

International Journal of Shape Modeling
© World Scientific Publishing Company

Morphing of Triangular Meshes in Shape Space*

Stefanie Wuhrer

*National Research Council of Canada, Ottawa, Ontario, Canada
stefanie.wuhrer@nrc-cnrc.gc.ca*

Prosenjit Bose

*School of Computer Science, Carleton University, Ottawa, Ontario, Canada
jit@scs.carleton.ca*

Chang Shu

*National Research Council of Canada, Ottawa, Ontario, Canada
chang.shu@nrc-cnrc.gc.ca*

Joseph O'Rourke

*Smith College, Northampton, USA
orourke@cs.smith.edu*

Alan Brunton

*University of Ottawa, Ottawa, Canada
National Research Council of Canada, Ottawa, Canada
abrunton@site.uottawa.ca*

Received (Day Month Year)

Revised (Day Month Year)

Accepted (Day Month Year)

Communicated by (xxxxxxxxxx)

We present a novel approach to morph between two isometric poses of the same non-rigid object given as triangular meshes. We model the morphs as linear interpolations in a suitable shape space \mathcal{S} . For triangulated 3D polygons, we prove that interpolating linearly in this shape space corresponds to the most isometric morph in \mathbb{R}^3 . We then extend this shape space to arbitrary triangulations in 3D using a heuristic approach and show the practical use of the approach using experiments. Furthermore, we discuss a modified shape space that is useful for isometric skeleton morphing. The novelty of the presented approaches is a solution to the morphing problem that does not need to solve a minimization problem.

Keywords: Morphing, shape space, geometry processing, computational geometry.

*A preliminary version of this work was accepted to CCCG 2008. Research supported in part by HPCVL and NSERC.

2 *S. Wuhler, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

1. Introduction

Interpolating smoothly between two given poses of the same non-rigid object, known as morphing, arises in many geometry processing problems. For example, morphing can compute an animation between given poses of a human or an animal. Two objects are *isometric* if they have the same intrinsic geometry. We consider morphs between isometric objects because the motion of humans and many animals preserves the intrinsic geometry of their bodies. We aim to solve the following problem. Given two isometric poses of the same non-rigid object modeled as triangular meshes $S^{(0)}$ and $S^{(1)}$ with known point-to-point correspondences, find a smooth isometric deformation from one pose to the other.

Recently, Kilian et al.⁹ showed that an isometric morph can be computed by finding a geodesic path in a shape space. A mesh that contains n vertices is usually represented in shape space as a point in \mathbb{R}^{3n} that contains all the vertex coordinates of the mesh.⁸ The mesh is arranged in a standard position so that it is translation and rotation invariant. Computing geodesic paths in this space involves solving a large-scale non-linear optimization problem. Non-linear optimization problems are usually solved iteratively starting from an initial solution. This is computationally expensive. It is difficult to find the best solution because convergence problems may arise if the initial solution is far from the optimum, and finding a good initial solution is not straight forward. Kilian et al. overcome the inefficiency of the approach and the difficulty of finding a good initial solution by implementing two multi-resolution schemes. The number of resolutions that are used have an effect on the result. These resolutions can be viewed as input parameters that are non-intuitive. The approach is difficult to implement.

The objective of this paper is to propose an alternative representation of the shape space that does not require solving large-scale non-linear optimization problems nor the use of non-intuitive input parameters. Instead of encoding the extrinsic geometry of the mesh in shape space as in Kilian et al., we encode the intrinsic geometry of the mesh in shape space. This has the effect that a point on a geodesic path in shape space linearly interpolates between the endpoints of the geodesic path. This is how we replace solving a large-scale non-linear optimization problem with linearly interpolating between two points. This is more efficient, conceptually simpler, and easier to implement.

A deformation of a shape represented by a triangular mesh is isometric if and only if all triangle edge lengths are preserved during the deformation.⁹ We call a morph $S^{(t)}$, $0 < t < 1$ between two (possibly non-isometric) shapes $S^{(0)}$ and $S^{(1)}$ *most isometric* if it minimizes the sum of the absolute values of the differences between the corresponding edge lengths of two consecutive shapes summed over all shapes $S^{(t)}$, for t in $[0, 1]$. Note that in a similar definition, Kilian et al.⁹ use the L_2 instead of the L_1 metric as we do to measure most isometric morphs. In this paper, we examine isometric morphs of general *triangular manifold meshes* in 3D. We start by examining the simpler problem of morphing between isometric

triangulated 3D polygons, which are triangular meshes with no interior vertices. In this case, we introduce a new shape space \mathcal{S} that has the property that interpolating linearly in shape space corresponds to the most isometric morph in \mathbb{R}^3 . For arbitrary triangular meshes, the existence of interior vertices means that this property is no longer guaranteed. However, we extend the shape space to arbitrary triangular manifold meshes using a heuristic approach. While we can compute morphs in this shape space using linear interpolations, the morphs are no longer guaranteed to be the most isometric morph. Furthermore, we discuss a modification of the shape space that is useful for the morphing of isometric skeletons.

2. Related Work

Computing a smooth morph from one pose of a shape in two or three dimensions to another pose of the same shape has received considerable attention.^{11, 3} A survey on this topic is by Alexa.²

Early work on morphing focused on the two-dimensional version of the problem. We focus on the case where the input is sampled over an irregular domain. The problem is to interpolate between two simple polygons in the plane. Several authors make use of intrinsic representations of the polygons. Sederberg et al.¹⁵ proposed to interpolate an intrinsic representation of two-dimensional polygons, namely the edge lengths and interior angles of the polygon. Surazhsky and Gotsman²¹ morphed by computing mean value barycentric coordinates based on an intrinsic representation of triangulated polygons. This method is guaranteed to be intersection free. Iben et al.⁷ morphed planar polygons while guaranteeing that no self-intersections occur using an approach based on energy minimization. This approach can be constrained to be as isometric as possible. Our approach can be viewed as a generalization of these techniques to three dimensions.

More recently, much work has focused on morphing between triangular meshes. We first review work that does not require energy minimizations. Sun et al.²⁰ morphed between three-dimensional manifold meshes. They extended the approach by Sederberg et al.¹⁵ to three dimensions by extending the intrinsic representation to polyhedra. However, the methods developed are computationally expensive.² Another extension of the approach by Sederberg et al.¹⁵ to three dimensions was presented by Lipman et al.¹² They gave a rotation-invariant mesh representation that is similar in spirit to the first and second fundamental form in differential geometry. This representation can be used to find an approximate morph. The method is again computationally expensive. Our approach is similar to these approaches in that no energy is minimized. However, it is more efficient in terms of the computational complexity.

Alexa¹ proposed an approach that can be used to compute local morphs between two meshes of the same topology. This approach makes use of Laplacian representation. While the approach is shown to perform well for deforming and morphing local areas, the approach is not tailored to morph between two whole

4 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

shapes representing two poses of the same non-rigid object. In fact, most of the examples show local morphs between meshes of drastically different shape.

Next we review approaches that require solving non-linear optimization problems. When solving non-linear optimization problems, it is often not guaranteed that a globally optimal solution is found. Kraevoy and Sheffer¹⁰ presented a representation that is based on mean-value encodings of vertices. This representation has the property that similar models have similar encodings and it can be used for morphing. Morphing requires minimizing a non-linear energy function. Sorkine and Alexa¹⁸ proposed an algorithm to deform a surface based on a given triangular surface and updated positions of few feature points. The surface is modeled as a covering by overlapping cells. The deformation aims to deform each cell as rigidly as possible. The overlap is necessary to avoid stretching along cell boundaries. The deformation is based on minimizing a global non-linear energy function. While the energy is guaranteed to converge, the algorithm is not guaranteed to find the global minimum. If the global minimum is found, the approach tends to preserve the edge lengths of the triangular mesh. Zhou et al.²⁴ proposed a method to deform triangular meshes based on the Laplacian of a graph representing the volume of the triangular mesh. The method is shown to prevent volumetric details to change unnaturally. The work by Kilian et al.⁹ was described in Section 1. In their approach, the geodesic paths in shape space are found using an energy-minimization approach. Note that Kilian et al.'s approach is more general than our approach in that it is possible to extrapolate isometric deformations. Eckstein et al.⁵ proposed a generalized gradient descent method similar to the approach by Kilian et al. that can be applied to deform triangular meshes.

In this paper, we propose a novel shape space representation for morphing that does not require solving a minimization problem. The representation has the property that interpolating linearly in shape space approximates the most isometric morph in \mathbb{R}^3 . For triangulated 3D polygons, we prove that the linear interpolation in shape space corresponds exactly to the most isometric morph in \mathbb{R}^3 . For arbitrary triangulated manifolds in 3D, we provide a heuristic that extends the approach developed for 3D polygons. Experimental results compare our algorithm to the approach by Kilian et al.

The problem of morphing between shapes is closely related to the problem of computing point-to-point correspondence between two shapes, since many morphing algorithms, as our does, assume that the point-to-point correspondence is known. Computing the point-to-point correspondence is an important problem and constitutes an active research area. For a recent survey on this topic, see van Kaick et al.²³

3. Theory of Shape Space for Triangulated 3D Polygons

Before considering the problem of morphing between triangulated manifolds in 3D, we consider the simpler problem of morphing between triangular meshes with no

interior vertices. We call these meshes *triangulated 3D polygons*.

We are given two triangulated 3D polygons $P^{(0)}$ and $P^{(1)}$ corresponding to two *near-isometric* poses of the same non-rigid object. Poses are considered near-isometric if the stretching between the poses is small. We assume that the point-to-point correspondence of the vertices of $P^{(0)}$ and $P^{(1)}$ is known. Furthermore, we assume that both $P^{(0)}$ and $P^{(1)}$ share the same underlying mesh structure M . Hence, we know the mesh structure M with two sets of ordered vertex coordinates $V^{(0)}$ and $V^{(1)}$ in \mathbb{R}^3 , where M is an outer-planar graph. We will show that we can represent $P^{(0)}$ and $P^{(1)}$ as points $p^{(0)}$ and $p^{(1)}$ in a shape space \mathcal{S} , such that each point $p^{(t)}$ that is a linear interpolation between $p^{(0)}$ and $p^{(1)}$ corresponds to a triangular mesh $P^{(t)}$ isometric to $P^{(0)}$ and $P^{(1)}$ in \mathbb{R}^3 .

Note that the assumption that the point-to-point correspondence of the vertices is known is a strong assumption in general since the problem of computing the point-to-point correspondence is difficult. However, it is possible to compute accurate correspondences based on a template shape of the class of shapes to be considered and a small set of manually placed marker positions. A variety of template-based techniques are reviewed in van Kaick et al.²³ Since we consider the problem of computing correspondences between two shapes of the same object in different poses, these approaches could be applied to obtain accurate correspondences. We therefore assume that the correspondences are given as part of the input to our method.

Since we know the point-to-point correspondence of the vertices of $P^{(0)}$ and $P^{(1)}$, we can find the best rigid alignment of the two shapes by solving an overdetermined linear system of equations and by modifying the solution to ensure a valid rotation matrix. To modify the solution to the overdetermined linear system, we compute a singular value decomposition of the solution matrix and replace the singular value matrix by the identity matrix.⁶ Note that the relative rigid alignment of $P^{(0)}$ and $P^{(1)}$ in \mathbb{R}^3 has an influence on the linear interpolation. The reason is that we aim to interpolate linearly between the edge lengths and the surface normals of the given poses.

Let M consist of n vertices. Since M is a triangulation of a 3D polygon with n vertices, M has $2n - 3$ edges and $n - 2$ triangles. We assign an arbitrary but fixed order on the vertices, edges, and faces of M . In the following, when referring to the ordering of vertices, edges, or faces of M , we refer to this fixed order. The shape space \mathcal{S} is defined as follows. The first 3 coordinates of a point $p \in \mathcal{S}$ correspond to the coordinates of the first vertex v in M . Coordinates 4 and 5 of p correspond to the direction of the first edge of M incident to v in spherical coordinates. The next $2n - 3$ coordinates of p are the lengths of the edges in M in order. The final $4n - 4$ coordinates of p describe the outer normal directions of the triangles in M in spherical coordinates, in order. Hence, the shape space \mathcal{S} has dimension $5 + 2n - 3 + 4n - 4 = 4n - 2 = \Theta(n)$.

In the following, we prove that interpolating linearly between $P^{(0)}$ and $P^{(1)}$ in shape space yields the most isometric morph. To interpolate linearly in shape

6 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

space, we interpolate the edge lengths by a simple linear interpolation. That is, $p_k^{(t)} = tp_k^{(0)} + (1-t)p_k^{(1)}$, where $p_k^{(x)}$ is the k th coordinate of $p^{(x)}$. The normal vectors are interpolated using geometric spherical linear interpolation (SLERP).¹⁶ That is, $p_k^{(t)} = \frac{\sin(1-t)\Theta}{\sin\Theta}p_k^{(0)} + \frac{\sin t\Theta}{\sin\Theta}p_k^{(1)}$, where Θ is the angle between the two directions that are interpolated.

To study interpolation in shape space, we make use of the *dual graph* $D(M)$ of M . The dual graph $D(M)$ has a node for each triangle of M . We denote the dual node corresponding to face f of M by $D(f)$. Two nodes of $D(M)$ are joined by an arc if the two corresponding triangles in M share an edge. We denote the dual arc corresponding to an edge e of M by $D(e)$. Note that because M meshes a 3D polygon, it is a triangulated outer-planar graph and so the dual graph of M is a binary tree. An example of a mesh M with its dual graph $D(M)$ is shown in Figure 1.

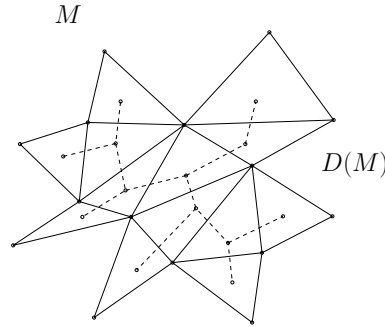


Fig. 1. A mesh M with its dual graph $D(M)$.

We call a triangular mesh *valid* if each triangle satisfies the triangle inequality.

Theorem 3.1. *Let M be the underlying mesh structure of the triangulated 3D polygons $P^{(0)}$ and $P^{(1)}$. The linear interpolation $p^{(t)}$ between $p^{(0)}$ and $p^{(1)}$ in shape space \mathcal{S} for $0 \leq t \leq 1$ has the following properties:*

- (1) *There exists a unique mesh $P^{(t)} \in \mathbb{R}^3$ that corresponds to $p^{(t)} \in \mathcal{S}$ that has the underlying mesh structure M . The mesh $P^{(t)}$ is a valid triangular mesh. We can compute this mesh using a traversal of the binary tree $D(M)$ in $\Theta(n)$ time.*
- (2) *If $P^{(0)}$ and $P^{(1)}$ are isometric, then $P^{(t)}$ is isometric to $P^{(0)}$ and $P^{(1)}$. If $P^{(0)}$ and $P^{(1)}$ are not isometric, then each edge length of $P^{(t)}$ linearly interpolates between the corresponding edge lengths of $P^{(0)}$ and $P^{(1)}$.*
- (3) *The coordinates of the vertices of $P^{(t)}$ are a continuous function of t .*

Proof. Part 1: To prove uniqueness, we note that the first vertex v of $P^{(t)}$ is uniquely determined by the first three coordinates of $p^{(t)}$. Let e denote the first

edge of M incident to v . The direction e is uniquely determined by coordinates 4 and 5 of $p^{(t)}$. The length of each edge of $P^{(t)}$ is uniquely determined by the following $2n-3$ coordinates. Hence, the edge e is uniquely determined. Furthermore, the outer normal of each triangle is uniquely determined by the following $4n-4$ coordinates. For a triangle f containing e , we now know the position of two vertices of f , the plane containing f , and the three lengths of the edges of f . Assuming that the normal vectors in shape space represent right-hand rule counterclockwise traversals of each triangle, this uniquely determines the position of the last vertex of f . We can now determine the coordinate of each vertex of $P^{(t)}$ uniquely by traversing $D(M)$.

Since $D(M)$ is a tree, it is cycle-free and the coordinates of each vertex of $P^{(t)}$ are set exactly once. Because the complexity of $D(M)$ is $\Theta(n)$, the algorithm terminates after $\Theta(n)$ steps.

The mesh $P^{(t)}$ is a valid triangular mesh because both input meshes are valid triangular meshes.

Part 2: The edge lengths of $P^{(t)}$ are linear interpolations between the edge lengths of $P^{(0)}$ and $P^{(1)}$. Hence, the claim follows.

Part 3: When varying t continuously, the point $p^{(t)} \in \mathcal{S}$ varies continuously. Hence, the coordinate of the lengths of all the edges vary continuously. Because a direction $[\sin v \cos u, \sin v \sin u, \cos v]^T$ varies continuously if u and v vary continuously, the normal directions vary continuously. Because all the vertex positions of the mesh $P^{(t)}$ are uniquely determined by continuous functions of those quantities, all vertex positions of $P^{(t)}$ vary continuously. \square

We do not need to solve minimization problems to find the shortest path in shape space as in Kilian et al. ⁹ The only computation required to find an intermediate deformation pose is a graph traversal of $D(M)$.

Because $D(M)$ has complexity $\Theta(n)$, we can traverse $D(M)$ in $\Theta(n)$ time. Hence, we can compute intermediate deformation poses in $\Theta(n)$ time each. We call this the *polygon algorithm*. In the following, we show that the polygon algorithm computes the most isometric morph.

Corollary 3.2. *The polygon algorithm computes the most isometric morph between two triangulated 3D polygons $P^{(0)}$ and $P^{(1)}$ with underlying mesh structure M .*

Proof. The following argument shows that Part 2 of Theorem 3.1 implies that the most isometric morph is found. Consider an edge e of M and let $l(e)^{(t)}$ denote the length of e in $P^{(t)}$. As all edge lengths are linearly interpolated during the morph, e is linearly interpolated during the morph. Let the morph consist of $k+1$ poses $P^{(0)}, P^{(\frac{1}{k})}, \dots, P^{(1)}$. Recall that a morph is most isometric if it minimizes

$$Q = \sum_{e \in E} \sum_{i=1}^k \left| l(e)^{\left(\frac{i}{k}\right)} - l(e)^{\left(\frac{i-1}{k}\right)} \right|,$$

8 *S. Wuhler, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

where E is the edge set of M . We have

$$\begin{aligned} Q &\geq \sum_{e \in E} \sum_{i=1}^k l(e)^{\left(\frac{i}{k}\right)} - l(e)^{\left(\frac{i-1}{k}\right)} \\ &\geq \sum_{e \in E} \left(l(e)^{\left(\frac{1}{k}\right)} - l(e)^{(0)} + l(e)^{\left(\frac{2}{k}\right)} - l(e)^{\left(\frac{1}{k}\right)} + \dots + l(e)^{(1)} - l(e)^{\left(\frac{k-1}{k}\right)} \right) \\ &\geq \sum_{e \in E} l(e)^{(1)} - l(e)^{(0)}. \end{aligned}$$

The linear interpolation achieves exactly $Q = \sum_{e \in E} l(e)^{(1)} - l(e)^{(0)}$, for it simply sums up the total difference $l(e)^{(1)} - l(e)^{(0)}$ in e 's lengths from start to finish in $k + 1$ little pieces. As linear interpolation achieves the lower bound on Q , the most isometric morph is found by the polygon algorithm. \square

Figure 2 shows an example of an isometric morph of a simple polygon. The start polygon is a 3D polygon obtained by discretizing the curve $y = \sin(x)$ and by adding thickness to the curve along the z -direction. The end polygon is obtained similarly from $y = -\sin(x)$.

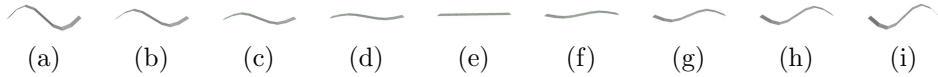


Fig. 2. *Isometric morph of a simple polygon from pose (a) to pose (i). Intermediate poses obtained using the polygon algorithm are given from left to right.*

4. Theory of Shape Space for Skeleton Morphing

A similar shape space to the one presented in Section 3 can be used to isometrically morph between two topologically equivalent skeletons. Let a *skeleton* in \mathbb{R}^3 be a set of joints connected by links arranged in a tree-structure. That is, we can consider a skeleton to be a tree ^a in \mathbb{R}^3 consisting of n vertices and $n - 1$ undirected edges.

The shape space presented in Section 3 can be simplified to a shape space for skeletons in \mathbb{R}^3 with the property that interpolating linearly in shape space corresponds to the most isometric morph in \mathbb{R}^3 . The dimensionality of the shape space is linear in the number of links of the skeleton.

We start with two skeletons $S^{(0)}$ and $S^{(1)}$ corresponding to two near-isometric poses. We assume that the point-to-point correspondence of $S^{(0)}$ and $S^{(1)}$ is known. Hence, we know the tree structure T with two sets of ordered vertex coordinates $V^{(0)}$ for $S^{(0)}$ and $V^{(1)}$ for $S^{(1)}$ in \mathbb{R}^3 . As before, we first find the best rigid alignment of the two skeletons.

The skeleton shape space \mathcal{S}_S is defined in a similar way as \mathcal{S} . We assign an arbitrary but fixed root to T and traverse the edges of T in a depth-first order. We assign an arbitrary order to the edges incident on each vertex of T . Note that

^aIn this context, a tree is an acyclic graph.

while the order is not unique, it is fixed. That is, the order we choose remains static throughout the algorithm. In the following, when referring to edges in depth-first order, we refer to this static order.

The first 3 coordinates of s correspond to the coordinates of the root in T . The next $n - 1$ coordinates of s are the lengths of the edges in T in depth-first order. The final $2(n - 1)$ coordinates of s describe the unit directions of the edges in spherical coordinates in depth-first order. All edges are oriented such that they point away from the root. Note that the shape space \mathcal{S} has dimension $\Theta(n)$.

Interpolating linearly between points in $\mathcal{S}_{\mathcal{S}}$ is performed the same way as interpolating linearly between points in \mathcal{S} . Namely, edge lengths are interpolated linearly and unit directions are interpolated via SLERP. With a technique similar to the proof of Theorem 3.1, we can prove the following theorem. In the proof, we do not need to consider a dual graph, but we can simply traverse the tree T in depth-first order to propagate the information.

Theorem 4.1. *Let T be the underlying tree structure of the skeletons $S^{(0)}$ and $S^{(1)}$. The linear interpolation $s^{(t)}$ between $s^{(0)}$ and $s^{(1)}$ in shape space $\mathcal{S}_{\mathcal{S}}$ for $0 \leq t \leq 1$ has the following properties:*

- (1) *The skeleton $S^{(t)} \in \mathbb{R}^3$ that corresponds to $s^{(t)} \in \mathcal{S}_{\mathcal{S}}$ is uniquely defined and has the underlying tree structure T . We can compute this tree using a depth-first traversal of the tree in $\Theta(n)$ time.*
- (2) *If $S^{(0)}$ and $S^{(1)}$ are isometric, then $S^{(t)}$ is isometric to $S^{(0)}$ and $S^{(1)}$. If $S^{(0)}$ and $S^{(1)}$ are not perfectly isometric, then each edge length of $S^{(t)}$ linearly interpolates between the corresponding edge lengths of $S^{(0)}$ and $S^{(1)}$.*
- (3) *The coordinates of the vertices of $S^{(t)}$ are a continuous function of t .*

This theorem allows us to morph isometrically between the skeletons of two shapes corresponding to two postures of the same articulated object in $\Theta(n)$ time. Figure 3 shows an example of such a morph for synthetic data.

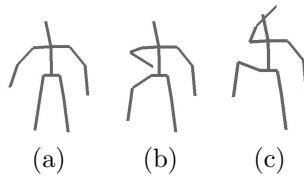


Fig. 3. *Example of most isometric morph between two skeleton poses. Input poses are shown in (a) and (c) and morph for $t = 0.5$ is shown in (b).*

In the remainder of this paper, we focus our attention on morphing between triangular meshes.

10 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

5. Generalization to Triangular Meshes

This section extends the shape space \mathcal{S} from Section 3 to arbitrary connected triangular manifold meshes.^b However, we can no longer guarantee the properties of Theorem 3.1, because the dual graph of the triangular mesh M is no longer a tree.

Given two triangular meshes $S^{(0)}$ and $S^{(1)}$ corresponding to two near-isometric poses of the same non-rigid object with known point-to-point correspondences, we have one mesh structure M with two sets of ordered vertex coordinates $V^{(0)}$ and $V^{(1)}$ in \mathbb{R}^3 .

As in Sections 3 and 4, we can use this information to find the best rigid alignment in \mathbb{R}^3 . We do this before representing the shapes in a shape space. As outlined above, this alignment has a major influence on the result of the morph.

We can represent $S^{(0)}$ and $S^{(1)}$ as points $s^{(0)}$ and $s^{(1)}$ in a shape space \mathcal{S} using the same shape space points as in Section 3. Let $s^{(t)}$ be the linear interpolation of $s^{(0)}$ and $s^{(1)}$ in \mathcal{S} , where the linear interpolation is computed as outlined in Section 3. The existence of a mesh $S^{(t)} \in \mathbb{R}^3$ that has the underlying mesh structure M and that corresponds to $s^{(t)}$ is no longer guaranteed. This can be seen using the example shown in Figure 4. Figure 4(a) and 4(b) show two isometric meshes $S^{(0)}$ and $S^{(1)}$. The dual graph $D(M)$ of the mesh structure M is a simple cycle. Note that although the start and the end poses are isometric, we cannot find an intermediate pose that satisfies all of the interpolated normal vectors with SLERP and that is isometric to $S^{(0)}$ and $S^{(1)}$. This yields the following observation.

Observation. Given a triangular mesh $S^{(t)}$ with underlying mesh structure M , point $s^{(t)}$ in \mathcal{S} is uniquely determined. However, the inverse operation, that is computing a triangular mesh $S^{(t)}$ given a point $s^{(t)} \in \mathcal{S}$, is ill-defined.

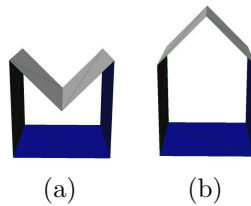


Fig. 4. *Example of isometric triangular meshes where intermediate poses interpolating all normals and edge lengths do not exist.*

To compute a unique triangular mesh $S^{(t)}$ given a point $s^{(t)} \in \mathcal{S}$ that linearly

^bA *triangular manifold mesh* represents a shape S by a triangular mesh that has the properties that each edge of S is adjacent to at most two triangles and that the triangles incident on an arbitrary vertex of S can be ordered locally around the vertex. A triangular manifold mesh is *connected* if its dual graph is connected.

interpolates between $s^{(0)}$ and $s^{(1)}$, such that $S^{(t)}$ approximates the information given in $s^{(t)}$ well, we use the dual graph $D(M)$ of M . Unlike in Section 3, $D(M)$ is not necessarily a tree. Our algorithm therefore operates on a minimum spanning tree $T(M)$ of $D(M)$. The tree $T(M)$ is computed by assigning a weight to each arc e of $D(M)$. The weight of e is equal to the difference in dihedral angle of the supporting planes of the two triangles of M corresponding to the two endpoints of e . That is, we compute the dihedral angle between the two supporting planes of the two triangles of M corresponding to the two endpoints of e for the start pose $S^{(0)}$ and for the end pose $S^{(1)}$, respectively. The weight of e is then set as the difference between those two dihedral angles, which corresponds to the change in dihedral angle during the deformation. The weight can therefore be seen as a measure of rigidity. The smaller the change in dihedral angle between the two triangles during the deformation, the smaller the weight, and the more rigidly the two triangles move with respect to each other. As $T(M)$ is a minimum spanning tree, $T(M)$ contains the arcs corresponding to the most rigid components of M , in some sense the skeleton of the surface.

As in Section 3, we compute $S^{(t)}$ by traversing $T(M)$. However, unlike in Section 3, setting the vertex coordinates of a vertex v of $S^{(t)}$ using two paths from the root of $T(M)$ to two triangles containing v can yield two different coordinates for v . An example of this situation is given in Figure 5, where the coordinates of v can be set by starting at $root(T(M))$, and traversing the arcs e_2 and e_3 of $T(M)$ or by traversing the arcs e_1 , e_4 , and e_5 of M . The different coordinates computed for v in $T(M)$ are *candidate coordinates* of v . Our algorithm computes the coordinates of each vertex $v \in S^{(t)}$ as the average of all the candidate coordinates of v .

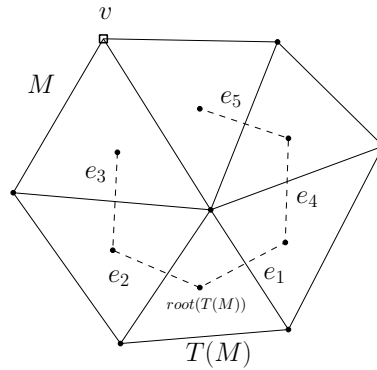


Fig. 5. A mesh M with its dual minimum spanning tree $T(M)$.

Let us analyze the maximum number of candidate coordinates that can occur for a vertex in $S^{(t)}$. Let e denote an edge of M such that $D(e)$ is in $T(M)$. Let v denote the vertex of $S^{(t)}$ opposite e in the triangle corresponding to an endpoint

12 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

of $D(e)$, such that the coordinates of v are computed when traversing $D(e)$. This situation is illustrated in Figure 6. Let d_1 and d_2 denote the total number of candidate coordinates of the two endpoints of e . By traversing $D(e)$, we compute $d_1 d_2$ candidate coordinates for v . We can therefore bound the number of candidate coordinates of v computed using the path through $D(e)$ by $d_1 d_2$. Note that the number of candidate coordinates for the two endpoints of the first edge is one. Furthermore, each vertex v can be reached by at most $\deg(v)$ paths in $T(M)$, where $\deg(v)$ denotes the degree of vertex v in M . As each path in $T(M)$ has length at most $m - 1$, where $m = O(n)$ is the number of triangles of M , we can bound the total number of candidate coordinates in $S^{(t)}$ by $\sum_{v \in V} 2^{m-1} \deg(v) = 2n2^{m-1}$, where V is the vertex set of M .

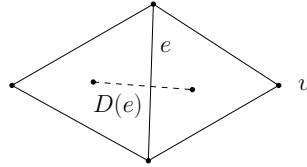


Fig. 6. Illustration of how to bound the number of candidate coordinates of v computed using the path through $D(e)$.

The use of an exponential number of candidate coordinates is computationally expensive. We use the following heuristic to reduce the total number of candidate coordinates considered at each vertex down to $O(n)$. We traverse each edge of $T(M)$ at most once. We traverse $T(M)$ in depth-first order. When an edge $D(e)$ is traversed, we add candidate coordinates to one vertex v as shown in Figure 6. However, we only add at most a linear number of candidate coordinates to v as described below. Denote the vertices of e by $v_0(e)$ and $v_1(e)$, denote the number of candidate coordinates that were added to $v_0(e)$ and $v_1(e)$, respectively, during the traversal of $T(M)$ before traversing the edge $D(e)$ by d_1 and d_2 , and let $v_0(e)$ be the vertex of e that was updated more recently in the traversal of $T(M)$. Let the candidate coordinates of $v_0(e)$ ($v_1(e)$, respectively) be given by $c_1^1, \dots, c_1^{d_1}$ ($c_2^1, \dots, c_2^{d_2}$, respectively) ordered from the least recently to the most recently added candidate coordinate. When traversing $D(e)$, we add d_2 candidate coordinates to v by computing coordinates of v based on the candidate pairs $(c_1^{d_1}, c_2^1), \dots, (c_1^1, c_2^{d_2})$.

This strategy computes $O(n)$ candidates per vertex of $S^{(t)}$, and hence a total of $O(n^2)$ candidates, thereby avoiding the computation of an exponential number of candidate coordinates. To find the final coordinates of a vertex v , we average all of the candidate coordinates of v .

Our algorithm finds a triangular mesh $S^{(t)}$ corresponding to $s^{(t)}$ that is isometric to $S^{(0)}$ and $S^{(1)}$ if such a mesh exists, because all of the candidate coordinates are equal in this case and taking their average yields the desired result. If there is

no isometric mesh corresponding to $s^{(t)}$, our algorithm finds a unique mesh. By choosing $T(M)$ as a minimum spanning tree based on weights representing rigidity, we allocate rigid parts of the model more emphasis than non-rigid parts. The reason for this is that in most near-isometric morphs, triangles close to non-rigid joints are deformed more than triangles in mainly rigid parts of the model. We conclude with the following.

Proposition 1. *Let $S^{(0)}$ and $S^{(1)}$ denote two connected triangular meshes that are isometric to each other and let $s^{(0)}$ and $s^{(1)}$ denote the corresponding shape space points, respectively. We can compute a unique triangular mesh $S^{(t)}$ representing the information given in the linear interpolation $s^{(t)}$, $0 \leq t \leq 1$ of $s^{(0)}$ and $s^{(1)}$, in exponential time. We find a triangular mesh $S^{(t)}$ corresponding to $s^{(t)}$ that is isometric to $S^{(0)}$ and $S^{(1)}$ if such a mesh exists.*

6. Experiments

The experiments were conducted using an implementation in C++ on an Intel (R) Pentium (R) D with 3.5 GB of RAM. OpenMP was used to improve the efficiency of the algorithms. To compute the minimum spanning tree $T(M)$, the boost graph library¹⁷ was used.

We use the following data sets in our experiments. The armadillo models are chosen from the AIM@SHAPE repository.^c The models contain 331904 triangles. The horse, elephant, and flamingo models were created and used by Sumner et al.¹⁹ The horse models contain 16843 triangles, the cat models contain 14410 triangles, the elephant models contain 84638 triangles, and the flamingo models contain 52895 triangles. The head models are from the CAESAR database.¹⁴ The correspondence between the two head models was found using the approach by Xi et al.²² The models contain 22091 triangles.

^c<http://shapes.aimatshape.net/releases.php>

14 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*

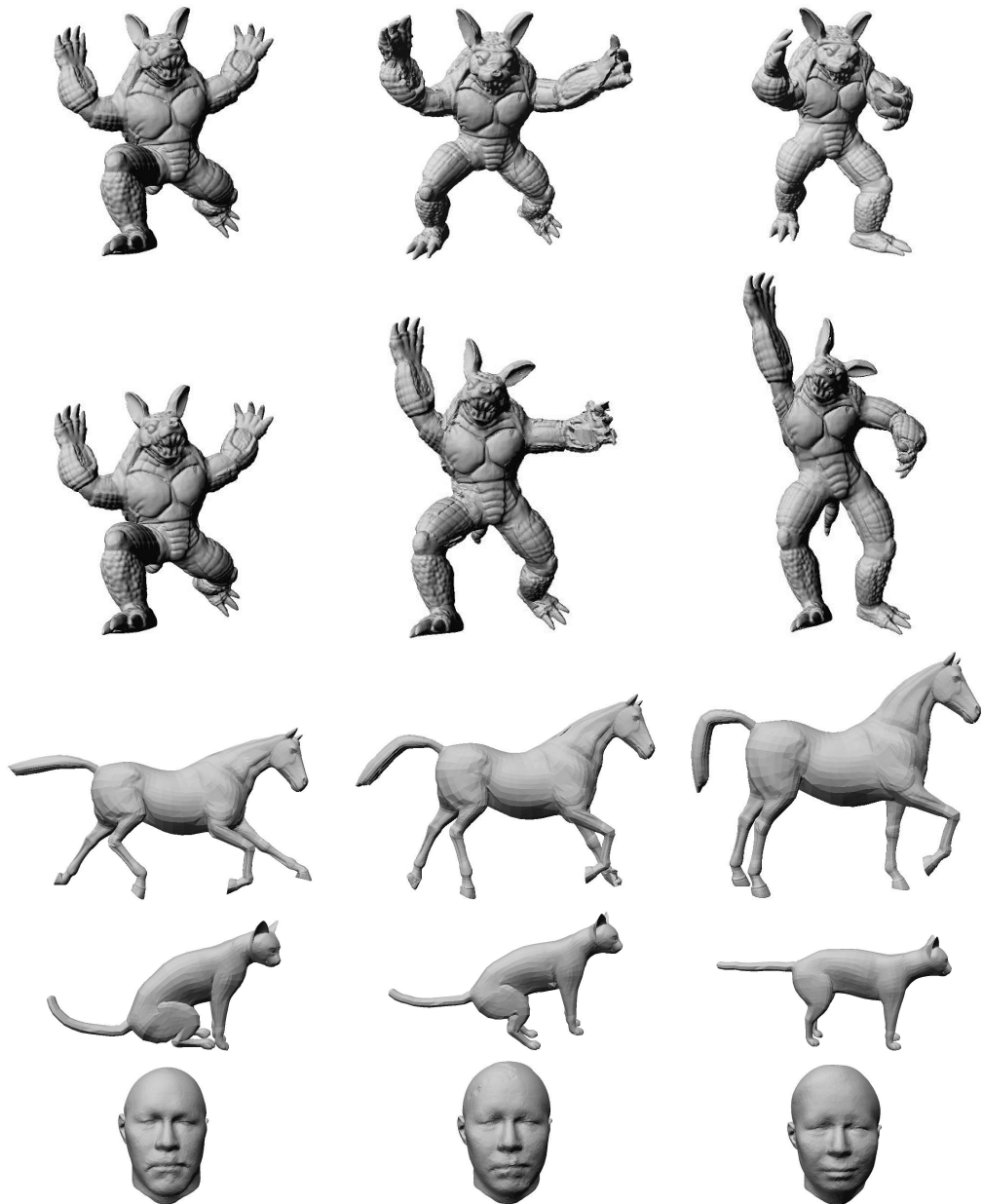


Fig. 7. The rows show start and end poses on the left and right, respectively. The intermediate poses shown in the middle columns are interpolations for $t = 0.5$.

6.1. Results

Figure 7 shows a number of morphs. Note that the intermediate poses are visually pleasing and that most details are preserved during the morph.

Many data sets stem from the digitalization of real world objects using laser range scanners or image-based reconstruction. This type of data is often noisy. We show that our approach is robust to noise. Figure 8 shows the effect of adding noise to the meshes. The top row shows the morph between two meshes of an elephant. The bottom row shows the morph after adding random Gaussian noise to the models. The noise has mean $0.005r$, where r is the resolution of the mesh. Note that while the morph of the noisy models is a noisy mesh, the overall shape of the elephant is preserved.

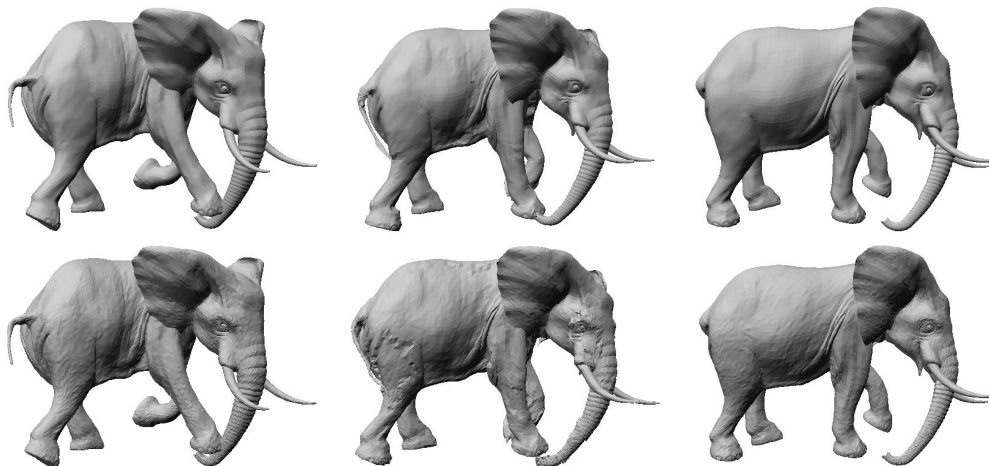


Fig. 8. *Effect of adding Gaussian noise to the models. The rows show start and end poses on the left and right, respectively. The intermediate poses shown in the middle columns are interpolations for $t = 0.5$.*

For all of the experiments conducted, we measured the time efficiency to compute one intermediate pose at $t = 0.5$. The running times range from a few seconds for the smaller models to about 30 minutes for the armadillo models. Note that the implementation is non-optimized and experimental.

6.2. Comparison

We compare the morphs obtained using our approach to the morphs obtained using the approach by Kilian et al.⁹ Recall that Kilian et al.'s approach is more powerful than our approach in that it is possible to extrapolate isometric deformations. In our implementation of Kilian et al.'s approach, we use the limited-memory Broyden-Fletcher-Goldfarb-Shanno scheme,¹³ which is a quasi-Newton method, to minimize

the energy. We halve the number of vertices in the mesh repeatedly as outlined by Kilian et al. until about 2000 vertices are left. Since we only compute the morph for $t = 0.5$, we do not use multiple resolutions on the shape path.

Figure 9 shows the morphs obtained by our method in column (b) and the morphs obtained by Kilian et al.'s method in column (c). For all of the shown models, the morphs obtained by Kilian et al.'s approach are noisy although the energy term that is minimized contains a small regularization term. The regularization term does not ensure that surface normals are varied smoothly between poses. The noise occurs at extremities of the mesh that are simplified significantly in lower resolutions. We emphasize that the results in column (c) were obtained using our implementation and that better results may be obtained by adjusting both the mesh resolutions and the shape path resolutions. However, the choice of the resolutions is not straightforward.

Our approach explicitly interpolates between surface normals without the need to solve an energy minimization problem. As a result, for all of the examples shown in column (b), the morphs preserve geometric details.

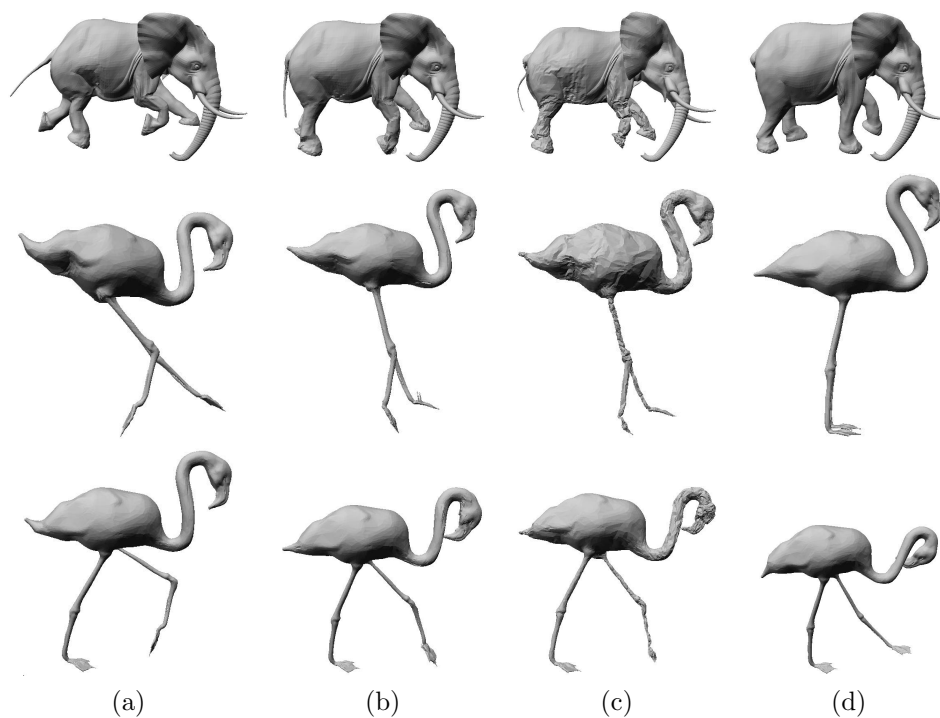


Fig. 9. Comparison to approach by Kilian et al. (a): Start pose. (b): Interpolation for $t = 0.5$ using our approach. (c): Interpolation for $t = 0.5$ using our implementation of Kilian et al.'s approach. (d): End pose.

7. Conclusion

We presented an approach to morph efficiently between near-isometric poses of triangular meshes in a novel shape space. The main advantage of this morphing method is that the most isometric morph is always found in linear time when triangulated 3D polygons are considered. For general triangular meshes, the approach cannot be proven to find the optimal solution. However, we present an efficient heuristic approach to find a morph for general triangular meshes that does not depend on solving a non-linear optimization problem.

The presented experimental results demonstrate that the heuristic approach yields visually pleasing results. The running time of the approach can be improved by employing a multi-resolution scheme.

An interesting direction for future work is to find an efficient way of morphing triangular meshes while guaranteeing that no self-intersections occur. For polygons in two dimensions, this problem was solved using an approach based on energy minimization.⁷

Acknowledgments

We thank Martin Kilian for sharing his insight on the topic of shape spaces with us. We thank Pengcheng Xi for providing us the data for the head experiment.

References

1. Marc Alexa. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 19(2–3): 105–114, 2003.
2. Marc Alexa. Recent advances in mesh morphing. *Computer Graphics Forum*, 21(2):173–196, 2002.
3. Helmut Alt and Leonidas J. Guibas. *Discrete Geometric Shapes: Matching, Interpolation, and Approximation*. In: Jörg-Rüdiger Sack and Jorge Urrutia (Editors). The handbook of computational geometry, pages 121–153. Elsevier Science, 2000.
4. Prosenjit Bose, Joseph O’Rourke, Chang Shu, and Stefanie Wuhrer. Isometric Morphing of Triangular Meshes. In *Proceedings of the Canadian Conference on Computational Geometry*, 2008.
5. Ilya Eckstein, Jean-Philippe Pons, Yiyang Tong, C. C. Jay Kuo, and Mathieu Desbrun. Generalized surface flows for mesh processing. In *Symposium on Geometry Processing*, pages 183–192, 2007.
6. Gene Golub and Charles van Loan. *Matrix Computations*, Third Edition. The John Hopkins University Press, 1996.
7. Hayley N. Iben, James F. O’Brien, and Erik D. Demaine. Refolding planar polygons. In *Proceedings of the 2006 Symposium on Computational Geometry*, pages 71–79, 2006.
8. David Kendall. Shape manifolds, Procrustean metrics and complex projective spaces. *Bulletin of the London Mathematical Society*, 16:81–121, 1984.
9. Martin Kilian, Niloy J. Mitra, and Helmut Pottmann. Geometric modeling in shape space. *ACM Transactions on Graphics*, 26(3), 2007. Proceedings of SIGGRAPH.
10. Vladislav Kraevoy and Alla Sheffer. Mean-Value Geometry Encoding. *International Journal of Shape Modeling*, 12(1):29–46, 2006.

- 18 *S. Wuhrer, P. Bose, C. Shu, J. O'Rourke, and A. Brunton*
11. Francis Lazarus and Anne Verroust. Three-dimensional metamorphosis: a survey. *The Visual Computer*, 14:373–389, 1998.
 12. Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear Rotation-invariant Coordinates for Meshes. *ACM Transactions on Graphics*, 24(3), 2005. Proceedings of SIGGRAPH.
 13. Dong C. Liu and Jorge Nocedal. On the limited memory method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.
 14. Kathleen Robinette, Hans Daanen, and Eric Paquet. The caesar project: A 3-d surface anthropometry survey. In *3-D Digital Imaging and Modeling*, pages 180–186, 1999.
 15. Thomas W. Sederberg, Peisheng Gao, Guojin Wang, and Hong Mu. 2-d shape blending: an intrinsic solution to the vertex path problem. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 15–18, 1993.
 16. Ken Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985.
 17. Jeremy G. Siek, Lie-Quan Lee, and Andrew Lumsdaine. *The boost graph library: user guide and reference manual*. Addison-Wesley Longman Publishing Co., Inc., 2002.
 18. Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 109–116, 2007.
 19. Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Transactions on Graphics*, 23(3):399–405, 2004. Proceedings of SIGGRAPH.
 20. Yue Man Sun, Wengping Wang, and Francis Chin. Interpolating polyhedral models using intrinsic shape parameters. *The Journal of Visualization and Computer Animation*, 8(2):81–96, 1997.
 21. Vitaly Surazhsky and Craig Gotsman. Intrinsic morphing of compatible triangulations. *International Journal of Shape Modeling*, 9:191–201, 2003.
 22. Pengcheng Xi, Won-Sook Lee, and Chang Shu. Analysis of segmented human body scans. In *Graphics Interface*, 2007.
 23. Oliver van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A Survey on Shape Correspondence. *Computer Graphics Forum*, In press, 2011.
 24. Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Transactions on Graphics*, 24(3):496–503, 2005.