DOI: 10.1111/cgf.12718

COMPUTER GRAPHICS *forum*
Volume 00 (2015), number 0 pp. 1–12

Mesh Sequence Morphing

Xue Chen[1], Jieqing Feng[1,*] and Dominique Bechmann[2]

[1]State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China
xuechen@zju.edu.cn, jqfeng@cad.zju.edu.cn
[2]ICube, Universite de Strasbourg, Strasbourg, France
bechmann@unistra.fr

Abstract

Morphing is an important technique for the generation of special effects in computer animation. However, an analogous technique has not yet been applied to the increasingly prevalent animation representation, i.e. 3D mesh sequences. In this paper, a technique for morphing between two mesh sequences is proposed to simultaneously blend motions and interpolate shapes. Based on all possible combinations of the motions and geometries, a universal framework is proposed to recreate various plausible mesh sequences. To enable a universal framework, we design a skeleton-driven cage-based deformation transfer scheme which can account for motion blending and geometry interpolation. To establish one-to-one correspondence for interpolating between two mesh sequences, a hybrid cross-parameterization scheme that fully utilizes the skeleton-driven cage control structure and adapts user-specified joint-like markers, is introduced. The experimental results demonstrate that the framework, not only accomplishes mesh sequence morphing, but also is suitable for a wide range of applications such as deformation transfer, motion blending or transition and dynamic shape interpolation.

Keywords: mesh sequence morphing, mesh sequence representation, cross-parameterization, deformation transfer

ACM CCS: I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—Hierarchy and geometric transformations; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Three-dimensional mesh sequences, which represent animations in the form of deforming meshes, have received extensive attention in the development of post-processing techniques in 3D animation. Deformation transfer (DT) and animation warping are two typical techniques used to recreate meaningful mesh animations from a given mesh sequence. However, these techniques focus on exploiting only one aspect of a given mesh sequence, i.e. geometry or motion. Is it possible to simultaneously process the two aspects to recreate meaningful 3D mesh sequences? This question is closely related to morphing techniques, which have been an active research topic in the fields of images [LLN*14b], videos [LLN*14a] and 3D shapes [GLHH13]. In this paper, we propose a new morphing technique for mesh sequences, namely, mesh sequence morphing.

Intuitively, mesh sequence morphing involves simultaneously blending the geometries and motions of two input sequences; the result is a new mesh sequence with features of both sequences in geometry and motion, as shown in Figure 1. Our goal is to address motion blending and shape interpolation in a unified manner. The challenges can be categorized into three aspects: (1) exploring natural integration between motion blending and shape interpolation; (2) developing a universal framework for addressing all possible blending combinations of the geometries and motions, i.e. morphing strategies (see Table 2 in Section 4.2) and (3) developing efficient algorithms to handle the batch processing of dynamic shape morphing. To address the above challenges, a skeleton-driven cage-based representation [CF14] is adopted as a high-level control structure to encode mesh sequences. The skeleton, as a low-dimensional abstraction of motion, is highly compatible with motion blending methods. The cage, as an agent for the embedded geometry, is responsible for reproducing the geometry and accelerating the subsequent geometry processing. To uniformly handle various morphing strategies, the proposed framework consists of four processing steps: mesh sequence representation, cross-parameterization, motion blending and dynamic shape interpolation. This framework fully utilizes the

*Corresponding author: J. Feng (jqfeng@cad.zju.edu.cn)

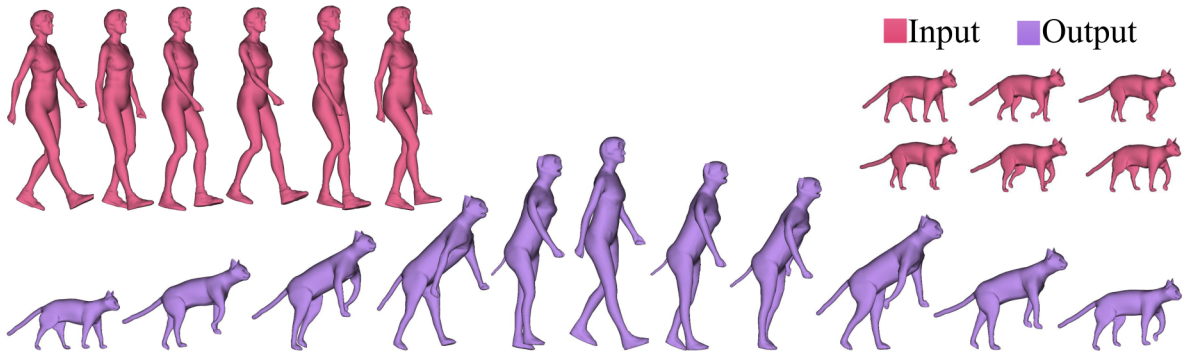

Figure 1: *Terminology of mesh sequence morphing. (Top) Two input mesh sequences with different geometries and motions. (Bottom) A new mesh sequence demonstrating 'in-between' motion with gradually varying geometries.*

skeleton-driven cage-based representation to provide a universal tool for generating a series of meaningful mesh sequences directly from inputs.

There are two main contributions in this paper. First, we propose the concept of mesh sequence morphing. Second, a universal framework is designed to generate various blending effects. To enable an efficient processing and a feasible framework, we introduce a hybrid cross-parameterization approach composed of domain-based parameterization and template-based fitting approaches tailored for our context. In addition, we propose a skeleton-driven cage-based DT approach that provides an interface for blending motions via the skeletons, bridges the motions and geometries via the skeleton-driven cages and reconstructs the geometries to be interpolated via the cage-based deformation.

2. Related Work

Although mesh sequences have been widely investigated and applied in data-driven animation, there are no published methods for blending both the geometries and motions of two different mesh sequences (i.e. mesh sequence morphing). Kircher and Garland [KG08] developed a powerful motion processing tool based on their proposed differential representation of surfaces. This tool provides various techniques, including key-frame insertion, temporal signal processing and motion blending. This method assumes that the input mesh sequences are derived from the same model; thus, it is not applicable to mesh sequences generated from different models. In other words, the motion blending operation in [KG08] can be considered a special case of motion transition in our context. Recently, Yang *et al.* [YXF14] presented a space–time morphing approach that gradually transfers the geometric details of a static model to a mesh sequence for detail enhancement. It is well known that traditional 3D shape morphing [Ale02] has two major issues: finding one-to-one correspondence and producing smooth interpolated shapes. Compared with 3D shape morphing, the proposed mesh sequence morphing technique should address these issues for both dynamic geometries and motions. In the following, only work relevant to the proposed framework is reviewed.

Mesh sequence representations are critical for efficiently and flexibly reusing mesh animations by employing a high-level control structure, e.g. skeleton or cage. Skeleton-based representations [DATTS08, dAU14, LD14] offer more natural control for rigid limb motions, and cage-based representations [XZY*07, TTB12, LLX*15] are more suitable for preserving time-varying details. In contrast, the combination of the skeleton and cage in a control structure [CF14] is an optimal choice for our morphing purpose, because the skeleton is necessary for blending motions and the extraction of the cages is more efficient than the computation of the skeletal parameters [DATTS08, dAU14, LD14].

Cross-parameterization is an essential pre-requisite for establishing one-to-one correspondence between two shapes in 3D mesh morphing. Alexa provided a useful survey of previous work [Ale02]. Recent approaches can be classified into two categories, i.e. domain-based parameterization approaches [KS04, SAPH04, WPZ*11] and template-based fitting approaches [ACP03, SP04, ZLJW06, YLSL11]. In the pre-processing step of both approaches, pairwise one-to-one corresponding markers should be manually specified on the source and target meshes, respectively. The former approach usually adopts an indirect parameterization domain (e.g. a square, sphere or other base domain) to build a bijective mapping between the source and target meshes. Regardless of the differences in the two meshes, the geometric details of the reparameterized mesh can be well reproduced even with joint-like markers [YCJL09]. The main difficulty in domain-based parameterization approaches is balancing efficiency and robustness to construct well-shaped patch layouts of the input meshes because many complicated geometric and topological operations are involved. As an alternative, template-based fitting approaches directly compute cross-parameterization by deforming a template mesh to closely approximate the target mesh in the least-squares sense. However, the direct approaches may not consider shape preservation property, which is prone to introducing relatively large approximation errors especially when semantic marker pairs are missed on regions like the ears. The proposed hybrid cross-parameterization combines the above two approaches to provide a trade-off between efficiency and quality under the condition of limited initial marker pairs.

Motion blending is widely used in interactive applications such as 3D computer animation, computer games and virtual reality systems. Using motions of one model obtained via motion capture or key-framed animation, the motion blending technique can generate a smooth 'in-between' motion or a seamless transition motion

between two inputs as realistically as possible. Many successful works have been proposed on motion blending, including a rule-based system for procedurally generated motions [Per95], multi-resolution motion filtering [BW95], motion parameter curves [WP95], motion graphs [KGP02], the timewarping curve method [GSKJ03] and registration curves [KG03]. Because motion blending is not our focus and it is well studied, we use the classical timewarping curve [GSKJ03] to produce blended motions in our universal framework.

Deformation transfer is an important technique for reusing the deformations in a mesh sequence. The technique retargets the deformation of the source shape to the target shape; the technique includes an analysis step for describing the source deformation and a synthesis step for inferring the deformed target shape. Similar to the pre-processing step of the cross-parameterization, pairwise one-to-one corresponding markers should be manually specified in DT as well. Here, two categories of recent approaches are reviewed: surface-based approaches and space-based approaches, which follow the trend of shape deformation methods [CO09]. Surface-based DT methods [SP04, ZRKS05, ZXTD10] adopt various desirable invariant properties to define deformations; these properties are generally called deformation gradients, e.g. a set of affine transformations between pairs of corresponding triangles and their normalized normals. The synthesis of the target mesh can be formulated as a Poisson problem in which the deformation gradient of the target mesh is similar to that of the source mesh. Although these approaches can transfer various mesh deformations, their flexibility for other shape representations is somewhat limited. As an alternative, space-based DT approaches [BCWG09a, CHSB10, YY12] cast the DT problem as a shape deformation problem by encoding the gradient constraints of the source deformation in terms of the cage geometry (i.e. vertex positions or face normals). Thus, space-based approaches can be applied to different shape representations in addition to triangle meshes. The space-based DT methods generally introduce non-linear embedding such as harmonic mappings [BCWG09b, BCWG09a] and Green coordinates [LLCO08, CHSB10, YY12]. However, non-linear embedding is not compatible with the skeleton-driven cage-based framework [CF14]. Therefore, the proposed DT employs the skeleton to capture the source deformation globally and uses the mean value coordinates (MVCs) [JSW05] to duplicate the geometric details, rather than directly applying surface-based DT [SP04] to the cage sequence as in [CF14].

3. Mesh Sequence Morphing

3.1. Method overview

In this paper, we design a new morphing framework for two mesh sequences that can simultaneously blend motions and interpolate geometries. Figure 2 illustrates the pipeline of the proposed framework. The notations in the forms of $G_{s/t/c}^{(r/d)}$ and $M_{s/t}$, where G and M, respectively, indicate geometry and motion, are introduced for conciseness. The superscripts 'r' and 'd' indicate the *rest* pose and *deformed* pose of the mesh, respectively. The subscripts 's', 't' and 'c' indicate the *source* mesh, *target* mesh and *compatible* mesh, respectively. Given the source mesh sequence, the target mesh sequence, and the corresponding rest pose meshes G_s^r and G_t^r with

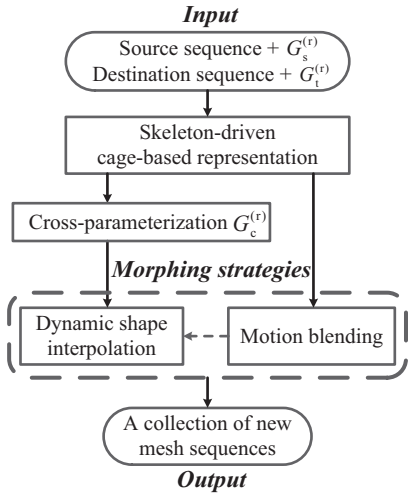

Figure 2: *Flowchart of the proposed unified framework.*

different mesh connectivity, the proposed morphing framework can be summarized in four steps:

(1) **Mesh sequence representation**. To enable a high-level control structure in the subsequent steps, the source and target mesh sequences are encoded as a skeleton-driven cage structure [CF14] with a few user interactions.
(2) **Cross-parameterization**. To establish one-to-one correspondence for dynamic shape interpolation, a hybrid cross-parameterization is introduced for generating the compatible mesh $G_c^{(r)}$ by combining local domain-based parameterization and local template-based fitting, which is well-suited for use with the adopted control structure.
(3) **Motion blending**. To produce natural 'in-between' motion based on the extracted skeleton sequences in the first step, the classic timewarping curve method [GSKJ03] is adopted to find the optimal frame pair set.
(4) **Dynamic shape interpolation**. To generate interpolated geometries for an 'in-between' motion, a reconstruction–interpolation scheme based on the proposed skeleton-driven cage-based DT is adopted.

Although both geometry and motion are encoded in a uniform control structure, they can be processed independently, which can exploit a wide range of morphing possibilities. To extend the framework to apply to various morphing strategies, we propose a hybrid cross-parameterization and a skeleton-driven cage-based DT, which take three aspects into account: compatibility with the adopted control structure, computational cost for processing mesh sequences and user's interactive effort.

3.2. Mesh sequence representation—skeleton-driven cage

During the construction of skeleton-driven cages for two given rest pose meshes, the kinematic skeletons are semi-automatically generated using consistent user-specified mesh vertices (markers) near main joints of both meshes [CF14]. Furthermore, the markers will guide the subsequent cross-parameterization and DT. In general,

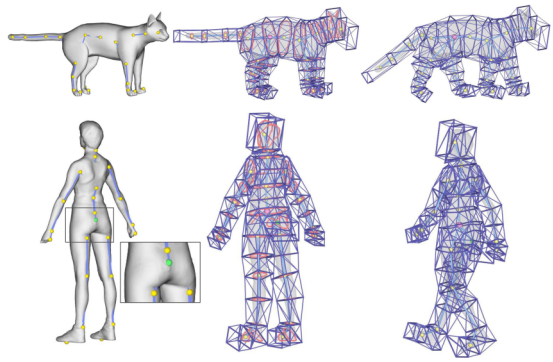

Figure 3: *Pipeline of generating skeleton-driven cage-based control structures. (Left) The compatible sketched skeletons, where the virtual joint of the Woman model is marked in green. (Middle) The constructed skeleton-driven cages for the rest poses. (Right) The reconstructed skeleton-driven cages for one frame in the sequence.*

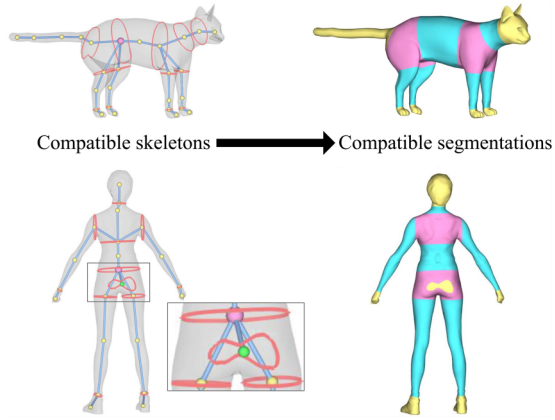

Figure 4: *Compatible segmentations of $G_c^{(r)}$ (Cat) and $G_t^{(r)}$ (Woman) according to the cross-sections at the corresponding joints of the compatible skeletons, where S1 sections are coloured in yellow and magenta, and S2 sections are coloured in blue.*

the skeletons of G_s^r and G_t^r should be roughly compatible. If G_s^r and G_t^r have different skeletal structures, e.g. different branches incident to a joint, virtual joints will be inserted interactively so that the corresponding joints will have the same valence (see Figure 3). Automatic skeleton extraction algorithms, such as [ATC*08], are generally designed to extract a curve skeleton that is not flexible to infer compatible kinematic skeletons for different geometries.

3.3. A hybrid cross-parameterization

Intuitively, domain-based parameterization methods [KS04, SAPH04, WPZ*11] or template-based fitting methods [ACP03, SP04, ZLJW06, YLSL11] can be used to generate the compatible mesh G_c^r. In our framework, initial joint-like marker pairs (first step) are reused to minimize user's effort. Unfortunately, joint-like marker pairs are not enough for template-based fitting approaches at regions such as ears, eyes and junctions on which semantic marker pairs are required to be selected by the user. Considering the obtained skeleton-driven cage, we can construct a quad-domain parameterization from the control structure to accomplish one-to-one correspondence as in [YCJL09]. As mentioned before, a trade-off between efficiency and robustness should be balanced due to many geometric and topological operations are involved in domain-based parameterization methods. Therefore, we propose a hybrid means by using a local quad-domain based parameterization or a local template-based fitting according to the type of the segmented mesh sections, in order to apply to the condition of limited markers and improve algorithm efficiency.

3.3.1. *Consistent mesh segmentation*

Initially, the compatible mesh G_c^r is a duplication of G_s^r. Before hybrid cross-parameterization, G_c^r and G_t^r are compatibly segmented according to the cross-sections attaching to the joints of the compatible skeletons (see Figure 4). Among the segmented sections, the one corresponding to a multi-branch (magenta) or an end joint (yellow) is denoted as S1 section, which will be processed by the quad-domain–based parameterization method. The cylinder-like sections (blue) are denoted as S2 section, which will be processed by the template-based fitting method.

3.3.2. *Local quad-domain–based parameterization*

Similar to [YCJL09], the parameterization consists of two steps: constructing compatible quad-patch layouts and building a smooth mapping between the source and target meshes, as shown in Figure 5. Here, the resources in the control structure will guide the construction of the quad patches. For each S1 section, a local frame adhering to a cross-section, which is propagated from the root joint down to its child joints in a rotation-minimization manner [CF14], is adopted to partition a cross-section into four curve segments (see Figure 5a). Actually, these four partition points are also basic elements to construct an adaptive skeleton-driven cage as in [CF14]. Then, the quad-based domains of S1 sections can be constructed by connecting partition points of the neighbouring cross-sections according to the connectivity of the cage (see Figure 5b). In this manner, the corresponding quad-patch layouts on G_c^r and G_t^r are constructed by mapping each edge of the domain meshes onto the corresponding surfaces (see Figure 5c).

To build a smooth mapping between G_c^r and G_t^r, the local parameterization method [KS04] is employed to reconstruct the geometry of G_c^r. First, each patch is mapped to the corresponding quad domain using mean value parameterization [Flo03]. This defines a mapping f_c from G_c^r to its base domain and a mapping f_t from G_t^r to its base domain. Then, each quad domain of G_c^r is mapped to the matching quad domain of G_t^r by mapping the corresponding corner vertices and using barycentric coordinates for the interior. This defines a mapping function f_{ct}. Finally, the geometry of G_c^r is reconstructed through a mapping f from G_c^r to G_t^r, which is formulated as $f = f_t^{-1} \cdot f_{ct} \cdot f_c$.

3.3.3. *Local template-based fitting parameterization*

Traditionally, template-fitting approaches are applied to an entire mesh, which causes the algorithm complexity to be proportional to the resolution of the entire mesh. In contrast, the proposed local

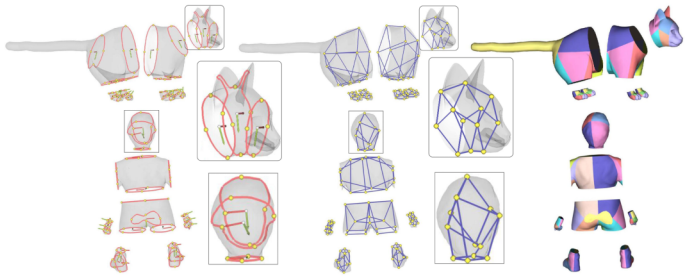

Figure 5: *Pipeline of quad-domain–based cross-parameterization for S1 sections between $G_c^{(r)}$ and $G_t^{(r)}$. The quad mesh domains (b) are constructed by connecting the corresponding partition points (yellow dots) on the neighbouring cross-sections based on the local frames at the joints (a). The patch layouts in (c) of S1 sections are generated by compatibly cutting the meshes according to their parametric domains in (b).*

scheme will effectively accelerate the fitting process by taking each S2 section as an input (see Figure 6a).

For each S2 section, we follow the basic fitting process of [ZLJW06] progressively fitting an initial S2 section of G_c^r to the corresponding S2 section of G_t^r by iteratively solving for a least-squares mesh (LS-mesh) [SCO04]. The solving for LS-mesh is formulated as the following energy function:

$$\min_{V'}(||LV'||^2 + \sum_{k\in C} ||\mathbf{v}_k{}' - \mathbf{v}_k||^2), \quad (1)$$

where L is the Laplacian matrix of a S2 section of G_c^r and a set of vertex $\mathbf{v}_k'$ of G_c^r are constrained to approximate the set of vertex $\mathbf{v}_k$ on the corresponding S2 section of G_t^r. The second term takes the position constraints as soft constraints, where C is the index set of constraints. Thus, the resulting solution trends to keep the smoothness and fairness for all vertices V' (unknowns in Equation (1)) of G_c^r. A good initial fitted G_c^r is helpful to generate a better approximation result for subsequent iteration. In the traditional method, the good initial fitted G_c^r is obtained by solving Equation (1) with a large number of user-specified maker pairs (74 markers in each mesh [ACP03]). To avoid introducing extra user interactions in this step, we utilize the cage-based deformation (MVC) to obtain initial aligned S2 sections. A local cage is trimmed from the faces of the initial cage belonging to the S2 section and extra faces are inserted to close two opening ends (see Figure 6 b). Then, non-rigid iterative closest points strategy [LSP08, ZLJW06] is adopted to establish one-to-one correspondence. During the iterations of solving the progressive LS-mesh for an S2 section, all inner vertices of the S2 section of G_t^r act as candidates to participate in the selection of position constraints by searching the closest vertices on the corresponding S2 section of G_c^r. To keep C^0 continuity of the resulting mesh G_c^r, the boundary vertices of the S2 sections are constrained to the corresponding boundary vertices of the adjacent S1 sections, which have been determined by local domain-based parameterization in Section 3.3.2. Here, the vertex normal and vertex distance are adopted as measures to remove unqualified position constraints, as in [ZLJW06]. In our experiments, the inner product of a normal pair should be greater than 0, and the distance between a vertex pair should be within a threshold (measured as 10% of the length of the bounding box diagonal). Finally, the inner vertices of the S2 section of G_c^r are projected onto the corresponding S2 section of G_t^r. As shown in Figure 6(c), the resulting G_c^r is formed by joining the solved S1 sections and S2 sections. The proposed piecewise parameterization is continuous and visually smooth because all the vertices of G_c^r are sampled on G_t^r.

3.4. Motion blending

Motion blending is a powerful tool for reusing motion capture data to creating realistic human animations. When only the raw motion parameters are given, the input motions should be quite similar to each other in order to produce a reliable and feasible motion blending [KG03]. Unfortunately, existing mesh sequence representations cannot offer prior information about the motions other than joint parameters for each frame. Thus, the proposed framework mainly addresses similar input motions. We adopt the timewarping curve algorithm [GSKJ03] to find optimal frame pairs of two motions by comparing the similarity of two frames. The similarity is measured as the distances between corresponding joints. Then, the 'in-between' motion is obtained by interpolating the corresponding joint angles and bone lengths.

3.5. Dynamic shape interpolation via skeleton-driven cage-based DT

As a new application, we can achieve dynamic shape interpolation in a reconstruction–interpolation manner. Given the blended motion in terms of kinematic skeletons, its attached geometries will be constructed by the proposed DT. Then, both reproduced geometries with identical mesh connectivity in the similar pose are interpolated. Based on the extracted skeleton-driven cage, we propose a new spatial DT approach to ensure high efficiency and natural deformation for various morphing strategies. The skeleton sequence is a high-level representation of articulated motion. Thus, it is natural to adopt joint transformations as the deformation gradient to capture the variations in the poses. In this way, the blended motion will not suffer from the interference of geometric details.

The proposed DT method takes the following data as input: a rest source pose, a deformed source pose, a rest target pose and their corresponding skeleton-driven cages. In our method, a skeleton is represented in terms of its joint positions and is organized in a hierarchical tree structure. Each cage vertex is attached to its

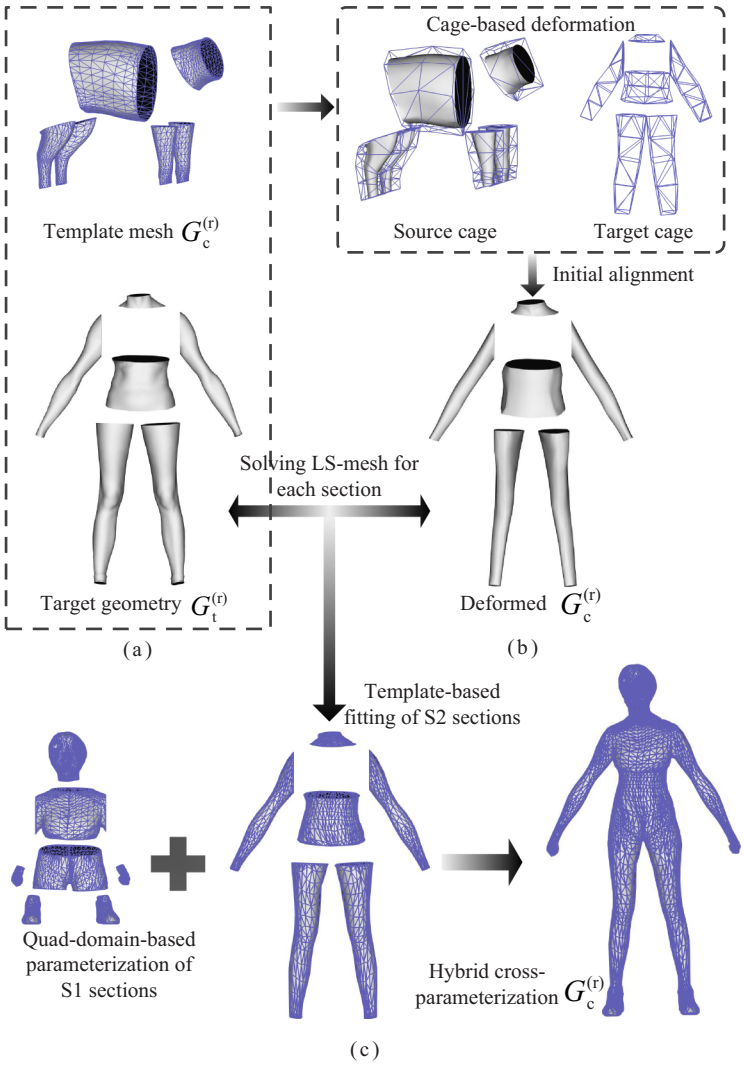

Figure 6: *Pipeline of local template-based fitting: approximate S2 sections of* G_c^r *to those of* G_t^r *as closely as possible. (a) Initial S2 sections of* G_c^r *and* G_t^r*. (b) Initial aligned* G_c^r *via the cage-based deformation. (c) The final compatible mesh* $G_c^{(r)}$ *(right) is composed of the parameterization resulting (left) and the solved LS-meshes (middle).*

corresponding joint according to the cage construction from the skeleton [CF14].

The first step is to generate the deformed target skeleton using the skeletal deformation gradient inferred from the source skeletons, as shown in Figure 7(a). For each joint, a local frame (**u**, **v**, **w**) is defined [CF14], where **u** is the bone direction from the parent joint to the current joint, **v** is a projection vector that the vector from the initial marker to the corresponding joint is projected onto the perpendicular plane of **u** and **w** is the cross product of **v** and **u**. Let $(\mathbf{u}_i^{(r)}, \mathbf{v}_i^{(r)}, \mathbf{w}_i^{(r)})$ and $(\mathbf{u}_i^{(d)}, \mathbf{v}_i^{(d)}, \mathbf{w}_i^{(d)})$ be the local frames at joint i of the rest source skeleton and deformed source skeleton. The corresponding joint transformation between $(\mathbf{u}_i^{(r)}, \mathbf{v}_i^{(r)}, \mathbf{w}_i^{(r)})$ and $(\mathbf{u}_i^{(d)}, \mathbf{v}_i^{(d)}, \mathbf{w}_i^{(d)})$ can be represented as a compound of two rotation transformations, i.e. $\mathbf{R}_{ui} \cdot \mathbf{R}_{vi}$, where $\mathbf{R}_{ui}$ and $\mathbf{R}_{vi}$ are used to align $\mathbf{u}_i^{(r)}$ to $\mathbf{u}_i^{(d)}$ and $\mathbf{v}_i^{(r)}$ to $\mathbf{v}_i^{(d)}$, respectively. Once the deformation descriptor, i.e. the compound rotation transformations of all joints of the source skeletons, is computed, the deformed target skeleton can be created by directly applying the joint transformations to the rest target skeleton. First, the deformed target skeleton is generated by applying the joint transformations between the rest source skeleton and deformed source skeleton to the rest target skeleton. Then, the cage deformation gradient is defined as relative vertex transformations in the joint frames between the rest source cage and deformed source cage. Finally, the rest target cage clones the source cage deformation gradient to fit the deformed target skeleton. Once the deformed target cage is reconstructed, the embedded geometry can be reproduced using the cage-based deformation. We can skip this step if the deformed target skeleton is generated via motion blending.

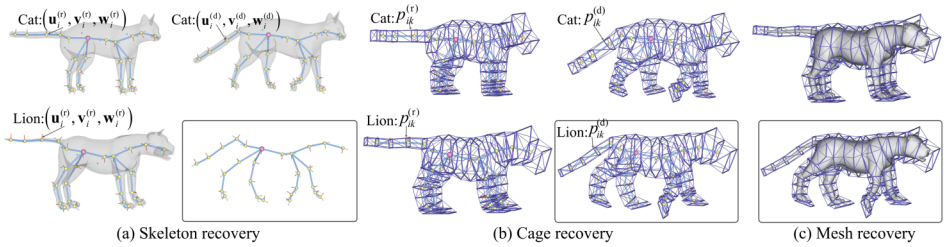

Figure 7: *Pipeline of skeleton-driven cage-based DT: transferring the source mesh deformation to the target mesh via skeleton and cage. (a) The target deformed skeleton is obtained by applying source joint transformations to the target rest pose skeleton. (b) The target deformed cage is obtained by cloning source cage vertex transformations in the corresponding joint frames to the target rest pose cage. (c) The target deformed mesh is reconstructed by MVC encoding.*

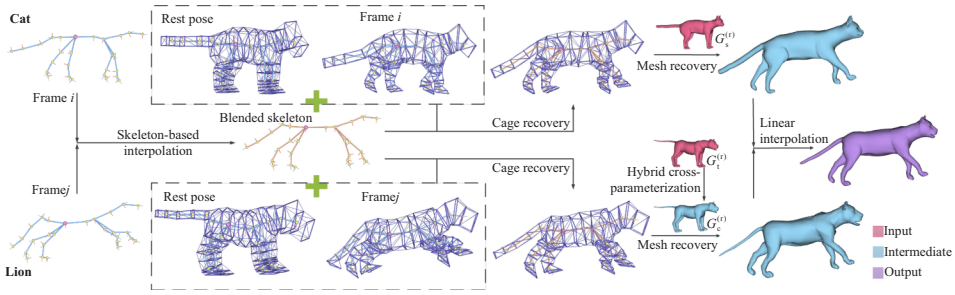

Figure 8: *Pipeline of shape interpolation based on motion blending. The brown skeleton is a frame generated by motion blending (left). Using the blended skeleton as a target skeleton, the cages of Cat and Lion are reconstructed for the target skeleton. Then, G_s^r and G_c^r are deformed by the reconstructed cages. Finally, the reproduced meshes of the blended skeleton are linearly interpolated to create an interpolated shape for that frame pair.*

The second step is to recover the cage from the deformed target skeleton (see Figure 7b). Considering that the cage can deform the embedded mesh non-rigidly while preserving geometric details, we mimic the relative variation in each vertex between the rest source cage and deformed source cage. For example, $p_{ik}^{(r)}$ and $p_{ik}^{(d)}$, which are attached to joint i, are two corresponding vertices of the rest source cage and deformed source cage, respectively. $\mathbf{T}_{vk}$ is the affine transformation from $p_{ik}^{(r)}$ to $p_{ik}^{(d)}$ in the local frame $(\mathbf{u}_i^{(d)}, \mathbf{v}_i^{(d)}, \mathbf{w}_i^{(d)})$. Therefore, the cage recovery process for the deformed target skeleton can be formulated as

$$q_{ik}^{(d)} = \mathbf{T}_{vk} \cdot \mathbf{R}_{ui} \cdot \mathbf{R}_{vi} \cdot q_{ik}^{(r)}. \tag{2}$$

Finally, the target embedded mesh can be reconstructed via the deformed cage with MVC encoding (see Figure 7c).

For the blended motion, we perform the proposed DT algorithm twice for both input motions with $G_s^{(r)}$ and $G_c^{(r)}$. As a result, the geometries of each frame pair in the blended motion have identical connectivity and similar poses. We employ simple linear interpolation to efficiently produce a natural dynamic morphing. Figure 8 shows the procedure of dynamic shape interpolation for one frame pair.

4. Results and Discussion

The proposed framework is implemented on a PC with an Intel Core 2 Quad 2.5 GHz CPU and 8 GB of memory. For the hybrid cross-parameterization approach, we test dissimilar articulated models with high resolutions to demonstrate its feasibility, robustness and performance. For the skeleton-driven cage-based DT approach, we test the data with large-scale deformations (a running Lion) and compare with the most related work [CF14]. Without motion blending, a galloping Camel generated from the data of [SP04] and a performance capture data with rich geometry details from multi-view silhouette reconstruction [VBMP08] are adopted to demonstrate dynamic shape interpolation between dissimilar models. For the entire morphing framework, we test three types of data to demonstrate its feasibility. The first type is a strolling Cat and a running Lion, the second type is a walking Woman and a jogging Man (see the accompany video V1) and the last type is a strolling Cat and a walking Woman with different skeletal structures. Each group should exhibit similar motions to satisfy the requirements of the motion blending method.

4.1. Hybrid cross-parameterization

The motivation of a hybrid cross-parameterization method is to naturally integrate with the skeleton-driven cage framework when only limited joint-like markers are available. Figure 9 shows the results of 3D shape morphing among three models based on the hybrid cross-parameterization. The performance of the test is summarized in Table 1. The results show that the proposed hybrid cross-parameterization is efficient and robust for meshes with different resolutions. Although the proposed approach keeps only C^0 continuity in theory, the reprojection mechanism contributes to produce

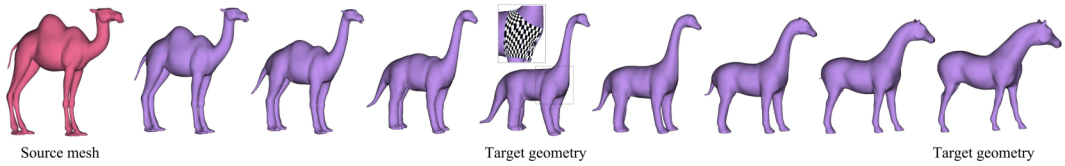

Figure 9: *A morphing sequence based on the hybrid cross-parameterization approach.*

Table 1: *Statistics of hybrid cross-parameterization method.*

Source	Target	Runtime (ms)			
Model (#V/#F) Cage (#V/#F) Marker (#)	Model (#V/#F) Cage (#V/#F) Marker (#)	*t*1	*t*2	*t*3	Total
Cat 7207/14 410 216/428 27	Lion 4556/9108 216/428 27	44	530	225	799
	Woman 25 172/50 340 220/436 23	95	875	971	1941
Camel 9770/19 536 236/468 32	Dino 23 982/47 960 236/468 32	92	999	854	1945
	Horse 80 660/161 316 221/438 28	236	3824	2211	6271
Man 10050/20096 202/400 22	Woman 25 172/50 340 202/400 22	112	938	3519	4569

Notes: *t*1: segmentation; *t*2: quad-based parameterization; *t*3: template-based fitting.

Table 2: *Morphing strategies.*

	Cat (*)	Lion (*)	X-cat
Strolling	**Input** Strolling Cat	**Strategy I** Strolling Lion	**Strategy II** Strolling X-cat
Running	**Strategy I** Running Cat	**Input** Running Lion	**Strategy II** Running X-cat
Moving	**Strategy III** Moving Cat	**Strategy III** Moving Lion	**Strategy IV** Moving X-cat

Notes: (*) indicates the rest pose mesh in the sequence; X-cat: interpolated shape; Moving: blended motion.

visually plausible, piecewise smooth cross-parameterization results, as shown in the zoomed-in window in Figure 9.

4.2. Morphing strategies

Because motion blending and shape interpolation can be independently processed, various combinations of them can be explored in the proposed framework. Consequently, we classify all the possible combinations into four strategies, as shown in Table 2, where the Cat/Lion sequence pair is used as an expressive illustration. The relative animations are shown in the accompany video V1.

4.2.1. *Strategy I—DT*

DT is the simplest one among the four strategies. It only involves one input sequence and two rest pose meshes. To evaluate the proposed skeleton-driven cage-based DT approach, we compare our results with the direct cage-based DT in [CF14] and the surface-based DT [SP04]. As shown in Figure 10, the proposed approach can generate natural results similar to the surface-based DT method, while the direct cage-based DT [CF14] suffers from artefacts because the deformation gradient used in [SP04] is defined on the cage so that geometric details of the embedded model cannot be adequately captured. As shown in Table 3, our approach is more efficient than the direct cage-based DT that contains a pre-factorization cost for solving an energy function. Benefiting from the cage structure, the proposed DT and the direct cage-based DT have advantage of computation efficiency, while the complexity of surface-based method [SP04] is in proportion to the mesh resolution. Here, we use hybrid cross-parameterization to establish correspondence for surface-based DT.

4.2.2. *Strategy II—Dynamic shape interpolation without motion blending*

This strategy is implemented by taking the motion of one input sequence as the 'blended motion' (described in Section 3.4). The shape gradually changes from one model into the other model while the model performs the motion of one input. Figure 11 shows several snapshots of dynamic shape interpolation from a galloping Camel into a galloping Dino.

4.2.3. *Strategy III—Motion blending without dynamic shape interpolation*

In our context, motion blending has two applications. One application is the interpolation of two input motions that creates the 'in-between' motion, as shown in Figure 12 (third row), where a

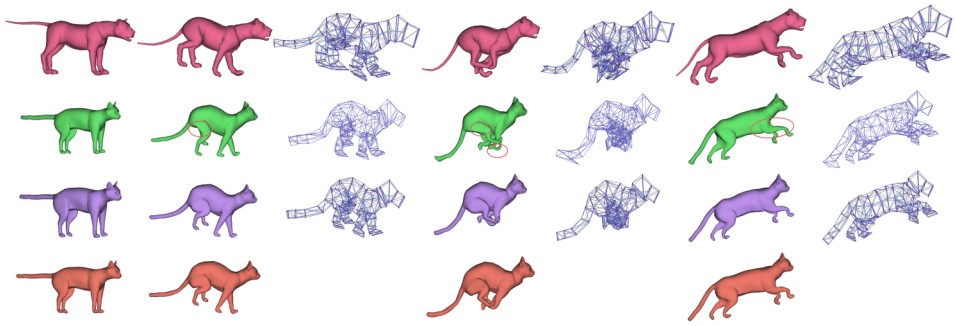

Figure 10: *Comparison of the proposed skeleton-driven cage-based DT with direct cage-based DT [CF14] and the surface-based DT [SP04]: source poses and their control structures (top row), results of [CF14] using only cages (second row), our results using both skeletons and cages (third row) and results of the surface-based method [SP04] (fourth row).*

Table 3: *Runtime comparison of the example in Figure 10 (in ms).*

Method	Pre-process		DT		Total
Our DT		–		Transfer	
	Scage		Scage sequence exaction	28*24	5056
[CF14]	3976	Pre-factorization	17*24	Transfer	
		1131		69*24	7171
[SP04]*	Correspondence	Pre-factorization	Transfer		
	4775	23 233	16*24		28 392

Notes: the correspondence in [SP04]* is built through our hybrid cross parameterization.

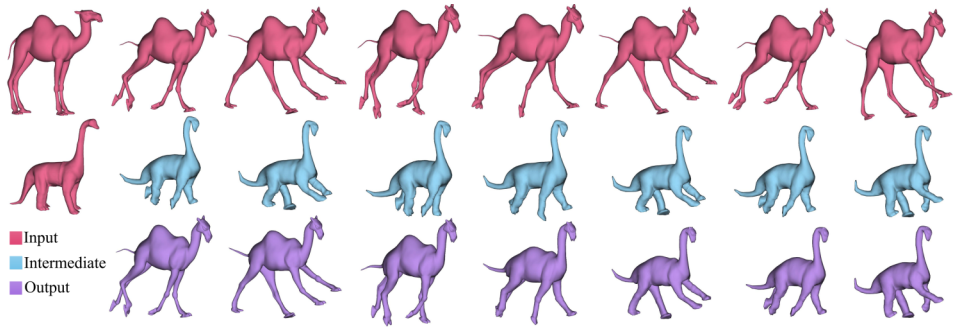

Figure 11: *A galloping Camel gradually changes into a galloping Dino. The Dino sequence (indicated in blue) is obtained by transferring the deformations of the Camel to the compatibly remeshed Dino in the rest pose.*

half-and-half mixture of strolling and running is demonstrated. The other one is blending partial frames to create a transition between two motions (see the accompany video V1 for a clear transition effect).

4.2.4. *Strategy IV—Dynamic shape interpolation with motion blending*

Strategy IV is a full version of mesh sequence morphing, including both shape interpolation and motion blending. Figure 1 and the fourth row in Figure 12 show several clips of morphing results.

Efficiency is an important factor in the design of a framework for processing mesh sequences. Strategy IV can be regarded as a combination of Strategies II and III; therefore, we only list the statistics of Strategy IV in Table 4. Mesh sequence morphing consists of two main steps: a pre-processing step and a morphing step. There are three substeps in the pre-processing step: the generation of skeleton-driven cages for $G_s^{(r)}$ and $G_t^{(r)}$, the proposed hybrid cross-parameterization for $G_c^{(r)}$ and the MVC computations for $G_s^{(r)}$, $G_t^{(r)}$ and $G_c^{(r)}$. The morphing step includes four substeps: the extraction of skeleton-driven cages, the blending of input motions, the mesh reconstruction for blended skeletons by our DT and the

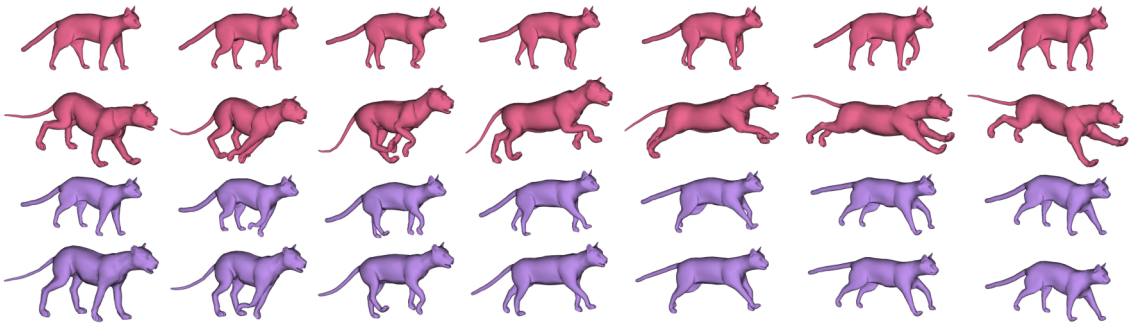

Figure 12: *An illustration of Strategies III and IV: motion blending without shape interpolation (third row) and with shape interpolation (bottom row).*

Table 4: *The running time of Strategy IV.*

		Pre-process (ms)			Pipeline (ms)				
Mesh sequence	Frame	Scage G_s^r/G_t^r	MVC $G_s^r/G_t^r/G_c^r$	HCP	Scages extraction M_s/M_t	MB	DT Source/Target	SI	Total (ms)
Cat & Woman	24/48	923/2619	2119/8145/2119	1941	1372/5686	38	1368/2575	199	29 104
Cat & Lion	24/24	765/476	1932/1714/1932	799	900/797	30	900/971	117	11 333
Man & Woman	48/48	1012/2431	2880/6518/2880	4569	1931/4914	55	2696/2678	362	32 926

Notes: Scage, skeleton-driven cage; HCP, hybrid cross-parameterization; MB, motion blending; SI, shape interpolation.

interpolation of shapes. Note that the complexities of the algorithms in the pre-processing step are related to the resolution of the input model. Thus, the proposed approach can efficiently generate a morphing sequence in motion and geometry within dozens of seconds.

4.3. Discussions and limitations

Each step in the proposed framework involves a related technique in computer graphics. However, a simple combination of state-of-the-art methods, such as mesh sequence representation, cross-parameterization, motion blending and DT methods, cannot address the problem of mesh sequence morphing. A flexible representation of a mesh sequence is essential for bridging these independent techniques in a unified framework. For example, it is cumbersome to achieve dynamic shape interpolation by performing cross-parameterization frame by frame. Second, various interaction requirements for shape morphing and DT must be unified to enable an efficient framework and minimize user interactions. Therefore, we choose the skeleton-driven cage to encode mesh sequences because of its compatibility with motion blending methods and low computation cost. Meanwhile, we need to elaborately devise cross-parameterization and DT to apply to the uniform interface and address all morphing strategies.

Skeleton-driven cage-based DT is a foundational technique in the proposed framework, which consists of three steps: skeleton recovery, cage recovery and mesh recovery. The cage is an essential media bridging the skeleton and the embedded geometry. Skeleton-driven cage-based DT is a foundational technique in the proposed framework, which consists of three steps: skeleton recovery, cage recovery and mesh recovery. The cage is an essential media to link the skeleton and the embedded geometry. Intuitively, the embedded mesh should be directly driven by the skeleton, e.g. via linear blend skinning (LBS). The concept of skeleton-driven cage is inspired by Ju *et al.* [JZvdP*08], where the use of cages can efficiently avoid the artefacts associated with LBS. In this paper, the mesh sequence morphing framework is built on the cage-based mesh sequence representation due to its simplicity and efficiency. In future, we would like to attempt to test the skeleton-based mesh sequence framework [DATTS08, dAU14, LD14] with skeleton-based DT to solve the proposed morphing problem.

The proposed framework is subject to a few limitations. (1) The initial construction of the skeleton-driven cage requires a standard rest pose and is not effective for regions with large geodesic distance but small Euclidean distance as a result of the MVC encoding. Thus, the Camel model at the rest pose in Figure 9 is manually adjusted in the regions around the knees. This problem can be solved by replacing with other non-negative linear embedding, such as Harmonic coordinates [JMD*07], at the cost of higher computational complexity. Here, the skeleton is determined based on the cross-sections so that sheet-like models are not restricted. However, cup-like models cannot be handled because the concave shape cannot be well captured by the control structure. Thus, the proposed mesh sequence morphing technique mainly focuses on articulated models. (2) The applicability of the hybrid cross-parameterization method is almost equivalent to the domain-based parameterization method [KS04] and the template-based fitting method [ZLJW06] for models with similar skeletal structure, but input models with different genus are beyond the capacity of it. (3) There are two requirements for input

sequences. First, each mesh sequence should have the same mesh connectivity, which is necessary to encode the mesh sequence. Second, the motions of two sequences should be similar on the whole, which is a pre-requisite for motion blending methods to generate a plausible blended motion as mentioned in [KG03].

5. Conclusion

In this paper, a novel morphing concept for mesh sequences is proposed and investigated. We present an efficient framework for two mesh sequences to recreate plausible mesh sequences by simultaneously blending motions and interpolating shapes. Furthermore, the framework can be used to exploit all possible mesh sequences using various morphing strategies. To enable an universal framework, a hybrid cross-parameterization method is elaborately designed for shape interpolation, and a skeleton-driven cage-based DT method is introduced to integrate motion blending and shape interpolation. In addition, both methods can seamlessly incorporate with the skeleton-driven cage representation of mesh animations to support an efficient framework with limited user interactions.

In future research, we will integrate advanced techniques into the proposed framework to expand its applicability and address the discussed limitations. For example, we could utilize bone merging to obtain compatible skeletons to handle models with different genus and we will also explore motion blending for dissimilar motions.

Acknowledgements

The authors would like to thank the anonymous reviewers who gave valuable suggestions to improve the quality of the paper. This work was supported by the National Natural Science Foundation of China under Grant Nos. 61170138 and 61472349.

References

[ACP03] Allen B., Curless B., Popović Z.: The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Transactions on Graphics 22*, 3 (2003), 587–594.

[Ale02] Alexa M.: Recent advances in mesh morphing. *Computer Graphics Forum 21*, 2 (2002), 173–198.

[ATC*08] Au O. K.-C., Tai C.-L., Chu H.-K., Cohen-Or D., Lee T.-Y.: Skeleton extraction by mesh contraction. *ACM Transactions on Graphics 27*, 3 (2008), 44:1–44:10.

[BCWG09a] Ben-Chen M., Weber O., Gotsman C.: Spatial deformation transfer. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2009), ACM, pp. 67–74.

[BCWG09b] Ben-Chen M., Weber O., Gotsman C.: Variational harmonic maps for space deformation. *ACM Transactions on Graphics 28*, 3 (2009), 34:1–34:11.

[BW95] Bruderlin A., Williams L.: Motion signal processing. In *SIGGRAPH '95 Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), ACM, pp. 97–104.

[CF14] Chen X., Feng J.: Adaptive skeleton-driven cages for mesh sequences. *Computer Animation and Virtual Worlds 25*, 3–4 (2014), 447–455.

[CHSB10] Chen L., Huang J., Sun H., Bao H.: Cage-based deformation transfer. *Computers & Graphics 34*, 2 (2010), 107–118.

[CO09] Cohen-Or D.: Space deformations, surface deformations and the opportunities in-between. *Journal of Computer Science and Technology 24*, 1 (2009), 2–5.

[DATTS08] De Aguiar E., Theobalt C., Thrun S., Seidel H.-P.: Automatic conversion of mesh animations into skeleton-based animations. *Computer Graphics Forum 27*, 2 (2008), 389–397.

[dAU14] De Aguiar E., Ukita N.: Representing mesh-based character animations. *Computers & Graphics 38*, (2014), 10–17.

[Flo03] Floater M. S.: Mean value coordinates. *Computer Aided Geometric Design 20*, 1 (2003), 19–27.

[GLHH13] Gao L., Lai Y.-K., Huang Q.-X., Hu S.-M.: A data-driven approach to realistic shape morphing. *Computer Graphics Forum 32*, 2pt4 (2013), 449–457.

[GSKJ03] Gleicher M., Shin H. J., Kovar L., Jepsen A.: Snap-together motion: Assembling run-time animations. *ACM Transactions on Graphics 22*, 3 (2003), 702–710.

[JMD*07] Joshi P., Meyer M., DeRose T., Green B., Sanocki T.: Harmonic coordinates for character articulation. *ACM Transactions on Graphics 26*, 3 (2007).

[JSW05] Ju T., Schaefer S., Warren J.: Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics 24*, 3 (2005), 561–566.

[JZvdP*08] 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 27*, 5 (2008), 122:1–122:10.

[KG03] Kovar L., Gleicher M.: Flexible automatic motion blending with registration curves. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 214–224.

[KG08] Kircher S., Garland M.: Free-form motion processing. *ACM Transactions on Graphics 27*, 2 (2008), 12:1–12:13.

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM Transactions on Graphics 21*, 3 (2002), 473–482.

[KS04] Kraevoy V., Sheffer A.: Cross-parameterization and compatible remeshing of 3D models. *ACM Transactions on Graphics 23*, 3 (2004), 861–869.

[LD14] Le B. H., Deng Z.: Robust and accurate skeletal rigging from mesh sequences. *ACM Transactions on Graphics 33*, 4 (2014), 84:1–84:10.

[LLCO08] Lipman Y., Levin D., Cohen-Or D.: Green coordinates. *ACM Transactions on Graphics 27* (2008), 78:1–78:10.

[LLN*14a] Liao J., Lima R. S., Nehab D., Hoppe H., Sander P. V.: Semi-automated video morphing. *Computer Graphics Forum 33*, 4 (2014), 51–60.

[LLN*14b] Liao J., Lima R. S., Nehab D., Hoppe H., Sander P. V., Yu J.: Automating image morphing using structural similarity on a halfway domain. *ACM Transactions on Graphics 33*, 5 (2014), 168:1–168:12.

[LLX*15] Lu H., Li G., Xian C., Zhang Z., Yin M.: EC-CageR: Error controllable cage reverse for animated meshes. *Computers & Graphics 46*, (2015), 138–148.

[LSP08] Li H., Sumner R. W., Pauly M.: Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum 27*, 5 (2008), 1421–1430.

[Per95] Perlin K.: Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics 1*, 1 (1995), 5–15.

[SAPH04] Schreiner J., Asirvatham A., Praun E., Hoppe H.: Inter-surface mapping. *ACM Transactions on Graphics 23*, 3 (2004), 870–877.

[SCO04] Sorkine O., Cohen-Or D.: Least-squares meshes. In *Proceedings of the Shape Modeling International 2004* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 191–199.

[SP04] Sumner R. W., Popović J.: Deformation transfer for triangle meshes. *ACM Transactions on Graphics 23*, 3 (2004), 399–405.

[TTB12] Thiery J.-M., Tierny J., Boubekeur T.: Cage R: Cage-based reverse engineering of animated 3D shapes. *Computer Graphics Forum 31*, 8 (2012), 2303–2316.

[VBMP08] Vlasic D., Baran I., Matusik W., Popović J.: Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics 27*, 3 (2008), 97:1–97:9.

[WP95] Witkin A., Popovic Z.: Motion warping. In *SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), ACM, pp. 105–108.

[WPZ*11] Wu H.-Y., Pan C., Zha H., Yang Q., Ma S.: Part-wise cross-parameterization via nonregular convex hull domains. *IEEE Transactions on Visualization and Computer Graphics 17*, 10 (2011), 1531–1544.

[XZY*07] Xu W., Zhou K., Yu Y., Tan Q., Peng Q., Guo B.: Gradient domain editing of deforming mesh sequences. *ACM Transactions on Graphics 26*, 3 (2007).

[YCJL09] Yao C.-Y., Chu H.-K., Ju T., Lee T.-Y.: Compatible quadrangulation by sketching. *Computer Animation and Virtual Worlds 20*, 2–3 (2009), 101–109.

[YLSL11] Yeh I.-C., Lin C.-H., Sorkine O., Lee T.-Y.: Template-based 3D model fitting using dual-domain relaxation. *IEEE Transactions on Visualization and Computer Graphics 17*, 8 (2011), 1178–1190.

[YXF14] Yang L., Xiao C., Fang J.: Multi-scale geometric detail enhancement for time-varying surfaces. *Graphical Models 76*, 5 (2014), 413–425.

[YY12] Yoshiyasu Y., Yamazaki N.: Detail-aware spatial deformation transfer. *Computer Animation and Virtual Worlds 23*, 3–4 (2012), 225–233.

[ZLJW06] Zhang L., Liu L., Ji Z., Wang G.: Manifold parameterization. Lecture Notes in Computer Science 4035 (2006), pp. 160–171.

[ZRKS05] Zayer R., Rössl C., Karni Z., Seidel H.-P.: Harmonic guidance for surface deformation. *Computer Graphics Forum 24*, 3 (2005), 601–609.

[ZXTD10] Zhou K., Xu W., Tong Y., Desbrun M.: Deformation transfer to multi-component objects. *Computer Graphics Forum 29*, 2 (2010), 319–325.

Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

Video V1