Noname manuscript No.
(will be inserted by the editor)

Shape Exploration of 3D Heterogeneous Models based on Cages

Weiliang Meng · Jianwei Guo · Xavier Bonaventura · Mateu Sbert · Xiaopeng Zhang

Received: date / Accepted: date

Abstract Shape exploration of 3D heterogeneous models is essential for movie and special effects in 3D animation and games. As heterogeneous models have different numbers of vertices and topology, the mapping between source and target model maybe ambiguous for deformation transfer. We propose a new framework for heterogeneous models shape exploration based on cages, which provides a feasible and fast solution for this open problem. Using a public cage as the intermediate medium, the deformation of the source models can be denoted as the position changing of the cage. When applying the cage change to the target model, rough deformation transfer can be achieved. After an optimization and interpolation to generate the explored shape of the heterogeneous target model, animation can be acquired. Our method is not only suitable for triangle meshes, but also for quadrilateral meshes or any other type of meshes. We demonstrate the validity of our scheme by a series of shape exploration experiments for different models.

Weiliang Meng
NLPR-LIAMA, Institute of Automation, CAS, China
E-mail: weiliang.meng@ia.ac.cn

Jianwei Guo
NLPR-LIAMA, Institute of Automation, CAS, China
E-mail: jianwei.guo@nlpr.ia.ac.cn

Xavier Bonaventura
Graphics and Imaging Laboratory, University of Girona, Spain
E-mail: xavier.bonaventura@udg.edu

Mateu Sbert
Graphics and Imaging Laboratory, University of Girona, Spain
Tianjin University, China
E-mail: mateu@ima.udg.edu

Xiaopeng Zhang
NLPR-LIAMA, Institute of Automation, CAS, China
E-mail: xiaopeng.zhang@ia.ac.cn

Keywords deformation transfer · cage coordinates · differential coordinates · shape space · interpolation

1 Introduction

Shape exploration for 3D heterogeneous models is an open and core modeling problem in computer modeling and animation, especially in film special effects. In order to accomplish shape exploration, mesh deformation is usually employed to generate complex model states. For artists, this may take tremendous amount of artistry, skill, and time to craft the vivid process. In order to reuse a deformation created for one shape to infer the other one, many researches often use specific parameters to adapt the deformation to the new shape. In most cases, adapting the parameter values is a time consuming process which may start from scratch. On the other hand, although current software such as Maya and 3D Max can interpolate models automatically, those methods are only suitable for isomorphic models, i.e. all the models must have the same number of vertices and the same topology. When in the case of heterogeneous models that have different vertices and different topology, current methods cannot deal with them, especially for the case that when the source models are triangle meshes and the target models are quadrilateral or other totally different type meshes. Although deformation methods for specific purposes exist, an automatic adaptation method designed for one type of deformation may fail to apply to the other models, and hand-sculpted alterations are often included. As a result, the methods are not universally applicable in practice.

In this paper, we propose a new framework to generate new poses between heterogeneous models. We consider the following problem: given two source models which represent two states of the same object such as a standing horse model and a running horse model, and another target model like a camel model, how to generate a series of models for the target model that roughly reflects the two source models variation process? Note that the target model may have different vertices numbers and even different topology with the source models, i.e., they are heterogeneous models. Our idea is based on the following hypotheses: (1) the target explored model may have no exact transfer result compared to the source models, as the shape could have great difference, such as bird vs. camel. (2) we only need the target deformed model to have similar but not exact deformation. This demand is natural for many movie special effects on the heterogeneous models.

In order to achieve such result, we use a public cage as the intermediate medium to implement the transfer process. Specifically, firstly we align the source and target models based on PCA (Principal Component Analysis [30]), and then we use a public polyhedron (called cage) to obtain the equations coefficients of a linear system based on the vertices positions of the source model and target model, while these corresponding coefficients (cage coefficients or CC) are preserved during the whole processing. Then, the source deformed

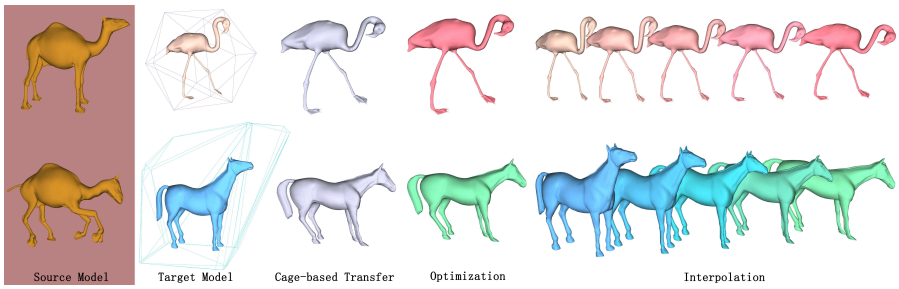

Fig. 1 Cage-based shape exploration process. Given two source models with different states, our purpose is to apply the deformation to target models to explore new states of them. Here two target models (flamingo and horse) are shown as examples. Firstly, a public cage is used as the deformation transfer medium where the cage can be different according to the target model. Here for the flamingo target model we use the icosahedron as the public cage on both the source models, while for the horse target model the public cage is a custom-made cage. Second, the cage-based transferred results for the target models are obtained, and optimization is executed to generate the optimized results in order to keep the local detail as untouched as possible. Finally, an interpolation algorithm based on shape space is employed to generate more shape results, which can be used for animation.

model is used to define the vertices deformed positions of the cage and to compute the deformed target models based on the linear system of equations and CC. In addition, the differential coordinates of the target model are used to optimize the deformed target model. Finally, we obtain the interpolated models based on shape space to generate an exploration sequence for animation. Translation, rotation and scale operation may be performed before the interpolation stage according to users need to generate more flexible results.

Our method avoids time-consuming parameters adjustment, and can generate new explored shapes in a few seconds for 3D heterogeneous models. The contributions of our paper can be listed as follows:

- A shape exploration generation framework for heterogeneous models based on two isomorphic source models.
- A cage-based deformation transfer method with a cage selection guide that is suitable for heterogeneous models.
- An optimization of the transferred-deformation model for local detail p-reservation.

Compared with existing deformation transfer methods, we avoid the exact mapping process between source and target models and may not achieve exact deformation transfer. Although Benchen et al.[4]'s method does not need the explicit mapping either, their deformation result must deform and adjust the model continually until reaching the expecting result, while our method can generate the deformed result induced by the source models and obtain the predicting result directly. On the other hand, as for many heterogeneous models there is no exact mapping between them, our method provides a feasible manner to approximate the transformation of any heterogeneous target model according to source models. The whole process can be fully automatic, with no

manual interaction. Figure 1 shows the cage-based shape exploration process. In Figure 1, we use two camel models as the source models and finish the shape exploration on flamingo model and horse model respectively with different cages used. Figure 2 shows our processing flowchart. Using our method, we can fully-automatically explore shapes for heterogeneous models (such as using quadruped models to fully-automatically explore bipedal models' poses in Figure 1), and this cannot be done by the previous work as far as we know.

2 Related Work

2.1 Cage-based Deformation

Model deformation has been studied for more than twenty years. In order to obtain high-quality shape-preserving deformations, selected handles are used including points [5], lines [2] or bones [38], and polygon grids [28]. By modifying the positions and orientations of handles interactively, an intuitive deformation can be achieved. Most mentioned methods above heavily rely on optimization at pose time, making them too slow for deforming 3D objects.

In recent years, cage-based deformation methods receive a wide range of interest, cage vertices are seen as handles for deformation. Using a linear weighted blend of handle transformation, the computation can be fast for new model generation and the interaction with handles can lead to real-time deformation. In order to make the deformation smooth, reasonable weights should be chosen in a cage-based method to keep the mathematical continuity. Some feasible methods have been proposed including Mean Value Coordinates (MVC) [10, 16,19,11,24], Harmonic Coordinates (HC) [9,18], Green Coordinates (GC) [25], and complex barycentric coordinates (CBC) and its variants [37,4]. Once the weights for cages are obtained in pre-computation, real time deformations can be achieved based on GPU computation as these weights keep fixed and can be stored as textures for further accessing used for general purpose computation [26] during the whole deformation.

Ju et al. [20] use skinning templates to define common deformation behaviors for common joint types, which allows skinning solutions to be shared and reused. The skinning templates use cage-based deformations, and many possible alternatives for the skinning behavior of a 3D character can be quickly explored. Jacobson et al. [17] develop bounded biharmonic weights that produce smooth and intuitive deformations using points, bones and cages of arbitrary topology by minimizing the Laplacian energy subject to bound constraints. Thiery et al. [34] convert animated 3D shape sequences into compact and stable cage-based representations, and faithfully reconstruct the enclosed object deformation. García et al. [13] propose using *Cages (star-cages) to preserve the smoothness of the mesh in the transitions between them, and the usage of multiple cages is allowed to enclose the model for easier manipulation. The proposed deformation scheme is flexible and versatile, and heterogeneous sets of coordinates can be used for different levels of deforma-

tion, either for local or global deformation. Li et al. [23] present closed-form formula as generalized barycentric coordinates for boundary constrains represented as polynomials up to degree 3, mainly interpolating both boundary values and gradients over a 2D polygonal domain.Zhang et al. [41] propose local barycentric coordinates (LBC), which select for each interior point a small set of control points and satisfy common requirements on barycentric coordinates, such as linearity, non-negativity, and smoothness. LBC provides more local and finer control on shape deformation than previous approaches. Wang et al. [36] propose a method to design linear deformation subspaces, unifying linear blend skinning and generalized barycentric coordinates, but this method requires discretization of the input domain which is difficult for 3D models.

Above all, most current cage-based deformation methods are mainly used for free interactive model deformation to deform models with similar appearance, and the deformed results are mainly dependent on the ceaseless adjustment for acceptable results.

2.2 Deformation transfer

Our work can be seen as a kind of general deformation transfer, without keeping an exact mapping between the source and target models, because heterogeneous models maybe have no unambiguity mapping between each others. Deformation transfer applies the deformation exhibited by a source mesh onto a different target mesh. In most cases, the source model and the target model have similar shape, even the same topology. Sumner et al. [33] propose a general transfer method for triangle meshes. The correspondence map is built by the user using a small set of vertex markers, and an optimization problem is solved to consistently apply the transformations to the target shape. The conspicuous limitation is that the source and the target meshes should be grossly similar. Baran et al. [1] propose semantic deformation transfer which infers a correspondence between the shape spaces of the two characters, leading to automatic transfer of new poses and animations, but seam artifacts may appear when the interpolated rotation of a face differs significantly. Ben-Chen et al. [3] propose the space deformation transfer method, which can be applied to a variety of shape representations including tet-meshes, polygon soups and multiple-component meshes. Given a sparse set of user-selected correspondence points between the source and target models, and by deforming the space where the shape is embedded based on a set of harmonic basis functions, the deformed target shape can be generated with similar properties to the source deformation. Chen et al. [7] present a cage-based method for transferring animation from a mesh sequence or motion capture data to geometric models invariant representations, but the deformation result mainly depends on the shape and tessellation of the cage. Zhou et al. [43] automatically computes spatial relationships between components of the target object and transfers deformation of the source onto the target while preserving cohesion between the targets components. Zhao et al. [42] perform the transfer process between

the dual meshes of the source and target models and can deal with meshes with poor sampling and complex shape, but the specific markers must be added by users. Yoshiyasu et al. [40] present a deformation transfer method that is applicable to multi-component objects for transferring fine-scale deformations. However, they require the source model be single-component.

Most methods need to construct a relatively exact correspondence relationship between the source and target model, which makes the process complex and not suitable for heterogeneous models. Ma et al. [27] design a transfer method by computing source-to-target analogies based on the geometric differences between source and target models, and achieves the desired transfer effect on exemplar.But the shape analogy problem maybe ill-posed as there may be multiple possible analogy solutions in some cases. This paper provides a new solution for this situation.

2.3 Differential coordinates

The differential coordinates of the model encapsulate the local normal direction and local mean curvature together, and can be seen as a discretization of the continuous Laplace-Beltrami operator from a differential geometry perspective [6]. The conception of differential coordinates of the model vertices was firstly proposed by Sorkine et al. in [32], and is used to preserve the high frequency details for encoding meshes. Other relative applications of differential coordinates include spectral methods [21] for obtaining compact 3D model representations, ROI (the region of interest) editing on meshes [31], surface reconstruction [8] and so on. We will use differential coordinates to optimize the generated target models, in order to keep the local detail after the deformation.

2.4 Space-time blending and model interpolation

In order to achieve shape exploration, intermediate models should be obtained between two given models. One way to implement space-time model blending is using implicit functions. This is very effective for heterogeneous models. Turk et al. [35] combine creating implicit functions and interpolating together, making the transformation appear smooth and natural for objects with different topologies. However, the models generated by this class of methods are too smooth because of the properties of the function, and cannot obtain sharp parts, making it to appear unreal compared to the ground truth.

On the other hand, as the transformed target model and the target model are isomorphic, model interpolation methods can be used to generate intermediate models. Linear method is a feasible way when the deformation is small. For complex exploration, using shape space to compute the intermediate models can achieve better results based on the Riemannian geometry. Kilian et al. [22] design the metrics based on the shape space metrics, and propose a framework of model interpolation for multiresolution models. Winkler et al. [39]

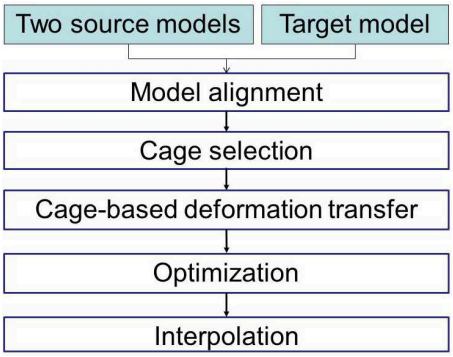

Fig. 2 The flowchart of our framework.

interpolate the local intrinsic properties of the mesh (edges and dihedral angles) using a hierarchical shape matching approach to generate models, and their mesh representation actually provides a new kind of shape space. But employing a complicated hierarchical shape matching technique prevents their method for directly using in a MeshIK system. Fröhlich et al. [12] explore the shape space through a geometrically meaningful interpolation and extrapolation technique, and the interpolation operator can be combined with previous deformation technique.

3 Cage-based Shape Exploration Framework

Given two isomorphic source models with the same topology but different poses and another heterogeneous target model, our goal is generating exploration results for the target model which will imitate the two source models transition process. The overview of the process is shown in Figure 1 and Figure 2.

The whole algorithm mainly includes 4 steps: model alignment, cage selection and cage-based deformation transfer, optimization, and interpolation based on shape space. If the initial states of the source models and the target models satisfy a coarse corresponding relation as shown in Figure 1, model alignment can be omitted.

For clearly description, we introduce the following notations for convenience.

S_0 the known source model, view as the initial state
S_1 the known source model, view as the final state
T_0 the known target model, view as the initial state
C_p the known public cage, view as the initial state of cage
cc cage coordinates
cc_s cage coordinates generated from the S_0, C_p
cc_t cage coordinates generated from the T_0, C_p
C_{dp} the unknown deformed public cage, generated from S_0, S_1, C_p
T_1 the unknown transferred target model, deformed from T_0 according S_0, S_1, C_p, C_{dp}

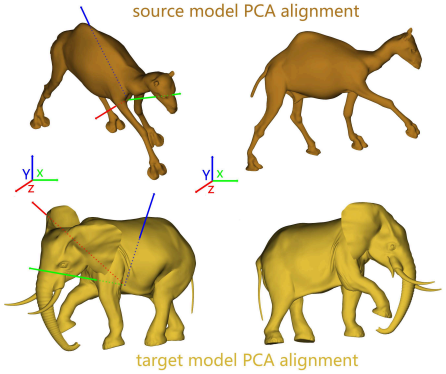

Fig. 3 Model alignment. After using PCA for the source and target model and rotating according to the eigenvectors, they can have a rough corresponding relationship, i.e., both heads of the model have the similar position as shown on the right.

Algorithm 1 Computation process

1: Read two isomorphic source meshes S_0, S_1 and another target model T_0.
2: **Model alignment**: constructing the correspondence between S_0 and T_0 and using the same transformation on S_1 based on PCA.
3: **Cage Selection and Cage-based deformation transfer:**.
4: Choose a suitable public cage C_0(user defined or the custom-made);
5: Compute the cage coordinates cc_s of S_0 on C_p ;
6: Compute the deformed public cage C_{dp} based on S_0, S_1, cc_s;
7: Compute the cage coordinates cc_t of T_0 on C_p ;
8: Compute the deformed target model T_1 based on cc_t, C_p, C_{dp} ;
9: **Optimization:** optimize T_1 to get T_{op} based on the local detail of T_0;
10: **Interpolation:** obtain the transitional models between T_0 and T_{op} based on shape space.
11: done.

T_{op} the optimized model of T_1

The pesudo-code for the framework is given by Alg.1. We will given the details in the following sections for the those **bold** computation process in the algorithm.

3.1 Model alignment

The aim of model alignment is to construct a coarse correspondence relationship between the source models(S_0, S_1) and the target model(T_0). As the target model is heterogeneous with the source models, which means having different vertices numbers, different genus, or different edges of each face, the correspondence relationship is fuzzy, and may not have the exact point-to-point map. Here we just try to make some coarse correspondence to generate a reasonable result from the the visual point of view. We use Principal Component Analysis(PCA)[30] to get the three eigenvectors of the vertices position matrix,

constructed with all the position information of the model. For a model with n vertices, the matrix is $n \times 3$ size.

Figure 3 shows the cross-reference model state before and after the alignment based on PCA. We can see that the source model and the target model can have a good match compared to the initial state. Of course the user can rotate the model for his/her own purpose to construct the corresponding relationship.

3.2 Cage Selection and Cage-based deformation transfer

The purpose of our method is not to transfer the deformation on the target model T_0 exactly, as for heterogeneous models there is no exact correspondence relationship between S_0 and T_0 because they may have different number of vertices, genus, and topology. For example, the flamingo model and the horse model have different legs, and we cannot make a mapping between these legs. Although Ovsjanikov et al. [29] design an mapping method between pairs of shapes that generalizes the standard notion of a map as a pairing of points, their method is suitable to the isometric shape matching benchmark but not to this case. In order to effectively transfer the deformation from the source model to the target model, traditional methods need to construct a point-to-point map and compute the transferred position according to the source model. But they cannot work here, because the mapping is difficult to obtain. As our goal is to approximate the change process of the source models using the target model and to generate dynamic changed effects for the target model, we do not need to implement exact mapping for these heterogeneous models. Based on this practical requirement, we will use a public cage C_p as the intermediate medium to transfer the deformation.

Cage selection. The cage C_p is a triangle mesh in 3D, whose shape can be any arbitrary polyhedron, such as a simple bounding box, a discretized sphere or even another model with relative few vertices, usually no more than several hundred vertices. On the other hand, the cage should be able to enclose the model, as discontinuity exists on the boundary of the cage for some cage-based deformation methods (such as GC-based deformation). We can magnify the cage to make sure all the parts of the model fall in the cage. Both the source models and target models use the same cage, so it will be better that the contour of the cage is similar to the shape of both source and target models. If the source and target models have great difference in appearance, we can use universal shapes such as cube or sphere (with limited number of vertices) as the cage. Figure 4 shows some general cages that can be used for deformation.

In addition, searching a rough close shape to the model seems difficult, while previous methods usually use manually modeling to generate a custom-made cage for the deformation. Here we provide an automatic way to obtain a custom-made cage for the model: the original model is firstly simplified into a mesh with fewer vertices and faces based on [15] or [14]. then the convex hull is computed. If we use the original model to generate the convex hull

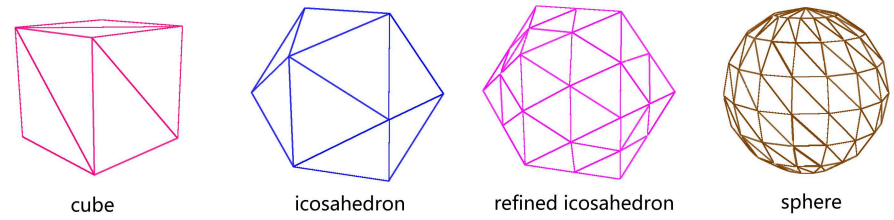

Fig. 4 General cage examples for deformation transfer. From left to right, the vertex number are 8, 12, 42 and 114 respectively.

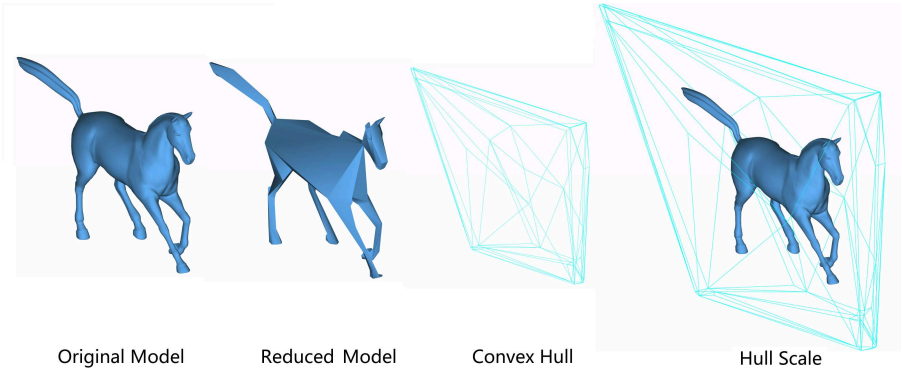

Fig. 5 Custom-made cage generation process. The vertex number of the cage is 40.

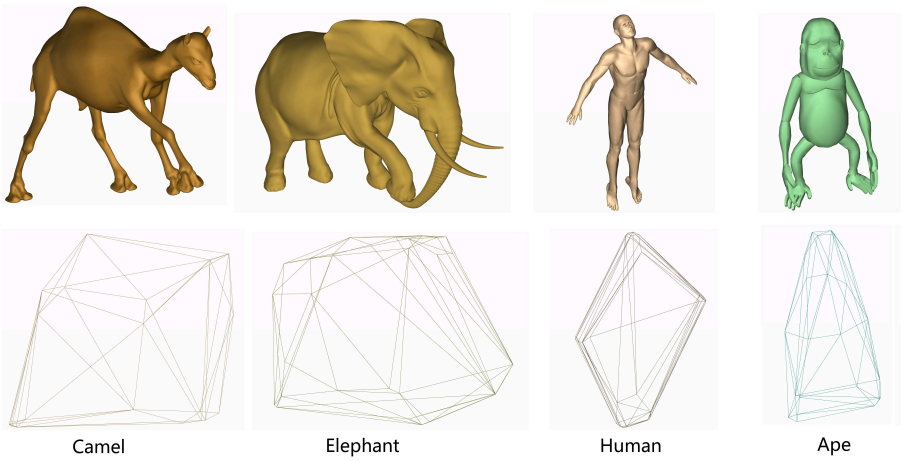

Fig. 6 Custom-made cages for different models. From left to right, the vertex number of the cages are 23, 28, 32 and 24 respectively.

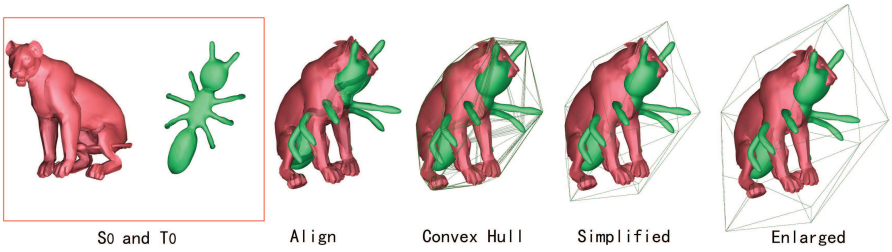

Fig. 7 Using S_0 and T_0 to generate the public cage. This method is more useful for S_0 and T_0 with great difference. Here the vertices number of the cage is 12 and the faces number is 20.

directly, the convex hull will have too many vertices and thin triangles, which are troublesome for further processing. After the simplification, the convex hull needs to be enlarged in order to make sure it can enclose the original model. So we scale the convex hull properly, and use the final convex hull as the custom-made cage. Figure 5 shows the custom-made cage generation process by our method,and Figure 6 shows a series of custom-made cages for different models that will be used for further deformation. Note that although we can generate custom made cages both for the source model and the target model, we only use one of them (not both) as the public cage to transfer the deformation, either of which will generate similar transfer results.

When the models S_0 and T_0 have great difference, we can align S_0 and T_0 together to compute the convex hull. Usually, this convex hull has too many slide triangles, and we still use [15] or [14] to simplify it. After the simplification, the simplified convex hull also needs to be enlarged in order to make sure it can enclose the original models S_0 and T_0, and the final convex hull will be used as the custom-made cage. Figure 7 shows this process. Above all, although it is difficult to design a special polyhedron for heterogeneous models, we found our custom-made cage is enough for most cases. **Transfer principle.** The principle of cage-based deformation transfer is: cage-based deformation can achieve continuous smooth transition. For two source models S_0 and S_1 with different state, it can be seen as a cage-based deformation from one state to another state. Using the computed cage state transition information, we can deform the target model based on the cage, and achieve a meaningful deformed model.

We set the public initial cage at the same position both for source models and target models. Using this configuration, the coefficients of the cage (cc_s and cc_t) for both models can be obtained based on the cage-based deformation method (MVC, HC, GC etc.) The vertices positions of the model can be uniformly expressed as the following equation in all the cage-based methods:

$$p = \Sigma_{i \in C_v} \phi_i(p) v_{c_i} + \Sigma_{j \in C_f} \psi_j(p) n(f_{c_j}) \tag{1}$$

Here, p is an arbitrary vertex position of the model, and $\phi_i(p)$ and $\psi_j(p)$ are the cage coefficients cc, and v_{c_i} denotes the i-th vertex position of the cage.

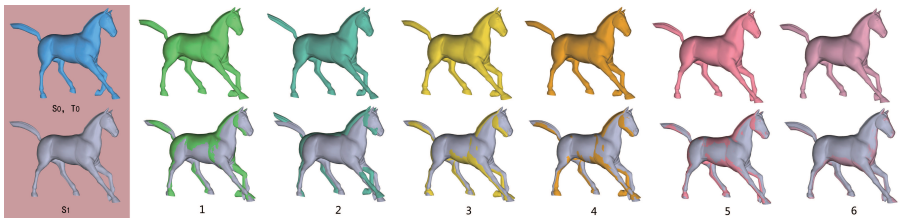

Fig. 8 Deformation transfer comparison. We use the left two horse models as the source models(S_0, S_1) and the top-left one as the target model T_0, then the transferred result T_1 should be a model close to the bottom-left model. Different T_1 models are shown in the top-right line, in which 1, 3 and 5 use mean value coordinates, while 2,4 and 6 use green coordinates. 1 and 2 use a cube as the public cage, 3 and 4 use a icosahedron, while 5 and 6 use a custom-made cage. We blend the results with the ideal result together in the bottom row for better observation.

f_{c_j} denotes the j-th face of the cage. $n(f_{c_j})$ denotes the normal of the j-th face of the cage. For MVC and HC, all $\psi_j(p)$ are 0. After deformation, the equation becomes as follows:

$$p' = \Sigma_{i \in C_v} \phi_i(p) k_i v'_{c_i} + \Sigma_{j \in C_f} \psi_j(p) s_j n'(f'_{c_j}) \tag{2}$$

$'$ denotes the new value after the deformation. We can see that $\phi_i(p)$ and $\psi_j(p)$ will keep constant during the deformation,and seen as cc in Alg. 1. $\{k_i\}$ and $\{s_j\}$ are scalars only related to the cage vertices' positions and cage faces' normals respectively. Corresponding computation methods for $\phi_i(p)$ and $\psi_j(p)$ can be referred [19][18][25], etc. The Equ. 2 can be written into a matrix form:

$$P = AV_C + BN_{F_c} \tag{3}$$

The sizes of P, A and V_C are $N \times 3$, $N \times N_C$ and $N_C \times 3$ respectively. N is the vertex number of the model, and N_C is the vertex number of the cage. The size of B and N_{F_c} are $N \times F_c$ and $F_c \times 3$ respectively, in which F_c is the face number of the cage. For MVC and HC, B is a zero matrix and the cage-based deformation transfer can be summarized as follows:

$$P_{t_1} = A_t V_{C_1} = A_t (A_s^T A_s)^{-1} A_s^T P_{s_1} \tag{4}$$

where A_s and A_t can be obtained from $P_{s_0} = A_s V_{C_0}$, and $P_{t_0} = A_t V_{C_0}$. For GC, the N_{F_c} can be computed using V_C, and is related to the topology of the cage. As $\Sigma\phi_i(p)$ equals 1, and the transfer is approximative, we can still use $(A_s^T A_s)^{-1} A_s^T P_{s_1}$ to compute V_{C_1}, and use Equ. 3 to generate the transferred target deformation model. Please note that A_s and A_t have different size because source models and target models usually are heterogeneous. However, they use the public cage to control the deformation and the matrix multiplication in the Equ.4 is reasonable.

As the public cage C_p can affect the deformation result T_1 of the target model T_0, the choice for the shape of the cage should consider the shape of the source model S_0 and the target model T_0. We can use a general cage such

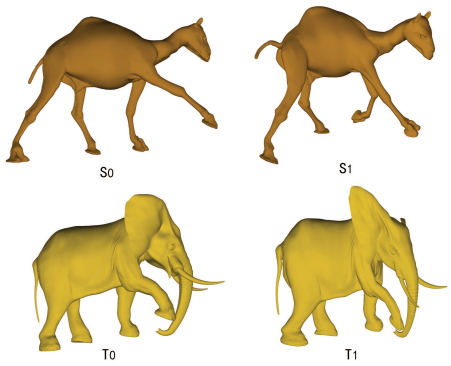

Fig. 9 Using source models to transfer the target elephant model in order to get the transferred result model T_1.

as a bounding box, or icosahedron or other polyhedrons (see Figure. 4) as the public cage for deformation transfer, although using a custom-made cage (see cage selection in 3.2) generated by our method can achieve a better result.

Figure 8 shows the distortion results using different cage coordinates and based on different cages. We use MVC and GC with cube, icosahedron and custom-made cages to transfer the deformation based on the left two horse models S_0 and S_1. The target model T_0 equals the top-left horse S_0, and the ideal transfer result should exactly equal the bottom-left horse S_1. However, the cage-based deformation is linear, while the deformation from S_0 to S_1 is not. As we use a public cage as the intermediate medium, there will be some information lost during our process and the difference can be detected in Figure 8.

Figure 9 shows a transfer result using the icosahedron as the public cage. We can see that the final result exhibits similar pose as "Source Model 2" (i.e.S_1 in Alg. 1) in Figure 9.

3.3 Optimization

After the cage-based deformation transfer, the generated deformed target model T_1 may have some rough appearance as the local details are destroyed, because the deformation does not consider the target model itself during the transfer process. In order to amend this, we design the following equation to optimize the transferred target model to get the model T_{op}:

$$\widetilde{x} = argmin_x(\sum_{x\in V}^{V}(\|Lx\| - \|\delta^{(x_0)}\|)^2 + \sum_{x\in V}^{V}\omega^2\|x - x_1\|^2) \tag{5}$$

Here, x_0 represents the vertex position of the target model T_0, x_1 denotes the vertex position of the transferred target model T_1, and $\widetilde{x}$ denotes the vertex

Table 1 Deformation transfer time in Figure 8

Number	Cage Type	Cage Vertex Number	Cage Coordinates Type	Computation Time(s)
1	cube	8	MVC	0.234
2	cube	8	GC	1.172
3	icosahedron	12	MVC	0.375
4	icosahedron	12	GC	1.922
5	custom-made	40	MVC	1.360
6	custom-made	40	GC	7.927

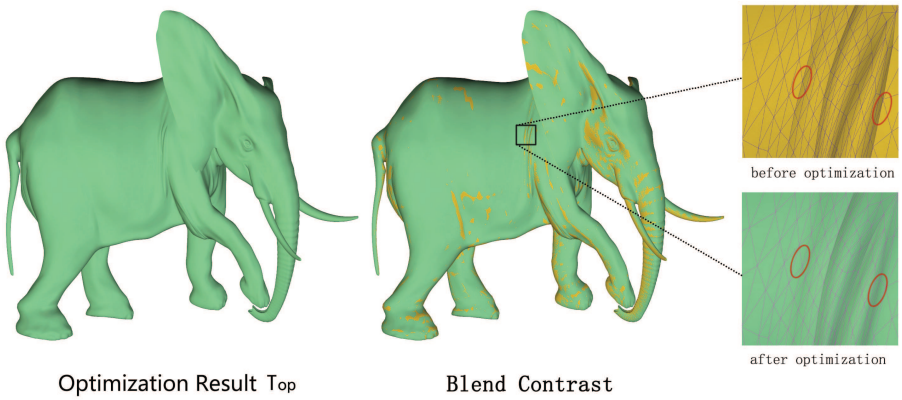

Fig. 10 Optimization Contrast. The left model is the optimization result of the "transfer result model" in Fig. 9. The right model blends these two models together, in which we can see that the optimized model is smoother.

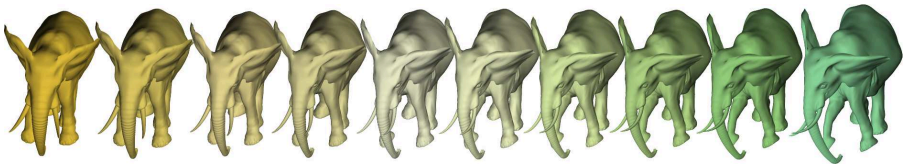

Fig. 11 Model interpolation. 8 interpolation models between the target elephant model and the optimized result model are computed based on shape space.

position of the optimized model T_{op}. $\delta^{(x_0)}$ denotes the differential coordinates of x_0, and $\|\delta^{(x_0)}\|$ is the absolute value of the mean curvature of x_0. The equation meaning is to search a new vertex position, which is close enough to the corresponding vertex position of the transferred target model, and keep the local mean curvature of the pre-deformed target model as possible.

Figure 10 shows the optimization result based on our equation. We can see that the optimized model keeps the local details better.

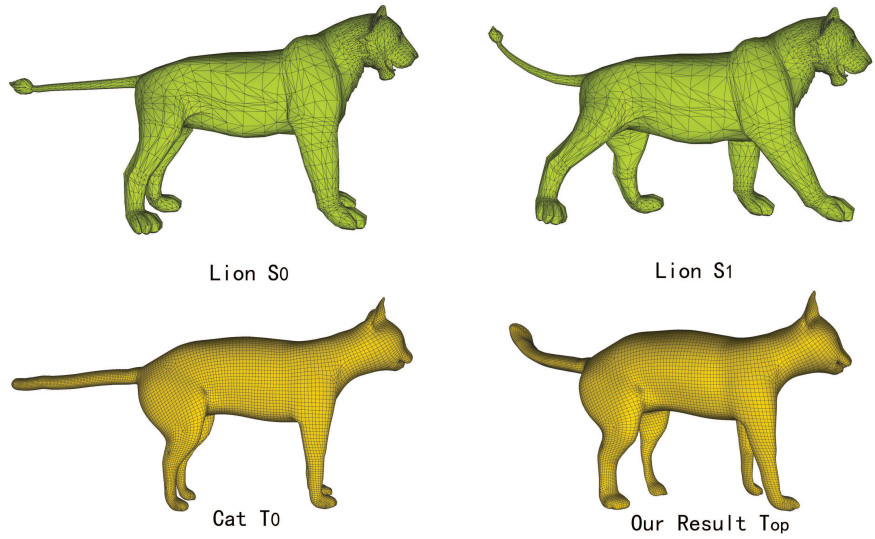

Fig. 12 Quadrilateral mesh deformation transfer. The source model lion 1 and lion 2 are triangle meshes (5000 vertices each), while the target model cat 1 is a quadrilateral mesh (15002 vertices). Using our method, we can naturally generate the shape exploration result cat 2. Mesh lines are rendered for better observation.

3.4 Interpolation

After optimizing the transferred target model to generate T_{op}, we can compute the interpolation models between T_0 and T_{op} in order to generate more shape exploration results for animation. Linear interpolation is available if the target model T_0 and the optimized model T_{op} have only small deformation. However, the optimized model T_{op} may have great changes, making the linear method inefficient for computing more exact models. We use the shape space method based on the Equ. 6 using the Riemannian metric as the object optimized function, which is proposed in [22] to get the interpolated models, and concatenate them to form the animation. Here, P_i represents the i-th interpolated model with $i = 1, 2, \cdots, n$, and P_0, P_{n+1} are the target model T_0 and the optimized model T_{op} respectively as the input. $\langle\langle X, Y\rangle\rangle$ can be rigid deformation metric or isometric deformation metric, and can blend L^2 shape metric. Readers can refer to [22] for details. Figure 11 shows the interpolation result for the elephant models.

$$E(P) := \sum_{i=0}^{n} (\langle\langle X_i, X_i\rangle\rangle_{P_i} + \langle\langle X_i, X_i\rangle\rangle_{P_{i+1}}) \tag{6}$$

Table 2 Time statistics for the optimization and the interpolation

Model Name	Vertex Number	Optimization Time(s)	Interpolation Number/Time(s)
Flamingo	26394	54.375	3/14.454
Horse	8431	8.078	3/ 3.828
Elephant	42321	127.485	8/40.906

4 Results and Discussion

Our experiments are implemented on the 32-bit Windows 7 operating system of a PC with an Intel Dual Core 2.40 GHz CPU, 4GB memory and an Nvidia GeForce 9800 GT display card.

Table 1 shows the deformation transfer time of Figure 8. The main time-consuming step is the computation of the cage coordinates cc_s and cc_t, and the computation of Equ. 4 is very fast. Although all the matrices in Equ. 4 are dense, the row (or column) size of $(A_s^T A_s)$ is very small, of the order of a few hundred, and equals the vertex number of the cage. Using LAPACK (Linear Algebra PACKage) library to solve the equation, the computation time is no more than 0.01 seconds.

The target model T_0 and the source model S_0 can have different vertices, topology or mesh structure, i.e, the source mesh S_0 can be a triangle mesh while the target model T_0 can be a quadrilateral mesh (see Figure 12), and vice versa. This case cannot be dealt with by previous deformation transfer methods. Figure 13 shows more results of heterogeneous models shape exploration.

We also test some cases that the source model and target model have great discrepancies. Note that for the following cases, previous state-of-art methods cannot work as the mapping between the source model and the target model is undefined. Using our method, we can give a plausible shape exploration result in the visual sense. Figure 14 shows how the lion deformation leads to an ant model's deform. Here the lion and the ant obviously have different number limbs. We can see that the shape exploration of Ant T_{op} has the similar action with Lion S_1. Figure 15 shows how horse affect a plant even there maybe no mapping between them, and Figure 16 shows the shape exploration of hand although the hand mesh is not closed while horse source models are. We also give the comparison result in our accompany video. For multi-resolution models, we can use the highest target resolution model and the source model S_0 to generate the public cage, and then other target resolution models can use this pubic cage for the shape exploration computing. Figure 17 shows the result, and we also shows the animation in the accompany video.

Table 2 shows the optimization time and interpolation numbers/time for the models in Figure 1 and Figure 11. Both times are proportional to the model vertex number, and the interpolation time additionally depends on the interpolation number.

On the other hand, as the deformation for the source models usually is nonlinear, while we are trying to use our cage-based method to express this

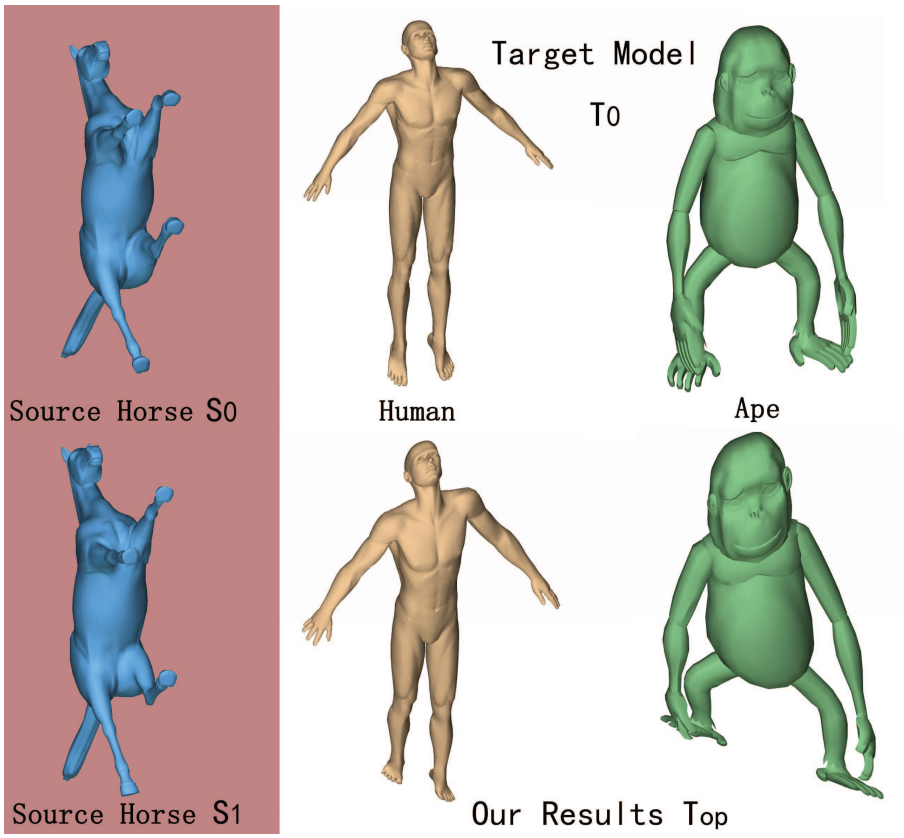

Fig. 13 Using two source models to generate shape exploration results on two target models respectively.

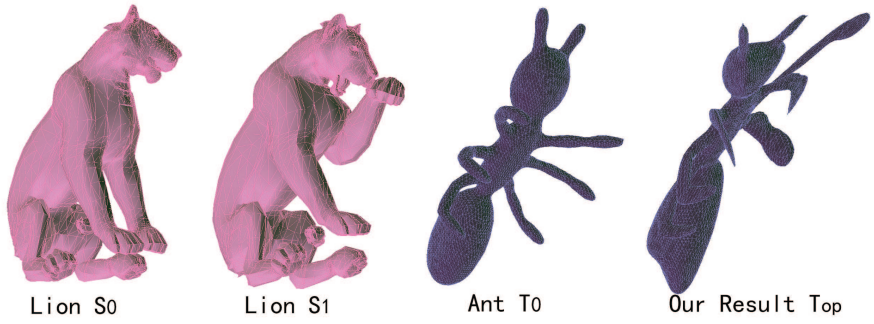

Fig. 14 The shape exploration result of AntT_{op} based on Lion S_0, S_1 and Ant T_0. We use the align state and the cage in Figure 7 to finish the process.

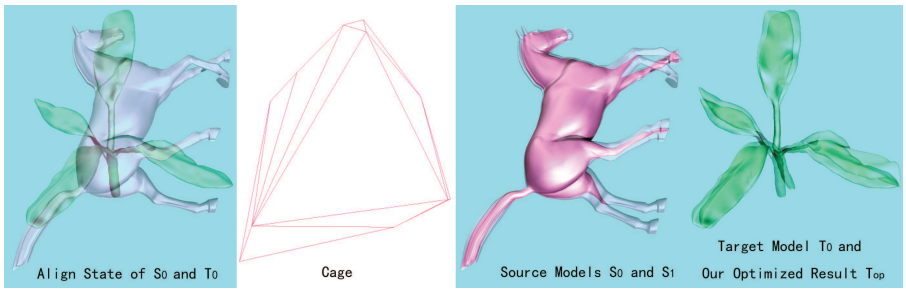

Fig. 15 The shape exploration result of plant T_{op} based on Horse S_0, S_1 and Plant T_0. Here we use translucent effect to render the models for better observation.The vertices number of the public cage is 12, and the face number is 20. Although there is no obvious mapping between Horse and Plant, similar model transition can be finished automatically. This cannot be done by other previous transfer methods.

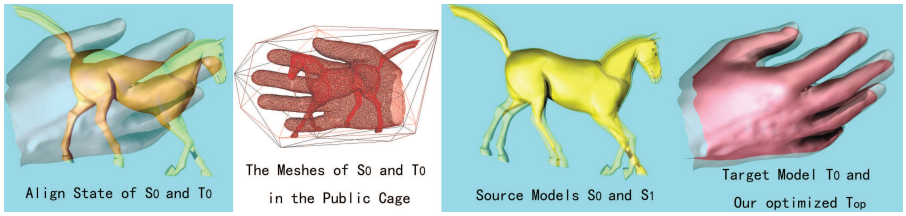

Fig. 16 The shape exploration result of plant T_{op} based on Horse S_0, S_1 and Hand T_0. The hand mesh is open, while the horse mesh is close. The vertices number of the public cage is 19 and the face number is 34.

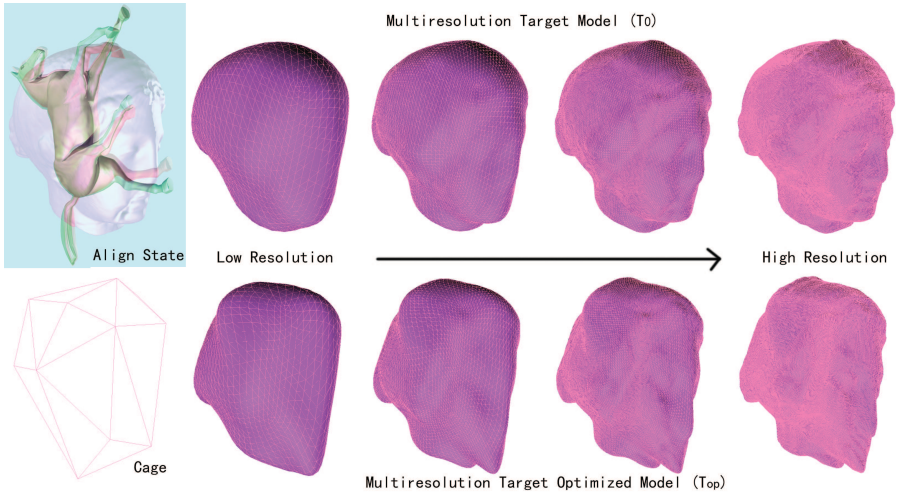

Fig. 17 The shape exploration results for multiresolution models. We use the highest resolution model and the source model to construct the public cage with 15 vertices and 26faces. The shape exploration for other multiresolution models can be smoothly generated.

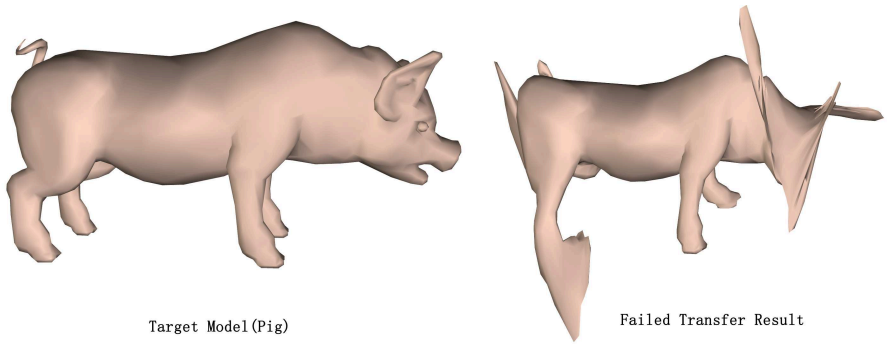

Fig. 18 Failed mesh transfer result. We use the lions in Figure 12 as the source models and the pig as the target model (shown in the left). Based on the sphere cage in Figure 4 and Green Coordinates, the transferred result is shown in the right. The main reason is that the deformed cage are intersected itself, and not a manifold anymore, which leads to an unreasonable deformation.

relation, which means using fewer vertices to simulate the process. In principle, there maybe no exact analytic expression existing at all. As a result, the vertex number of the cage used for deformation should be small. Too many vertices will increase the computation time, and may not bring better result, as the deformation only considered the change of the cage vertices' positions but not the intersections of the cage faces that may occur. This means the result may lead to cage intersection with itself, which in turn generates unreasonable deformed target models that have mesh inversion(See Figure 18).

5 Conclusions and Future work

We present a shape exploration framework for 3D heterogeneous models based on cages. Given two source models and another target model, we use a public cage as a transfer medium to implement shape transfer on the target model, then optimize the result based on the differential coordinates, and interpolate the models using shape space which in turn generates more shape exploration results for animation. The cage can be a general cage, or a custom-made cage for the model which can be generated automatically using our proposed custom-made cage generation method. Existing cage coordinates are employed to finish the transfer task, and optimization is executed subsequently to obtain a local-detail-preserving deformed model. More shape exploration results can be acquired by interpolation based on shape space. Differing from the previous method, our scheme can deal with quadrilateral target meshes (Figure 12) as well as triangle meshes. The whole process is feasible and efficient according to our experiments.

More improvements should be considered in the future. For the cage-based transfer, we should search a new solution to avoid the cage intersect with itself, in order to generate reasonable transferred results and eliminate the

failure cases such as in Figure 18. Besides, more methods like [41] and other constraints should be considered in order to transfer the details more exactly. For instance, the distance of front leg movement of the cat in Figure 12 is obviously less than that of the source model lion 2. Maybe small cages of local parts can be used to help transfer the local details, and the algorithm should be designed to avoid affecting other transferred states. For the model optimization, currently we just use the differential coordinates as the reference, and a more effective condition should be considered as the constraint condition in order to generate refined results. In addition, more user defined conditions should be included, aiming to make our method more practical and robust for production.

Acknowledgements This work is supported in part by the National High-Tech Research and Development Program of China (863 Program) with No.2015AA016402, and in part by National Natural Science Foundation of China with Nos. 61571439, 61561003, 61471261,61372190, and 61202324.

References

1. Baran, I., Vlasic, D., Grinspun, E., Popović, J.: Semantic deformation transfer. ACM Trans. Graph. **28**(3), 36:1–36:6 (2009)
2. Beier, T., Neely, S.: Feature-based image metamorphosis. In: SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques, pp. 35–42. ACM, New York, NY, USA (1992)
3. Ben-Chen, M., Weber, O., Gotsman, C.: Spatial deformation transfer. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 67–74. ACM (2009)
4. Ben-Chen, M., Weber, O., Gotsman, C.: Variational harmonic maps for space deformation. ACM Trans. Graph. **28**(3), 1–11 (2009)
5. Bookstein, F.L.: Principal warps: Thin-plate splines and the decomposition of deformations. IEEE Trans. Pattern Anal. Mach. Intell. **11**(6), 567–585 (1989)
6. Carmo, M.P.D.: Differential Geometry of Curves and Surfaces, chap. 1, pp. 16–23. Prentice-Hall (1976)
7. Chen, L., Huang, J., Sun, H., Bao, H.: Cage-based deformation transfer. Computers & Graphics **34**(2), 107–118 (2010)
8. Cohen-Or, D., Sorkine, O.: Encoding meshes in differential coordinates. In: (SCCG06)Proceedings of the 22nd Spring Conference on Computer Graphics, ACM Press. (2006)
9. Derose, T., Meyer, M.: Harmonic coordinates. Tech. rep., Pixar Animation Studios (2006)
10. Floater, M.S.: Mean value coordinates. Computer Aided Geometric Design **20**(1), 19–27 (2003)
11. Floater, M.S., Kós, G., Reimers, M.: Mean value coordinates in 3d. Comput. Aided Geom. Des. **22**(7), 623–631 (2005)
12. Fröhlich, S., Botsch, M.: Example-driven deformations based on discrete shells. In: Computer Graphics Forum, vol. 30, pp. 2246–2257. Wiley Online Library (2011)
13. García, F.G., Paradinas, T., Coll, N., Patow, G.: *cages:: A multilevel, multi-cage-based system for mesh deformation. pp. 24:1–24:13. ACM, New York, NY, USA (2013)
14. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. Computer Graphics **31**(Annual Conference Series), 209–216 (1997)
15. Hoppe, H.: Progressive meshes. In: SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 99–108. ACM, New York, NY, USA (1996)

16. Hormann, K., Floater, M.S.: Mean value coordinates for arbitrary planar polygons. ACM Trans. Graph. **25**(4), 1424–1441 (2006)
17. Jacobson, A., Baran, I., Popovic, J., Sorkine, O.: Bounded biharmonic weights for real-time deformation. In: SIGGRAPH '11: ACM SIGGRAPH 2011 Papers (2011)
18. Joshi, P., Meyer, M., DeRose, T., Green, B., Sanocki, T.: Harmonic coordinates for character articulation. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, p. 71. ACM, New York, NY, USA (2007)
19. Ju, T., Schaefer, S., Warren, J.: Mean value coordinates for closed triangular meshes. ACM Trans. Graph. **24**(3), 561–566 (2005)
20. 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), 1 (2008)
21. Karni, Z., Gotsman, C.: Spectral compression of mesh geometry. In: SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 279–286. ACM Press, New York, NY, USA (2000)
22. Kilian, M., Mitra, N.J., Pottmann, H.: Geometric modeling in shape space. ACM Trans. Graph. **26**(3), 64 (2007)
23. Li, X.Y., Ju, T., Hu, S.M.: Cubic mean value coordinates. ACM Transactions on Graphics **32**(4), 98:1–10 (2013)
24. Lipman, Y., Kopf, J., Cohen-Or, D., Levin, D.: Gpu-assisted positive mean value coordinates for mesh deformations. In: SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing, pp. 117–123. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2007)
25. Lipman, Y., Levin, D., Cohen-Or, D.: Green coordinates. In: SIGGRAPH '08: ACM SIGGRAPH 2008 papers, pp. 1–10. ACM, New York, NY, USA (2008)
26. Luebke, D., Harris, M., Krüger, J., Purcell, T., Govindaraju, N., Buck, I., Woolley, C., Lefohn, A.: Gpgpu: general purpose computation on graphics hardware. In: SIGGRAPH '04: ACM SIGGRAPH 2004 Course Notes, p. 33. ACM, New York, NY, USA (2004)
27. Ma, C., Huang, H., Sheffer, A., Kalogerakis, E., Wang, R.: Analogy-driven 3D style transfer. Computer Graphics Forum **33**(2), 175–184 (2014)
28. MacCracken, R., Joy, K.I.: Free-form deformations with lattices of arbitrary topology. In: SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 181–188. ACM, New York, NY, USA (1996)
29. Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A., Guibas, L.: Functional maps: A flexible representation of maps between shapes. ACM Transactions on Graphics (TOG) **31**(4), 30 (2012)
30. Shlens, J.: A tutorial on principal component analysis. Systems Neurobiology Laboratory, University of California at San Diego (2005)
31. Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.P.: Laplacian surface editing. In: SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, pp. 175–184. ACM, New York, NY, USA (2004)
32. Sorkine, O., Cohen-Or, D., Toledo, S.: High-pass quantization for mesh encoding. In: SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing, pp. 42–51. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2003)
33. Sumner, R.W., Popović, J.: Deformation transfer for triangle meshes. In: SIGGRAPH '04: ACM SIGGRAPH 2004 Papers, pp. 399–405. ACM, New York, NY, USA (2004)
34. Thiery, J.M., Tierny, J., Boubekeur, T.: Cager: Cage-based reverse engineering of animated 3d shapes. Computer Graphics Forum **31**(8), 2303–2316 (2012)
35. Turk, G., O'Brien, J.F.: Shape transformation using variational implicit functions. In: ACM SIGGRAPH 2005 Courses, p. 13. ACM (2005)
36. Wang, Y., Jacobson, A., Barbič, J., Kavan, L.: Linear subspace design for real-time shape deformation. ACM Trans. Graph. **34**(4), 57:1–57:11 (2015)
37. Weber, O., Ben-Chen, M., Gotsman, C.: Complex barycentric coordinates with applications to planar shape deformation. Computer Graphics Forum (Proceedings of Eurographics) **28**(2) (2009)
38. Weber, O., Sorkine, O., Lipman, Y., Gotsman, C.: Context-aware skeletal shape deformation. Computer Graphics Forum (Proceedings of Eurographics) **26**(3) (2007)

39. Winkler, T., Drieseberg, J., Alexa, M., Hormann, K.: Multi-scale geometry interpolation. In: Computer graphics forum, vol. 29, pp. 309–318. Wiley Online Library (2010)
40. Yoshiyasu, Y., Yamazaki, N.: Detail-aware spatial deformation transfer. Computer Animation and Virtual Worlds **23**(3-4), 225–233 (2012)
41. Zhang, J., Deng, B., Liu, Z., Patanè, G., Bouaziz, S., Hormann, K., Liu, L.: Local barycentric coordinates. ACM Trans. Graph. **33**(6), 188:1–188:12 (2014)
42. Zhao, Y., Pan, B., Xiao, C., Peng, Q.: Dual-domain deformation transfer for triangular meshes. Computer Animation and Virtual Worlds **23**(3-4), 447–456 (2012)
43. Zhou, K., Xu, W., Tong, Y., Desbrun, M.: Deformation transfer to multi-component objects. In: Computer Graphics Forum, vol. 29, pp. 319–325 (2010)