COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2014; **25**:447–455
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1577

SPECIAL ISSUE PAPER

Adaptive skeleton-driven cages for mesh sequences

Xue Chen and Jieqing Feng*

State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China

ABSTRACT

The design of a compact and effective representation is a critical step in fully utilizing the abundant raw mesh-based animation resources. In this paper, we present a high-level control structure for animated mesh sequences referred to as an adaptive skeleton-driven cage representation. This approach combines a skeleton and cage to express time-varying shape details and offer flexible control for rigid limb motions. The initial skeleton is inferred by sketching on the rest pose mesh. The corresponding cage is constructed by using an adaptive cross-section-based method. Then, the initial skeleton and cage are propagated to the other meshes in the sequence. Additionally, the generated cage sequence can be automatically refined to improve the mesh reconstruction quality. Our results and comparisons demonstrate that the proposed approach is an intuitive, compact, and efficient representation. We demonstrate its potential through a variety of applications, such as animation compression, deformation transfer and pose editing. Copyright © 2014 John Wiley & Sons, Ltd.

KEYWORDS
adaptive cage generation; skeleton-driven; mesh sequence representation

***Correspondence**
Jieqing Feng, State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China.
E-mail: jqfeng@cad.zju.edu.cn

1. INTRODUCTION

Recent advances in performance capture [1–3] and deformation transfer [4,5] techniques have produced an increasing number of detailed mesh-based animation sequences. They are valuable resources for data-driven animations. However, the reuse, editing, and post-processing of mesh-based sequence data are not straightforward. A high-level control structure should be extracted and defined in advance to facilitate mesh sequence applications. There are two features that a feasible high-level control structure should possess: (i) compactness and intuitiveness and (ii) the expression of both rigid limb motions and time-varying details.

There are two prevalent high-level control structures adopted for representations, skeleton-based representations, and cage-based representations. The former representation offers natural control for rigid limb motions [6] but is not suitable for flexible animated details. The cage-based representation can faithfully reproduce time-varying details [7] but is not suitable for posing an articulated shape. Combining the two control structures, Ju *et al.* [8] proposed a template-based cage driven by a kinematic skeleton for reusable skinning animations.

Inspired by their work [8], we propose a high-level control structure for encoding mesh sequences, referred to as adaptive skeleton-driven cages. The main contributions of the proposed technique are the following:

- A simple sketch-based method for extracting a hierarchical skeleton and constructing a skeleton-driven cage from the rest pose mesh.
- An efficient framework for combining the strengths of skeleton-based and cage-based representation to handle animated sequences with time-varying details.
- The ability to faithfully reproduce mesh sequences benefits from the proposed adaptive cage generation method, which makes it feasible to automatically refine the cage, thus improving the quality further.

Our experiments demonstrate that the proposed method is a compact and intuitive representation of the mesh sequences and preserves time-varying details. This attractive new tool can accomplish various post-processing tasks, such as mesh sequence compression, pose editing and deformation transfer. The paper is organized as follows: Section 2 describes the related work; Section 3 details the adaptive skeleton-driven cage sequence generation algorithm; Section 4 discusses the results and potential applications; and finally, Section 5 provides the conclusions and potential avenues for future research.

2. RELATED WORK

Skeleton-Based Control Structure. The skeleton is the dominant control structure in articulated animations because it is flexible for pose-editing and compatible with conventional animation software. de Aguiar *et al.* [6] introduced the first framework that converts a mesh animation into a compact, editable skeleton-based representation. However, skeleton-based representations are less flexible when expressing time-varying details. To compensate for this lack of flexibility, a hybrid representation for human-like character animations [9] was proposed by decomposing the input sequence into coarse and fine deformation components, where the coarse component is recovered from an underlying skeleton structure and the fine component is encoded with a nonlinear probabilistic model; and then, it was extended to represent and manipulate cloth simulation data [10]. Our approach is more versatile than these methods, as it extends beyond human-like animations and can also preserve fine time-varying shape details.

Cage-Based Control Structure. Because of their additional degrees of freedom, cage-based representations are more suitable than skeleton-based representations for capturing time-varying details. Furthermore, the well-known artifacts of conventional skinning methods can be avoided via cage encoding [8], representing a significant advantage for cage-based representations. In cage-based representations, a vertex is encoded as a linear combination of cage geometries and cage coordinates, such as mean value coordinates (MVC) [11], harmonic coordinates (HC) [12], and Green coordinates (GC) [13]. Xu *et al.* [14] cast the initial cage transfer problem as a linear skinning process. However, the reproduced result is sensitive to the quality of key frame cages. Ben-Chen *et al.* [5] achieved deformation transfer based on variational harmonic maps [15], where the harmonic basis functions defined inside the cage are adopted to encode the mesh sequence. Their method does not generate an explicit geometric control structure, and the harmonic mapping is appropriate for expressing "as-rigid-as-possible" deformation [15]. An alternative is to directly solve the problem in terms of cage vertices. Savoye *et al.* [16] enforced a global regularization term to preserve the differential coordinates of cage vertices given the user's constraints. However, differential coordinates are not rotation invariant, which may lead to notable shrinking artifacts. From the algebraic perspective, Thiery *et al.* [7] accelerated and stabilized the deformation inversion process based on maximum volume sub-matrices and a new spectral regularization. As a result, the reproduced sequence is at high fidelity compared with that obtained in [6,16]. Duveau *et al.* [17] addressed the cage sequence recovery problem for raw captured data via the machine learning method. However, intuitive cage manipulation is still not addressed by the methods described previously. Thus, our method attempts to provide a user-friendly cage sequence that is driven by the corresponding skeletons.

3. ADAPTIVE SKELETON-DRIVEN CAGE SEQUENCE GENERATION

3.1. Overview

The proposed method faithfully converts an animated mesh sequence into a skeleton-driven cage-based representation that acts as an intuitive and flexible high-level control structure for reusing the animated mesh sequence. Given an animated mesh sequence with temporal coherency, the framework of the proposed method is shown in Figure 1, which contains three steps:

- The initial skeleton of the rest pose is derived through an intuitive sketch. Then, the corresponding initial cage, which is bound to the skeleton, is generated adaptively.
- The skeleton sequence is efficiently generated by transferring the initial skeleton to the other meshes.

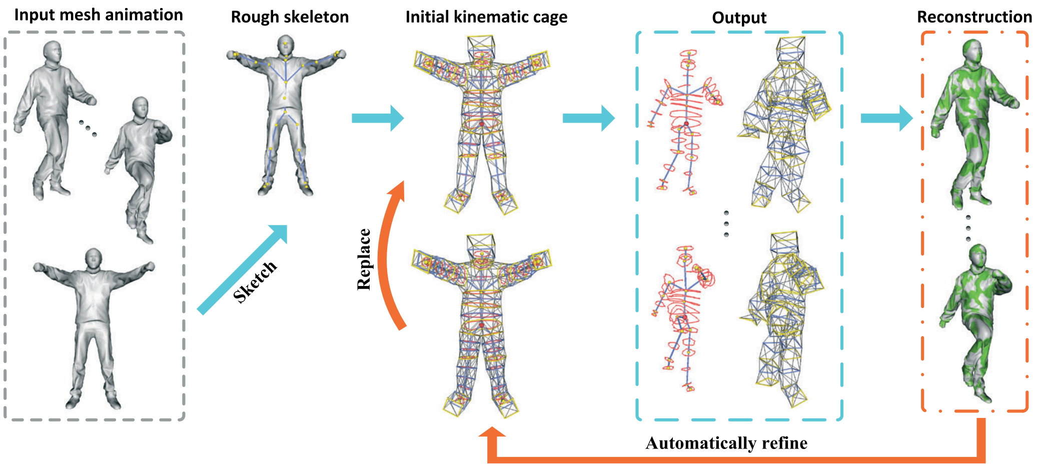


Figure 1. Overview of the proposed algorithm.

The corresponding cage sequence is generated by robustly solving a least-squares fitting problem.

- Optionally, the cage sequence can be automatically refined to reconstruct time-varying details more faithfully.

3.2. Initial Skeleton and Cage Generation

3.2.1. Initial Skeleton Generation.

Sketching is a robust and efficient method of building a kinematic skeleton with minimal effort [18,19]. To facilitate the subsequent processing, we sketch on the rest pose mesh to determine a rough skeleton (Figure 2(a)). The selected mesh vertices are adopted as joints, and the line segments between them are adopted as bones, which are also used to infer the skeleton hierarchy. The sketching process normally takes less than one minute (cf. video).

Then, some shape features are exploited on the rest pose mesh, which can capture the change in shape well. The prominent cross-section (PCS) [20], which is computed at each joint of the rough skeleton, can assume this role. Although the internal skeleton can be adopted as an input for the proposed framework, the rough skeleton helps to efficiently compute the PCS. The proposition of the PCS is that the vertex normals of the cross-section are on a great circle or parallel to a great circle in the sectional Gauss map. The sectional Gauss map is constructed by mapping the normals onto a unit sphere. Given an initial plane defined by the normal and one principle curvature vector at the selected vertex, the process of computing the best-fitting plane on a Gauss map is repeated until the normal of the plane and the pole of the unit sphere coincide within a user specified threshold (10^{-6}). The iterations converge rapidly within a few steps. The algorithm details and its robustness are provided in [20]. The internal skeleton (Figure 2(c)) is obtained by replacing the joint positions in the rough skeleton with the centroids of the corresponding PCSs (Figure 2).

3.2.2. Initial Skeleton-Driven Cage Generation.

Now, we will present a skeleton-driven cage generation scheme based on the PCSs. Our method is inspired by Ju *et al.* [8], where a cage is constructed by connecting the adjacent templates at joints and bones in a predefined manner. The predefined templates are the base elements for a cage. In contrast to using predefined templates as the base elements of the cage, we adaptively compute the base elements from PCSs and avoid the need for a predefined template library. There are two advantages of the adaptive scheme. First, it can be easily extended to other articulated character cases. Second, it makes the cage conform to the embedded shape well, such that the fine time-varying details can be reproduced faithfully.

Each base element in our cage is an offset of a simplified cross-section (OSCS). A 2D Cartesian coordinate system should be defined at the plane of each PCS to connect adjacent OSCSs in a rotation minimizing manner. The coordinate system is defined as follows. First, the root 2D Cartesian coordinate system $(\mathbf{u}_0, \mathbf{v}_0)$ is determined via principle component analysis of the PCS at the root joint $\mathbf{c}_0$. Then, $(\mathbf{u}_0, \mathbf{v}_0)$ is progressively propagated to the other PCSs by traversing the skeleton structure. The propagation is achieved while minimizing rotation distortions, that is, its parent $\mathbf{u}$-axis is projected on the PCS plane along the normal of the PCS, and the $\mathbf{v}$-axis is determined accordingly (Figure 3(a)).

An OSCS for each PCS with a corresponding coordinates system $(\mathbf{u}, \mathbf{v})$ is generated adaptively (Figure 3(b)) as follows. First, the PCS curve is partitioned into four curve segments based on the quadrants of $(\mathbf{u}, \mathbf{v})$. For each curve segment C, its approximate curve is denoted C_{App}. The outward offset of C_{App} is denoted C_{Off}. The offset distance can be used to adjust the tightness of the cage with respect to the embedded model, whose default value is set

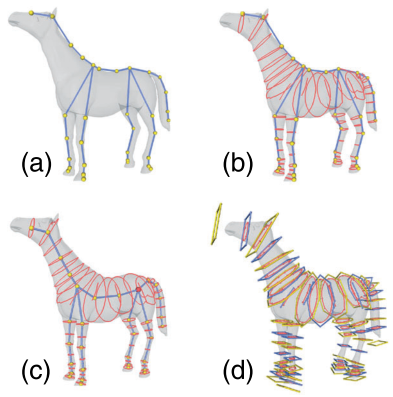


Figure 2. (a) A rough skeleton is gained by sketching. (b) Prominent cross-sections are generated at each joint, (c) the internal skeleton, and (d) joint offset of a simplified cross-sections (OSCSs) (yellow) and interpolated OSCSs (blue).

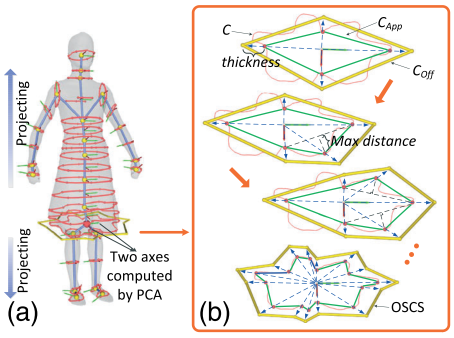


Figure 3. (a) Propagation of the 2D coordinate system at the root joint to other joints and (b) generation of an offset of a simplified cross-section (OSCS) by adaptively approximating the prominent cross-section (PCS). The red curves are Cs, the green polylines are C_{App}s, the yellow polylines are C_{Off}s, and the red points are the inserted points for C_{App}s.

as one third of the average length of the bones. The initial C_{App} is the line segment connecting two ends of C. Then, C_{App} and C_{Off} are refined adaptively until the approximation error is less than the threshold and C lies inside C_{Off}. Here, the approximation error is defined by the maximum distance between the points on C and C_{App}. Finally, four C_{Off}s are merged as a whole OSCS. If the generated OSCS is a quadrilateral, it will be replaced by its axis-aligned bounding box (AABB) to avoid intersections between the model and generated cage.

After generating OSCSs at the joints, some interpolated OSCSs on the bones are introduced (Figure 2(d)) to capture the shape well and to avoid intersection between the generated cage and embedded mesh. Given a bone of the skeleton, the normal of interpolated plane is computed by spherical interpolation of two plane normals of the adjacent joint PCSs. After the cross-section between the plane and mesh is obtained, its OSCS can be computed as described previously.

Once all OSCSs are generated, a modified algorithm for surface reconstruction from cross-sections [21] is proposed to generate the cage (Figure 4). First, the skeleton is decomposed into several branches at the joints of valence of 3 or more; at this stage, the branches conforming to the long axis of the model's AABB are merged (i.e., the torso). Along each branch, the partial cage is constructed by properly connecting the nearest points of the adjacent OSCSs. The nearest point correspondences are established in each segment of the OSCSs partitioned by their AABBs. Finally, all partial cages are stitched to generate a closed two-manifold cage.

In our approach, the cross-section-based cage is bound to the skeleton. When the user edits the joints interactively (cf. video), only the interpolated OSCSs on the bones undergo rigid transformations. The OSCS at a joint is computed by blending the transformed OSCSs that are yielded based on each bone adjacent to the joint. To avoid artifacts, such as skin collapsing, the spherical averaging method [22] is adopted for skin blending as in [8].

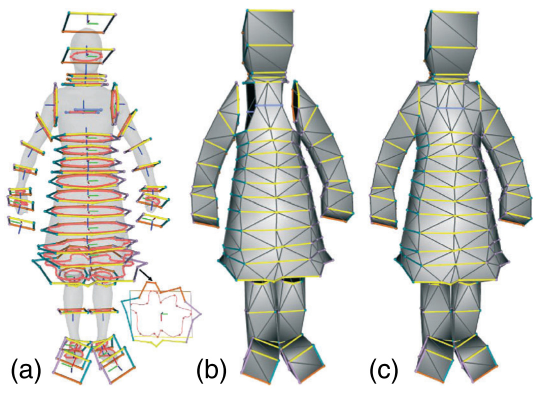


Figure 4. (a) Subdivision of each offset of a simplified cross-section into four segments, (b) generated partial cages, and (c) stitched partial cages.

3.3. Skeleton Propagation and Cage Sequence Recovery

The cross-sections on the rest pose mesh can be mapped onto other meshes because the meshes in the sequence have one-to-one correspondences. In general, the mapped cross-section on the mesh is no longer on a plane. The centroid of the mapped cross-section can still be adopted as the joint position of the skeleton. The skeletons of other meshes have the same hierarchical structure as the initial skeleton but have time-varying joint positions.

Unlike the skeleton, the rest pose cage is propagated to other meshes in a cage-based deformation manner, which is formulated as a least-squares fitting problem. Denote the points of the rest pose mesh and rest pose cage as $\mathbf{P}_r = \{p_i \in \mathbf{R}^3\}_{i=0}^{n-1}$ and $\mathbf{C}_r = \{c_j \in \mathbf{R}^3\}_{j=0}^{m-1}$, respectively, where n and m are the number of vertices in the mesh and cage, respectively. Each point p_i can be represented as a linear combination of cage coordinates (e.g., MVC, HC, or GC), and cage vertex positions as $p_i = \sum_j \omega_j(i) \cdot c_j$, or in matrix form:

$$\mathbf{P}_r = \mathbf{W} \cdot \mathbf{C}_r \tag{1}$$

where $\mathbf{W}$ is the $n \times m$ cage coordinate matrix. For the case of GC, $\mathbf{C}_r$ includes both the vertex positions and face normals. From the perspective of cage-based deformation, a given mesh $\mathbf{P}_t$ in the sequence can be regarded as a deformation of the rest pose mesh, where the cage coordinate matrix $\mathbf{W}$ is fixed. The deformed cage $\mathbf{C}_t$ can be computed by solving the following least-squares fitting problem:

$$\underset{\mathbf{C}_t}{argmin} \parallel \mathbf{W} \cdot \mathbf{C}_t - \mathbf{P}_t \parallel^2 \tag{2}$$

This is an overdetermined fitting problem, which has the following explicit solution in the least-squares sense:

$$\mathbf{C}_t = \left(\mathbf{W}^T\mathbf{W}\right)^{-1} \mathbf{W}^T\mathbf{P}_t \tag{3}$$

From Equation (3), we can determine that the cage $\mathbf{C}_t$ is primarily dominated by the cage coordinates $\mathbf{W}$ for the given mesh $\mathbf{P}_t$, where $\mathbf{W}$ is primarily determined by the relationship between the initial cage and rest pose mesh. If the initial cage is not well defined (i.e., it is sparse, has an unevenly distributed geometry, or is not close to the embedded shape), the least-squares solution may introduce serious spikes on the resulting cage, which makes the cage infeasible (Figure 5). Our skeleton-driven cage generation considers both the global structure and local geometric details of the mesh sequence, which makes the initial cage approximate the embedded shape well. Thus, the solution is always feasible. Furthermore, the coefficient matrix $(\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$ can be precomputed and prefactorized to reduce the computational cost. Additionally, the well-defined cage $\mathbf{C}_r$, which is generated by our approach, can be used as the input for CageR [7].

In theory, the proposed approach is independent of the cage coordinates as in [7]. Similar to the results in [7],

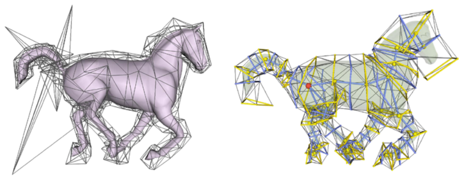


Figure 5. Cages computed via least-squares fitting with mean value coordinates embedding. Left: Serious spikes obtained with the manual cage [7]. Right: The spike-free, well-defined result obtained with our skeleton-driven cage.

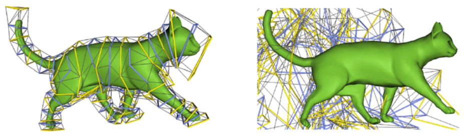


Figure 6. Left: Cage computed using mean value coordinates. Right: Invalid cage computed using Green coordinates.

the cage generated using GC is not stable because all the unknowns in Equation (3) are processed independently but the face normals are closely related to the cage vertices. Thus, MVC are adopted in our examples for simplicity and efficiency. Figure 6 presents the cages computed using MVC and GC, respectively.

3.4. Adaptive Cage Refinement

At this point, a skeleton-driven cage representation of the mesh sequence has been obtained. This representation can be regarded as a high-level control structure that constitutes a combination of the skeleton-based and cage-based approaches. Because of the appropriate way our cage is constructed, the cage can be refined adaptively to improve the fidelity of the geometric detail reconstruction.

Utilizing the cage construction, we can easily refine the cage in two ways. The first is radial refinement, that is, increasing the approximate accuracy of OSCS with respect to the cross-section. The other is axial refinement, that is, increasing the number of interpolated OSCSs on the bone. Adaptive refinement is controlled under the mesh reconstruction errors via a heuristic approach. First, the rest pose mesh is simply segmented according to the planes of PCSs. For each mesh, the L_2 (average) and L_∞(maximum) reconstruction errors are computed. For each point on the mesh, the reconstruction error is defined as the percentage of the point-to-point Euclidean distance to the bounding box diagonal of the mesh as in [7]. If the percentage of the number of larger error vertices exceeds a given threshold t in a segment, the surrounding initial cage will be refined axially and radially, where larger error vertices denotes reconstruction errors greater than the L_2 error, or the segment contains the L_∞ error. Finally, the cage propagation in Section 3.3 will be performed again to obtain a refined cage sequence. The number of segments to be refined is in inverse proportion to the threshold t. Experimental results show that the value of the threshold t is set to 10%, and the refinement process performs once, which can significantly improve the quality and keep the cage simplicity.

4. RESULTS AND APPLICATIONS

The proposed algorithm has been implemented on a PC with an Intel Core2 Quad 2.5GHz CPU and a GeForce

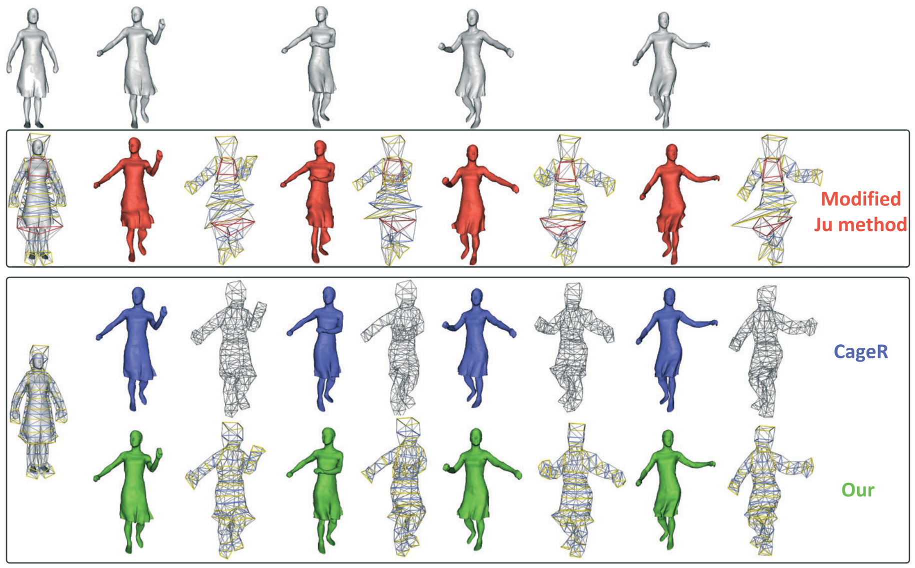
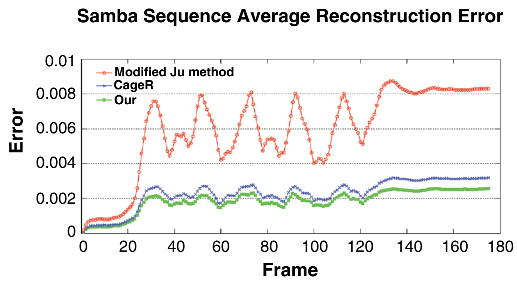


Figure 7. Comparison between the modified Ju method, CageR [7], and our method for the Samba sequence with a time-varying nonrigid swinging skirt. The initial cage is shown in the leftmost column. For each example pose, the reconstructed pose and corresponding cage are shown on the left and right, respectively.



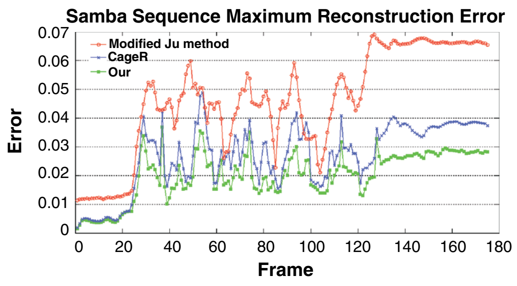


Figure 8. Comparison of the L_2 and L_∞ reconstruction errors of the Samba sequence in Figure 7.

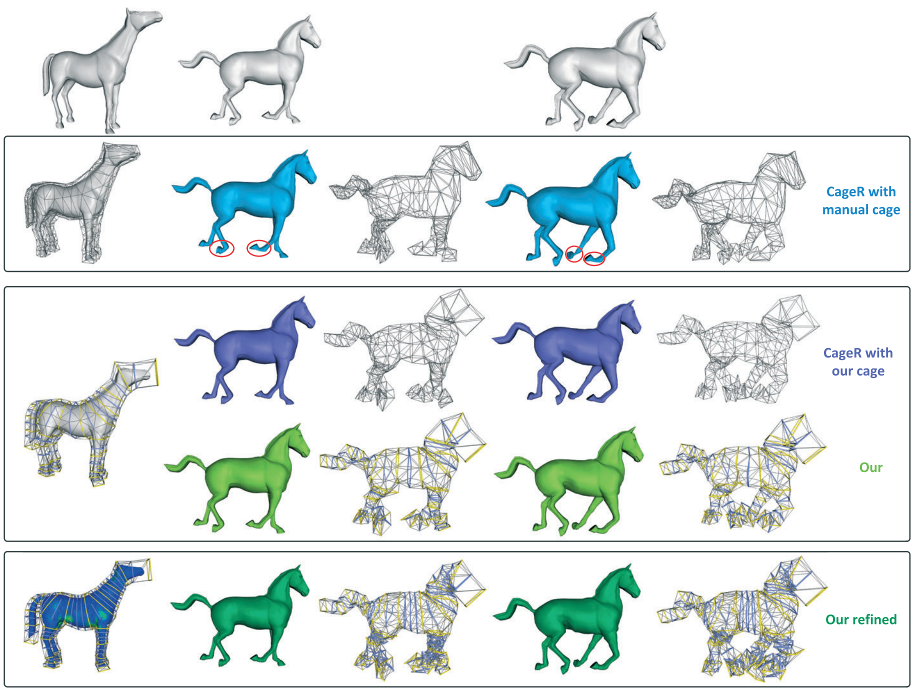
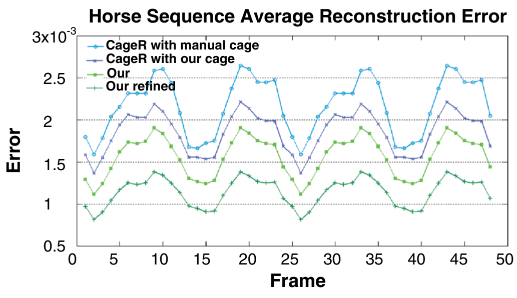


Figure 9. Comparison between CageR [7] using either the manual cage or our cage, and our method either without or with the refinement process. The error distribution of our approach using an unrefined cage is visualized in the leftmost column of the last row.



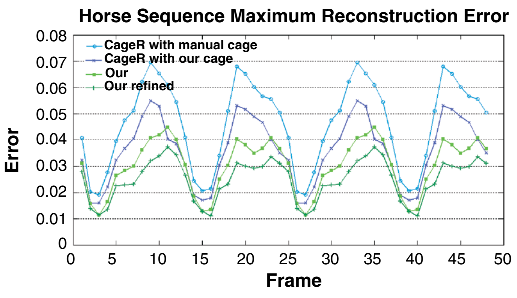


Figure 10. Comparison of the L_2 and L_∞ reconstruction errors of the horse sequence in Figure 9.

GTX 260 GPU. The algorithm runs in a single thread without GPU acceleration.

CageR [7] and the method of Ju *et al.* [8] are implemented for comparison purposes. To make it applicable to the context of mesh animations, the method of Ju *et al.* [8] is modified as follows: the input skeleton is generated using our method, the initial cage is reconstructed via the original method [8], and the cage sequence is generated using the least-squares fitting approach. The animated mesh sequences captured from multiview silhouette reconstruction [1] and synthesized by deformation transfer [4] are adopted as test examples. In particular, the Samba mesh sequence contains highly deformed time-varying geometric details on the swinging skirt, as shown in Figure 7.

4.1. Quality and Performance

Figure 7 presents the input Samba sequence and the sequences reproduced by the modified Ju method, CageR, and our method. For a fair comparison, our proposed rest pose cage is used as the input for CageR. Among the three methods, our method and CageR can reconstruct the Samba mesh sequence more faithfully than the modified Ju method. This result is unsurprising because Ju's method is designed for reusing skinning animations [8]. Although it has been modified for mesh sequence representation, the fixed templates are not appropriate for representing highly deformed geometric details. In addition to a visual quality comparison, the quantitative comparison of the L_2 and L_∞ reconstruction errors is also given. As shown in Figure 8, our approach can reconstruct the mesh sequence more faithfully than the other approaches, even if the mesh sequence contains flexible regions.

Using the manual cage generously provided to us by Thiery *et al.* [7], we can compare the results produced by CageR using the manual cage and our cage. As shown in the first two rows of Figure 9, the visual reconstruction qualities appear similar, except for the distortions at the horse ankles obtained by CageR with the manual cage. Furthermore, the reconstruction errors in Figure 10 demonstrate that our elaborately generated cage can gain higher fidelity than the manual cage. In addition, the last two rows of Figure 9 demonstrate that our approach, either without or with cage refinement, can achieve results indistinguishable from the input data. If the application requires greater accuracy, the refined cage sequence, which is an optimal option, offers the lowest reconstruction errors, as shown in Figure 10.

Table I lists the detailed runtime comparison of CageR and our approach. Because of the efficient skeleton sequence extraction and the coefficient matrix prefactorization in Equation (3), the proposed approach is more efficient than CageR, even if the preprocessing step of CageR is accelerated by the GPU. The runtime of our method in the preprocessing step contains the cage coordinate matrix factorization and the initial skeleton-driven cage generation. The runtime of our method in the reconstruction step includes the skeleton propagation and cage propagation. The figures demonstrate that the proposed skeleton extraction occurs in real time.

4.2. Applications

Animation Lossy Compression. Similar to CageR [7], the reconstruction step of our method depends only on P_r, C_r, and the generated cage sequence. Thus, our method can be applied to animation lossy compression. For instance, the 234 MB Samba sequence encoded in ASCII format can be compressed to 14 MB with almost no visual changes by reproduction.

Deformation Transfer. The one-to-one correspondences between the source cage and target cage will be automatically built once the same skeleton structure is sketched on the source model and target model. Therefore, the classical deformation transfer method [4] can be directly applied to the cages. Figure 11 shows the horse poses are transferred to a camel.

Pose and Animation Editing. We can intuitively produce new postures of the input mesh by modifying the joint positions and orientations. Our method also has the potential to edit the original animation locally by altering the joint parameters of selected joints (cf. video).

Table I. Detailed runtime comparison between CageR and our method.

Input sequence	Mesh (#V/#Frame)	Cage (#V/#T)	Preprocess (ms) MVC	Preprocess (ms) CageR	Preprocess (ms) Ours	Reconstruction (ms) CageR	Reconstruction (ms) Ours	Total (ms) CageR	Total (ms) Ours
Samba	9971/175	247/490	3082	4298	1244+89	81	1+80	21,555	18,590
Horse	8431/48	300/596	3140	6885	1586+98	121	2+74	15,833	8472
Man	10002/250	198/392	2436	2757	828+95	70	2+51	22,693	16,609

MVC, mean value coordinates.

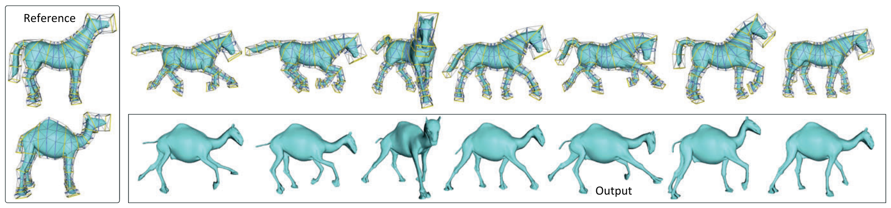


Figure 11. Horse poses transferred to a camel using our skeleton-driven cages.

5. CONCLUSION

We present a compact and intuitive mesh animation representation method that effectively combines the skeleton-based and cage-based approaches, which is suitable for representing mesh animations with fine time-varying details. To this end, an initial skeleton structure is obtained through an efficient sketching interface. Meanwhile, the initial cage is adaptively constructed along the skeleton from PCSs. The skeleton sequence can be extracted efficiently and robustly because of the one-to-one mapping of cross-sections. The cage sequence is generated by solving a least-squares problem. Furthermore, the cage refinement represents a method for achieving more accurate reconstructions. As a new high-level control structure, the proposed approach is applicable to a variety of mainstream applications, such as animation compression, pose and animation editing, and deformation transfer.

Our approach suffers from the same limitation as many existing methods, that is, the mesh sequence is temporally coherent. Second, the proposed cage generation algorithm will be robust if the model is in a rest pose that helps to avoid the self-intersection of the cage. In future work, we will extend our approach to handle mesh sequences without temporal coherence. Furthermore, we will implement more cage coordinates (e.g., HC and PMVC) in our framework and explore the sufficient conditions that make the result of Equation (3) a valid cage. Additionally, we will develop more possible applications based on the proposed control structure and combine it with other works, such as *Cages [23] and data-driven shape morphing [24].

ACKNOWLEDGEMENTS

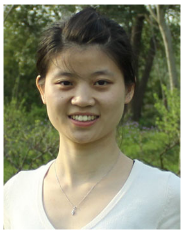
We would like to thank the anonymous reviewers for their valuable comments and Jean-Marc Thiery for sharing the horse cage sequence. This work is supported by the National Natural Science Foundation of China under Grant nos. 61170138, 60933007 and the Program for New Century Excellent Talents in University under Grant no. NCET-10-0728.

REFERENCES

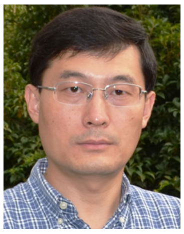
1. Daniel V, Baran I, Matusik W, Popović J. Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics* 2008; **27**(3): 97:1–97:9.
2. de Aguiar E, Stoll C, Theobalt C, Ahmed N, Seidel H-P, Thrun S. Performance capture from sparse multi-view video. *ACM Transactions on Graphics* 2008; **27**(3): 98:1–98:10.
3. Popa T, South-Dickinson I, Bradley D, Sheffer A, Heidrich W. Globally consistent space-time reconstruction. *Computer Graphics Forum* 2010; **29**(5): 1633–1642.
4. Sumner RW, Popović J. Deformation transfer for triangle meshes. *ACM Transactions on Graphics* 2004; **23**(3): 399–405.
5. Ben-Chen M, Weber O, Gotsman C. Spatial deformation transfer. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM: New York, NY, USA, 2009; 67–74.
6. De Aguiar E, Theobalt C, Thrun S, Seidel H-P. Automatic conversion of mesh animations into skeleton-based animations. *Computer Graphics Forum* 2008; **27**(2): 389–397.
7. Thiery J-M, Tierny J, Boubekeur T. CageR: cage-based reverse engineering of animated 3D shapes. *Computer Graphics Forum* 2012; **31**(8): 2303–2316.
8. Ju T, Zhou Q-Y, van de Panne M, Cohen-Or D, Neumann U. Reusable skinning templates using cage-based deformations. *ACM Transactions on Graphics* 2008; **27**(5): 122:1–122:10.
9. de Aguiar E, Ukita N. Representing and manipulating mesh-based character animations. In *Graphics, Patterns and Images (SIBGRAPI)*. IEEE Computer Society Press: Ouro Preto, Minas Gerais, Brazil, 2012; 198–204.
10. de Aguiar E, Ukita N. Representing mesh-based character animations. *Computers & Graphics* 2014; **38**(0): 10–17.
11. Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics* 2005; **24**(3): 561–566.
12. Joshi P, Meyer M, DeRose T, Green B, Sanocki T. Harmonic coordinates for character articulation. *ACM Transactions on Graphics* 2007; **26**(3): 71:1–71:10.
13. Lipman Y, Levin D, Cohen-Or D. Green coordinates. *ACM Transactions on Graphics* 2008; **27**(3): 78:1–78:10.
14. Xu W, Zhou K, Yu Y, Tan Q, Peng Q, Guo B. Gradient domain editing of deforming mesh sequences. *ACM Transactions on Graphics* 2007; **26**(3): 84:1–84:10.
15. Ben-Chen M, Weber O, Gotsman C. Variational harmonic maps for space deformation. *ACM Transactions on Graphics* 2009; **28**(3): 34:1–34:11.
16. Savoye Y, Franco J-S. Cage-based tracking for performance animation. In *Asian Conference on Computer Vision*. Springer-Verlag: Berlin, Heidelberg, 2011; 599–612.
17. Duveau E, Courtemanche S, Reveret L, Boyer E. Cage-based motion recovery using manifold learning. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*. IEEE Computer Society Press: Zurich, Switzerland, 2012; 206–213.
18. Yao C-Y, Chu H-K, Ju T, Lee T-Y. Compatible quadrangulation by sketching. *Computer Animation and Virtual Worlds* 2009; **20**(2-3): 101–109.

19. Chen C-H, Tsai M-H, Lin I-C, Lu P-H. Skeleton-driven surface deformation through lattices for real-time character animation. *The Visual Computer* 2013; **29**(4): 241–251.
20. Sellamani S, Muthuganapathy R, Kalyanaraman Y, Murugappan S, Goyal M, Ramani K, Hoffman CM. PCS: prominent cross-sections for mesh models. *Computer-Aided Design and Applications* 2010; **7**(4): 601–620.
21. Chen SE, Parent RE. Shape averaging and its applications to industrial design. *Computer Graphics and Applications* 1989; **9**(1): 47–54.
22. Buss SR, Fillmore JP. Spherical averages and applications to spherical splines and interpolation. *ACM Transactions on Graphics* 2001; **20**(2): 95–126.
23. García FG, Paradinas T, Coll N, Patow G. *Cages: a multilevel, multi-cage-based system for mesh deformation. *ACM Transactions on Graphics* 2013; **32**(3): Article 24.
24. Gao L, Lai Y-K, Huang Q-X, Hu S-M. A data-driven approach to realistic shape morphing. *Computer Graphics Forum* 2013; **32**(2pt4): 449–457.

AUTHORS' BIOGRAPHIES



Xue Chen is a PhD student of the State Key Lab of CAD&CG, Zhejiang University, China. She received her BS degree in computer science from Hunan University, China, in 2009. She is particularly interested in geometric modeling and computer animation.



Jieqing Feng is a professor in the State Key Lab of CAD&CG, Zhejiang University, China. He received his BSc in applied mathematics from the National University of Defense Technology in 1992 and his PhD in computer graphics from Zhejiang University in 1997. His research interests include geometric modeling, real-time rendering, and computer animation.