}. We then search for the best matching voxel in t_{i+1} using a small radius around the predicted position, offering a trivial but highly efficient speedup of 60% compared to the original method [BLNL14].

3. Mesh animation

We construct an initial (template) mesh based on the first volume of the sequence which then needs to be advected in the motion flow. We proceed in two main steps. First, a global deformation of the mesh derived from a sparse set of anchors will ensure robustness of the mesh deformation even in the presence of a noisy motion flow. Second, a local adjustment (Section 4) will improve both mesh quality and match between surface and volumetric input data. Note that using these two steps brings robustness: the first step adds resilience to noise by catching the most rigid parts of the deformation, while the second step allows for non-rigid deformation—and will thus remove the potential geometric features present at the initial time which should not be kept "as is" in time (for instance, a wrinkle on a dress).

Anchors' selection. We select a vertex at time t_i to be an *anchor* point if it belongs to a high curvature part of the surface (computed using the method described in [PGK02]) and/or if its associated motion vector is significant (*i.e.* its matched voxel have a high correspondence as defined by the matching function described in [BLNL14]). The sum of these two normalized criteria gives a score to each vertex. We then select a subset of points, corresponding to the highest scores. These points define, via the associated motion vectors, a sparse set of anchors for the mesh deformation step (Section 3), using the scores as weights w_{a_i}. Figure 1c illustrates how these anchors effectively capture shape extremities, prominent edges, and articulations.

Animation. The motion flow gives us the global deformation we have to apply on the template, but not a precise motion to apply on each vertex (see Figure 2c). Thus, we only use it to guide the motion of the reduced set of sampled anchors. In order to be resilient to noisy motion flows, we use a variational method by searching for a deformed mesh M' with locally rigid transformations, while retaining as much

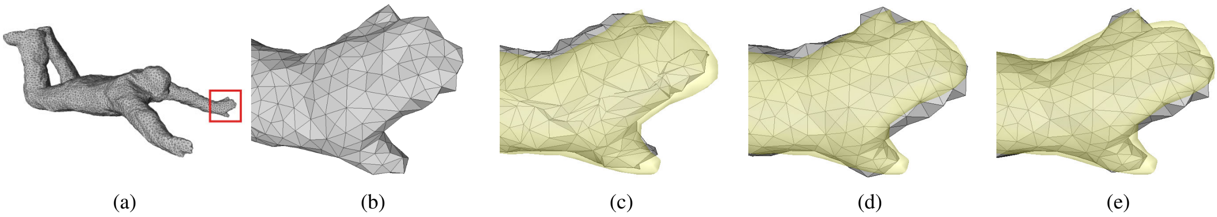


Figure 2: **Comparisons.** Several mesh deformation algorithms applied to the *astronaut* sequence (a). Template mesh at the initial pose (b). Deformed template (grey mesh) by directly applying the motion vectors computed using [BLNL14] (c). Deformed template using the ARAP technique [SA07] without anchors (d). Deformed template using our anchor-based ARAP method (e). The visual hull of the next pose is displayed in yellow.

as possible the final positions of anchor points (Figure 1e). We minimize the following energy:

$$E(M') = E_{ARAP}(M') + E_{ANC}(M'),$$

where E_{ARAP} is the *As-Rigid-As-Possible* energy

$$E_{ARAP}(M') = \sum_{i=1}^{n} \sum_{j \in N(i)} w_{ij} \left\| (p'_i - p'_j) - R_i(p_i - p_j) \right\|^2, \tag{1}$$

with $N(i)$ denoting the one-ring neighborhood of i, and E_{ANC} is a quadratic energy measuring the error in the displacement of anchors

$$E_{ANC}(M') = \sum_{i=1}^{n} w_{a_i} \left\| p'_i - p_i \right\|^2, \tag{2}$$

where the weight w_{a_i} represents the degree of confidence given to an anchor point, described in anchors' selection. We fix $w_{a_i} = 0$ if the vertex p_i does not belong to the set of anchors. The optimality condition for the minimum of our energy basically mirrors the result of [SA07], to which terms coming from the quadratic form (2) are added. That is, the optimal positions p' must satisfy:

$$\sum_{j \in N(i)} w_{ij}(p'_i - p'_j) + w_{a_i} p'_i = \sum_{j \in N(i)} \frac{w_{ij}}{2} (R_i - R_j)(p_i - p_j) + w_{a_i} p_i \tag{3}$$

where R_i is a local rotation best matching p_i and its one ring to p'_i. The global deformation is thus computed by iteratively solving a linear system and an optimal set of rotation matrices: we begin by computing the set of $\{p'_i\}_i$ that satisfy the optimal condition for a fixed set of initial rotations $\{R_i\}_i$ by solving a linear system of the form: $Lp' = b$, where L corresponds to the Laplacian operator applied to the mesh M' in which we add the w_{a_i} weights related to each anchor point (see eq (2)) on the diagonal. The column matrix b contains the righthand side of the expression (3). Optimal rotations R_i are computed using an SVD from the positions of p_i and p'_i as derived in [SA07]. These two steps are repeated until convergence.

4. Local optimization

While the global deformation step provides a robust way to match the next pose, details due to non-rigid deformation (such as cloth folds or hair) will be missed. Moreover, mesh quality may also degrade over time as large deformation happens, making mesh regularization desirable. We thus compute local vertex displacements based on both fitting accuracy and regularization as follows.

Regularization. We regularize the mesh by applying spring-like forces to favor equi-length edges:

$$f_r(p_i) = \sum_{j \in N(i)} (\|p_i - p_j\| - \bar{r}_i) \frac{p_i - p_j}{\|p_i - p_j\|}$$

where p_j is a vertex from the one-ring neighborhood of p_i, while the rest length $\bar{r}_i$ is set to the current average length of the edges adjacent to p_i. We use only the tangential component of the resulting vector.

Silhouette fitting. Using the EDT, we also push each vertex toward the visual hull surface by adding the following force:

$$f_s(p_i) = \vec{n_{p_i}} \cdot EDT(p_i)$$

with $\vec{n_{p_i}}$ and $EDT(p_i)$ being the normal vector and the EDT value at p_i, respectively.

We integrate the sum of these two forces over 250 time steps by updating position and velocity of each vertex (assumed to be all of unit mass) using a simple Runge-Kutta explicit integrator to make the integration trivially parallelizable. Weighting the two forces further allows the user to control regularization vs shape fitting depending on the noise present in the volume sequence.

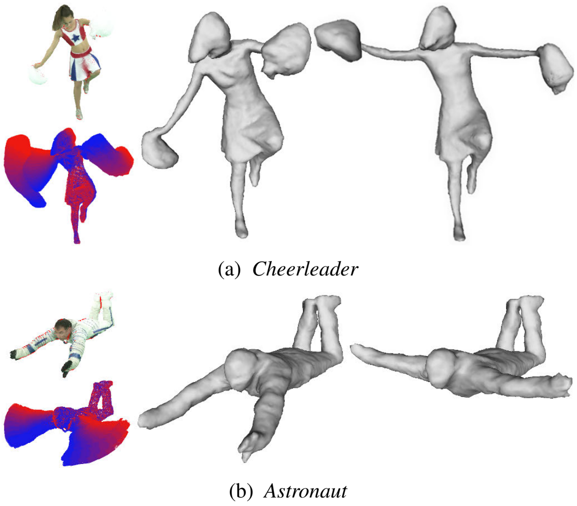
5. Preliminary results and discussion

We tested our method on two datasets obtained through volumetric visual hull reconstruction, from an indoor studio shoot using a 24-camera rig, with an average 180 × 270 × 170 voxel resolution. The template mesh contains 19234 vertices for the *cheerleader* sequence and 8048 for the *astronaut* sequence. We used 10% of the vertices as anchor points. The mesh deformation algorithm (Section 3) needs an average of 200 iterations to reach convergence. Mesh adjustment (Section 4) was applied with weights 0.6 and 0.4 for regularization vs. fitting. The total computational time, from motion estimation to mesh optimization, reaches an average of 130s per frame, on an Intel Core i7 laptop. Results from these two sequences, staying consistents during

15 frames, are presented in Figure 3, demonstrating robustness of our approach given the coarseness of the input volumes. It should also be noticed that our use of weights based on the reliability of the anchors nicely extends the ARAP modeling technique, rendering it particularly robust to the inherent noise present in the motion flow. Figure 2 demonstrates the benefit of our mesh animation approach described in Section 3 (see Figure 2e) compared to a simple advection of the vertices by the motion flow (see Figure 2c) as in [BLNL14], or a pure ARAP method (see Figure 2d). The final local optimization step (Section 4) adapts the mesh to the non-rigid part of the motion, allowing to recover details in clothes and accessories. The *cheerleader* dateset shows that shape of the pom-poms is correctly adjusted after the global deformation phase (see Figures 1e and 1f). This last step also leads to an adaptation of the template during the sequence, avoiding some of the model-based inconveniences, as in [dAST*08], where the tracked model retains some of the surface details (clothing folds) from the initial pose during the whole sequence. In several extreme cases, our local regularization step may degrade some of the details of the characters including very thin features, as they are of a size too close to the size of a grid element. A local optimization with subgrid accuracy could solve this issue. Our approach also assumes that the topology of the first frame of the input data is kept throughout the sequence. However, changes in the topology of the visual hull could occur in the captured sequence, possibly due to occultations if not enough view angles are available. Currently, these events are not supported by our system, but could be handled through, for instance, the method proposed by Letouzey and Boyer [LB12].

6. Conclusion

We presented a new approach for generating a time-evolving mesh from a sequence of volumetric data representing an unstructured motion. After computing a motion flow of the sequence, a template mesh generated by a reconstruction of the first volume is deformed via as-rigid-as-possible, detail-preserving transformations guided by the motion flow and based on a sparse set of weighted anchors. A final local optimization adjusts the mesh, leading to a robust, temporally-consistent mesh reconstruction of the motion.



(a) *Cheerleader*

(b) *Astronaut*

Figure 3: **Results.** Left to right: first frame of the volumetric visual hull sequence and the extracted motion flows through the sequence (motion vectors oriented from blue to red), the initial mesh, and the result of the mesh animation.

References

[BLNL14] BLACHE L., LOSCOS C., NOCENT O., LUCAS L.: 3d volume matching for mesh animation of moving actors. In *Eurographics Workshop on 3D Object Retrieval, 3DOR* (Apr. 2014), pp. 69–76. 1, 2, 3, 4

[CBI10] CAGNIART C., BOYER E., ILIC S.: Probabilistic deformable surface tracking from multiple videos. In *11th European conference on Computer vision (ECCV)* (2010), vol. 6314 of *Lecture Notes in Computer Science*, pp. 326–339. 1

[dAST*08] DE AGUIAR E., STOLL C., THEOBALT C., AHMED N., SEIDEL H.-P., THRUN S.: Performance capture from sparse multi-view video. *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 98:1–98:10. 1, 4

[GSdA*09] GALL J., STOLL C., DE AGUIAR E., THEOBALT C., ROSENHAHN B., SEIDEL H.-P.: Motion capture using joint skeleton tracking and surface estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2009), pp. 1746–1753. 1

[LB12] LETOUZEY A., BOYER E.: Progressive shape models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2012), pp. 190–197. 4

[NM04] NOBUHARA S., MATSUYAMA T.: Heterogeneous deformation model for 3D shape and motion recovery from multiviewpoint images. In *Proceedings of the 2nd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)* (Sept. 2004), pp. 566–573. 1

[PGK02] PAULY M., GROSS M., KOBBELT L.: Efficient simplification of point-sampled surfaces. In *IEEE Visualization, VIS* (Nov. 2002), pp. 163–170. 2

[PLBF11] PETIT B., LETOUZEY A., BOYER E., FRANCO J.-S.: Surface flow from visual cues. In *International Workshop on Vision, Modeling and Visualization (VMV)* (Oct. 2011), pp. 1–8. 1

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, SGP* (2007), pp. 109–116. 2, 3

[SH07] STARCK J., HILTON A.: Correspondence labelling for wide-timeframe free-form surface matching. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (Oct. 2007), pp. 1–8. 1

[VBMP08] VLASIC D., BARAN I., MATUSIK W., POPOVIĆ J.: Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 97:1–97:9. 1

[VZBH08] VARANASI K., ZAHARESCU A., BOYER E., HORAUD R.: Temporal surface tracking using mesh evolution. In *10th European Conference on Computer Vision (ECCV)* (2008), vol. 5303 of *Lecture Notes in Computer Science*, pp. 30–43. 1