A Complete Framework for 3D Mesh Morphing

Bogdan Mocanu*, Titus Zaharia†
Institut Télécom / Télécom SudParis, ARTEMIS Department
UMR CNRS 8145 MAP5 Evry, France

Abstract

In this paper, we propose a complete 3D mesh morphing technique dedicated to closed, genus-0 3D models. The two 3D objects are first mapped onto a common spherical domain. For this purpose, we employ a parameterization method, based on a modified version of the Gaussian curvature, that returns a locally flattened version of the original model with a quasi convex structure, which can be simply projected onto the unit sphere. By overlapping the two embeddings and warping them in a suitable manner with the aid of RBF functions, we establish a correspondence between the models. We also introduce a new method to create a metamesh model that share the topology of both input objects and which can easily be transformed from the source model into the target. The experimental results obtained show that the obtained transitions are smooth, consistent with respect to both geometry and topology, and visually pleasant.

CR Categories: I.3.7. [Computer Graphics]: Three-Dimensional Graphics and Realism – Animation.

Keywords: morphing, spherical parameterization, Gaussian curvature, radial basis functions, metamesh.

1 Introduction

When investigating nature we can observe that living creatures are able to change their shape over time in a smooth and gradual manner. Plants or animals are growing gradually. The growth process is a highly complex mechanism that generates internal forces which constrain organisms to modify their shape and appearance. Starting from a simple seed, a plant can grow in a complete tree, with stem, branches and leaves. Such evolutions and changes that occur in the natural world have attracted the attention of a significant number of computer science researchers who have tried over time to simulate such phenomena by computer, creating different animation techniques for shape transformation of artificial objects. Such techniques are called morphing or metamorphosis.

*e-mail: bogdan.mocanu@it-sudparis.eu
†e-mail: titus.zaharia@it-sudparis.eu

VRCAI 2012, Singapore, December 2 – 4, 2012.
© 2012 ACM 978-1-4503-1825-9/12/0012 $15.00

The morphing process can be defined as the construction of an animated sequence corresponding to a gradual transition between two different objects, so-called source (initial) and target (final) models. Morphing techniques are widely used in computer graphics to simulate the transformation between two completely different objects or to create new shapes by a combination of other existing shapes. They have a variety of applications ranging from special effects in film industry and other visual arts to medical imaging and scientific purposes.

Elaborating efficient 3D morphing methods could have a strong economical impact on the graphics industry, specifically within the framework of content/special effects production.

In this paper, we propose a 3D mesh morphing methodology which can ensure high quality transition sequences. The obtained transitions are smooth, consistent with respect to both geometry and topology, and visually pleasant.

The rest of this paper is organized as follows: After a brief recall of the most important methods dedicated to 3D mesh morphing, Section III details the steps of the proposed algorithm. Section IV presents and discusses the experimental results obtained. Finally, Section V concludes the paper and opens some perspectives of future work.

2 Related Work

When elaborating morphing between 3D objects, two main questions have to be solved: (1) how to establish the correspondence between the models, and (2) what interpolation method is more appropriate to create intermediary states of morphing.

The correspondence problem cannot be directly solved, because of the complexity of the topological and geometric information involved in the case of 3D meshes. Instead, the correspondence is achieved in an indirect manner with the help of parameterization techniques, which consists of establishing a bijective mapping between the mesh surface and a common planar or spherical domain. For example, [Kanai *et al.* 1998] use harmonic mappings to embed two models onto the unit disk, then the parameterizations are merged and a new mesh with connectivity inherited from both models is created. The resulting mesh has a total number of vertices equal to $N_{M_S} + N_{M_T} + N_{Int}$, where N_{M_S} and N_{M_T} are the number of vertices in source and target respectively, and N_{Int} is the number of edge intersections. This will obviously leads to huge meshes in the case of objects described by a large number of faces/vertices. A second drawback of the method consists in the fact that excepting the boundary loop no other feature vertex is set up. Thus, the feature preservation principle cannot be effectively satisfied.

In order to overcome such limitations, an extension of the method is proposed in [Kanai *et al.* 2000]. Here, the user is allowed to select interactively multiple feature points for each of the two

meshes to be morphed. Based on this set of vertices, the models are cut into correspondent patches (also called *tiles*) such that the control points remain on the boundaries. By allocating such corresponding vertices to the boundaries of the tiles, the vertex correspondence is satisfied automatically. Next, the mesh patches are individually parameterized onto the plane and the supermesh is constructed in a similar way as presented in [Kanai *et al.* 1998].

Let us note that the specification of the feature vertices and the way the models are cut into patches have to be performed manually, which is poorly intuitive for the user. However, the quality and precision of this process has a great impact on the resulting correspondence.

Furthermore, for closed genus-0 objects, different authors [Urtasun *et al.* 2004], [Yu *et al.* 2003] describe a user-specified technique to cut the models into patches and then use a similar method as [Kanai *et al.* 1998] to obtain the embeddings. However, since closed genus-0 models are topologically equivalent with the sphere, Alexa [2000] propose to establish the correspondence in the spherical domain. The mesh vertices are projected onto the unit sphere and then a relaxation process that repeatedly places each vertex in the centre of its neighbors is applied. In order to avoid triangle overlapping or mesh collapse problems, Alexa defines sets of anchor points in the parametric domain, which are modified several times during the relaxation process. However, the final embedding is not guaranteed to be valid in all cases.

Lee *et al*. [1999] employ a multiresolution analysis in order to solve the correspondence problem by generating coarse, simplified models of the two input meshes which are used as base domains. Here, MAPS (Multiresolution Adaptive Parameterization of Surfaces) [Lee *et al.* 1998] parameterizations of the source and target objects are first constructed. The MAPS algorithm uses a course mesh built through successive removal of a maximally independent set of vertices, followed by re-triangulation of the resulting holes. A set of feature points specified by the user is here needed. Such feature points are never removed in the simplification process, thus guaranteeing that they are included in the base domain. After the base domains are aligned in a semi-automatic manner, the source map is projected onto the target base domain. The initial projection is improved through an iterative relaxation procedure [Turk 1992]. However, the method can lead to fold-overs and the user intervention is required in order to manually fix the problem.

Another multiresolution mesh morphing approach is the MIMesh (Multiresolution Interpolation Mesh) technique proposed by Michikawa *et al.* [2001]. First, a base interpolation mesh M^0 is manually created by the user from the input meshes M_S and M_T, which are partitioned into several patches according to the faces of the base interpolation mesh in a similar way as proposed in [Kanai *et al.* 2000]. Each patch is then parameterized in the planar domain, using a shape preserving mapping algorithm [Floater 1997]. Next, a subdivision fitting scheme, inspired from the remeshing technique of Guskov *et al.* [2000], is applied.

A different approach is proposed by Wu *et al*. [2007] which directly maps the connectivity of the source mesh onto the target mesh without partitioning or flattening the models onto 2D plane/3D sphere domain. Based on a mean-value Laplacian fitting scheme they succeed to compute a shape preserving correspondence directly in 3D space.

Athanasiadis *et al.* [2012] present a method that performs morphing in a completely automatic manner, but which works well only in the case where similar source and target models, belonging to the same category are considered. Object alignment, feature detection and feature point matching is performed automatically with the help of a geometric local characteristic, so-called concavity intensity, inspired from [Stamati 2007]. This feature is combined with an algorithm that detects the rapid variations of the surface normal, in order to obtain a region growing method that results in sets of points corresponding to the object individual features. The object features are then represented with the help of a connectivity graph that captures their adjacency information.

Once the source and target models are parameterized and aligned with respect to their corresponding features of interest, the final step necessary in the morphing process is the interpolation between objects. This can be done simply by determining the trajectory of the corresponding vertices on the representation obtained in the previous step. The simplest way to interpolate between these points is a linear interpolation, but which could cause self intersections.

Different approaches attempt to improve the quality of the morphing sequence. Alexa *et al*. [2001] propose to interpolate Laplacian coordinates instead of Cartesian coordinates in order to determine trajectories of the mesh vertices during the transformation. Since the Laplacian coordinates are not invariant under rotation and scaling the results are not satisfactory in all cases. Better morphing results can be obtained using the dual Laplacian coordinates [Hu *et al.* 2007], or the pyramid coordinates introduced in [Sheffer and Kraevoy 2004].

The analysis of the state of the art shows that none of the existing techniques guarantees valid morphing sequences. In addition, the construction of complex meta-meshes is most often required in order to construct a common connectivity, adapted for the two shapes to be morphed. The approach proposed in this paper and described in the following section aims at overcoming such limitations.

3 The Proposed Mesh Morphing Algorithm

Figure 1 illustrates the main steps involved in the proposed mesh morphing algorithm. We first apply, for both source and target models, a pose normalization and a mesh simplification schemes as two pre-processing steps. Then, we employ our previous work presented in [Mocanu and Zaharia 2011] that introduces a spherical parameterization method for 3D genus-0, two-manifold meshes. The method exploits a modified version of the Gaussian curvature in order to iteratively flatten the 3D models until some quasi-convex version of the initial objects are obtained. Such objects can be then directly projected onto the unit sphere, ensuring valid embeddings.

Next, the mesh structure is reconstructed through a progressive mesh sequence which optimally reinserts the vertices removed in the simplification process. Then, in order to align the main features of the two models a warping scheme based on RBF functions is employed. Additional constraints are here introduced in order to maintain all vertices on the sphere and to avoid triangles overlapping. The two embeddings are then overlapped in order to create a supermesh. We introduce here a novel and simple method of overlapping that creates a mesh structure able to

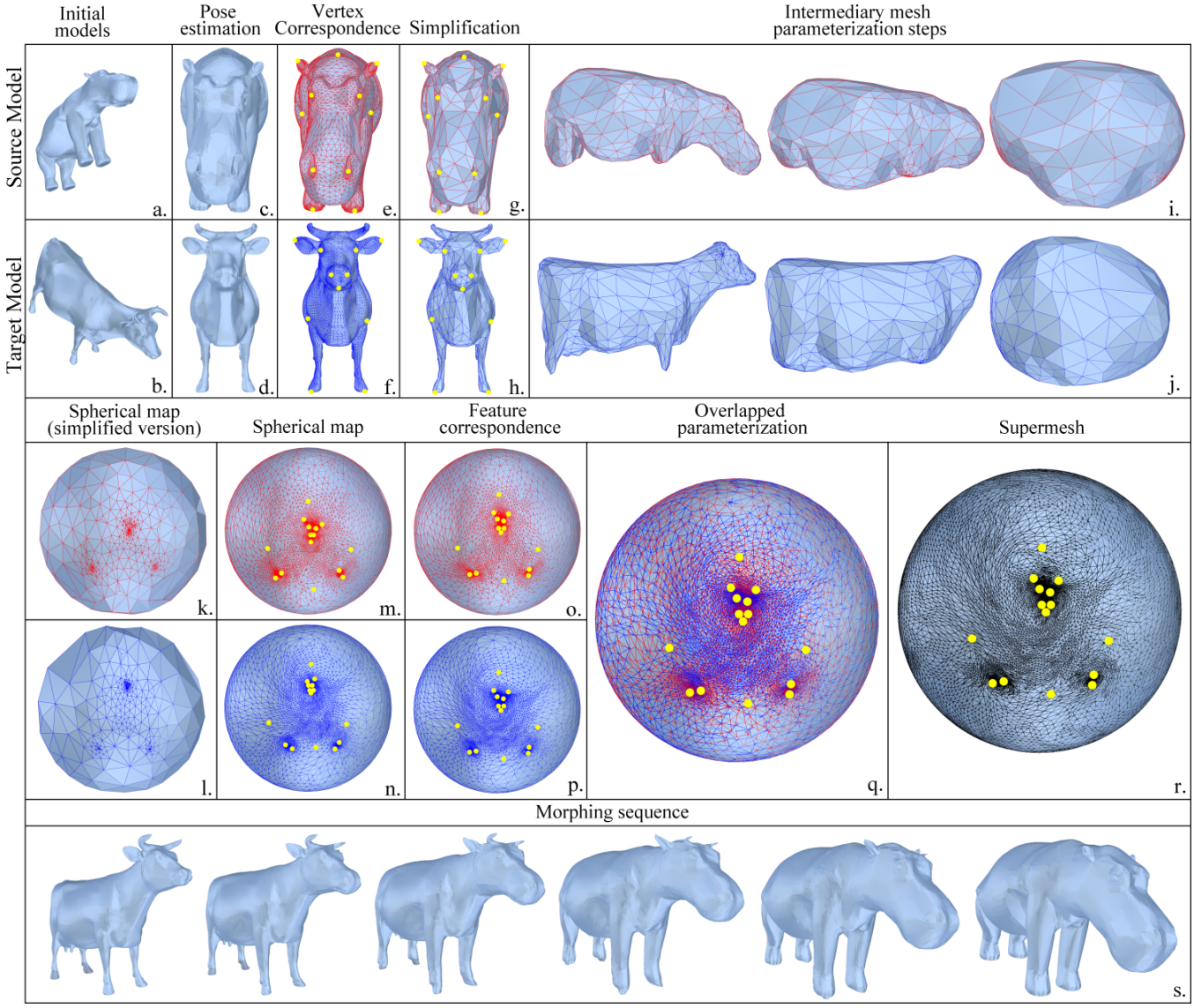

Figure 1: *Steps involved in our morphing process.*

approximate both the source and target topologies and geometries. The various stages involved in our mesh morphing approach are detailed in the following sections.

3.1 3D Model Normalization

Mesh models can be generated using a variety of techniques (*e.g.* 3D designers use CAD software, optical devices of 3D scanners) and, therefore, most of the 3D models available over the internet may have arbitrary scales, orientations and positions in the 3D space. For this reason, we first employ a PCA-based normalization [Jolliffe 2002] in order to align the object with respect to its principal axes and scale it to the unit sphere.

3.2 Mesh Simplification

The parameterization process may require relatively important computational resources when dealing with highly complex meshes, described by thousands of vertices/triangles. The goal of this step is to produce coarser versions of the input meshes by iteratively reducing the number of vertices and triangles. The proposed approach is based on the well known edge collapsing operator introduced in [Hoppe 1993]. However, instead of using the measure proposed in [Hoppe 1993] in order to establish the order of the removed vertices, which suffers from a high computational complexity, we have adopted the quadric error metric proposed by Garland and Heckbert [1997]. In contrast with the original technique presented in [Garland and Heckbert 1997], which allows the contraction of arbitrary pairs of vertices, in our case we join solely vertices that define an edge in the mesh. This constraint is useful for preserving the original mesh topology.

Let us note that the simplification process was also used by others authors, in different ways, in the context of mesh parameterization. For example, Praun and Hoppe [2003] exploit a mesh decimation technique in order to reduce the original mesh to a tetrahedron. However, in contrast with [Praun and Hoppe 2003] our decimation process is controlled such that the simplified mesh obtained still preserve the main geometrical features of the initial model. Thus, the simplification process ends when the mean square error between two simplified versions M' and M" of the

original model exceeds a pre-established threshold T_{err}. This measure allows us to dynamically determine the simplification level where the mesh geometry variation becomes important. In our experiments, we have chosen empirically a value of 0.0025 for the T_{err} parameter and an interval of 100 collapsed edges between M' and M". These values provide a good compromise between quality of the resulting simplified meshes and the computational time required.

The proposed decimation algorithm yields high quality results that preserve the initial model's shape and topology (Figure 2) even for drastic simplification rates, in a relatively short time.

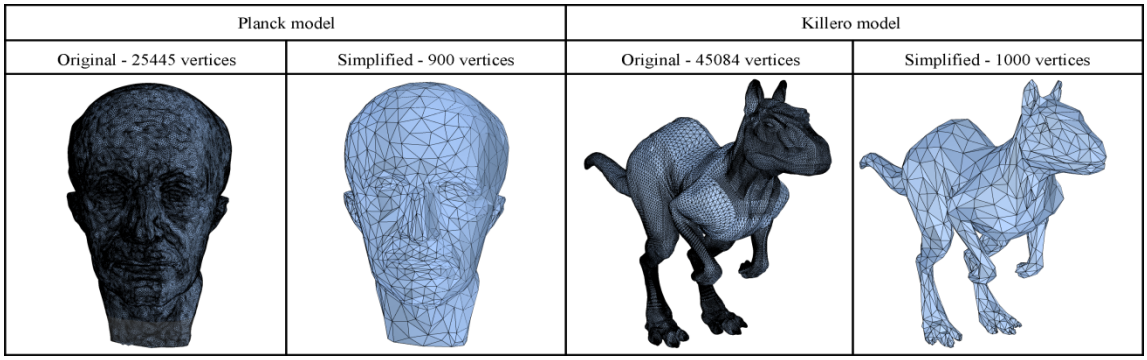

Figure 2: *Examples of mesh simplification.*

3.3 Spherical Mapping

When considering a morphing process between two different 3D meshes, the main difficulty to be overcome concerns the topological problems related to different connectivities, numbers of vertices/faces that can describe the source and target shapes. Obviously, it is impossible to associate in a bilateral manner one vertex from the source to one vertex in the target model. The only solution is to consider the mapping between the two meshes as a mapping between their corresponding $\mathbb{R}^3$ surfaces. Thus, for each vertex on the source (resp. target) model, a correspondent point on the target (resp. source) model has to be identified.

However, directly establishing such a correspondence is quite impossible, because of the complexity and diversity of shapes that can be modeled with 3D meshes. Instead, an indirect mapping method is preferred, which consist of:

1. Parameterizing both source and target models onto a common, parametric domain;

2. Warping the parametric domains in order to ensure a feature correspondence between the two 3D shapes to be morphed.

In order to solve the first problem we employ here a parameterization algorithm based on one of our previous work presented in [Mocanu and Zaharia 2011] and recalled here below. The algorithm exploits the Gaussian curvature and consists in the following three steps:

1) *Iterative Flattening based on Gaussian Curvature*

We compute first for each vertex p of the mesh a modified version of the Gaussian curvature K_p as follows:

$$K_p = (2\pi - \sum_i \alpha_i)/(\chi_S + \sum_i A_i(p)/3) \quad (1)$$

where $A_i(p)$ is the area of each triangle adjacent to p and α_i the angle determined by p_i, p, and p_{i+1}. We denote with $\{p_i \in Neighbors(p) | i = \overline{1,l}\}$ the set of the one-ring neighbor vertices and with χ_S the average triangle area, computed over the entire mesh. For the vertex p_{max} with the maximum absolute value of the Gaussian curvature, we compute the position p'_{max} as the barycenter of its neighboring nodes. If the Euclidian distance between new and initial positions $||p'_{max} - p_{max}||$ is superior to a threshold $dist$, its position is changed to p'_{max}. Otherwise, the considered vertex is not affected and the algorithm selects as a candidate the following highest curvature vertex, reiterating the process. If the vertex is displaced to the new position we re-compute K_p for that vertex and for its neighbors. The process is repeated recursively ensuring that salient mesh vertices are firstly detected and processed. Thus, after each iteration, the algorithm leads to a locally flattened version of the 3D mesh model. At the end of the process, a sphere-like surface is obtained (Figure 3).

The classical Gaussian curvature was also used by others authors [Ben *et al.* 2008] in the context of mesh parameterization. However, in contrast with these methods which try to identify the high-curvature vertices in order to concentrate the entire mesh curvature in some points, we aim to determine such vertices in order to distribute the curvature to its neighbors and thus construct models with constant curvature values, like the unit sphere.

In addition, a modified version of the Gaussian curvature is here introduced (*cf.* equation (1)). The classical Gaussian curvature privileges the selection of vertices located in densely sampled mesh regions, where the triangle areas tend to zero. Therefore, we introduced the correction factor χ_S that makes it possible to reinforce, in the selection process, the influence of the angular defect term $(2\pi - \sum_i \alpha_i)$ and thus to avoid long loops in densely sampled regions characterized by low values of triangle areas.

2) *Spherical Projection*

We apply step 1 for a number *It* of iterations, after which, employing a ray cast operation, we check if the mesh can be projected onto the unit sphere. If all mesh vertices are visible from the object's gravity center, the mapping onto the sphere is simply obtained by a vertex projection which ensures a bijective parameterization. If the visibility condition is not satisfied, step 1 is re-iterate *It* times.

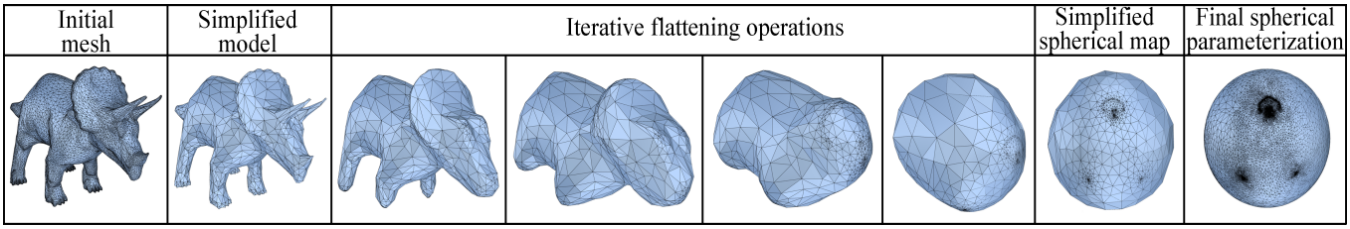

Figure 3: *Spherical parameterization process.*

3) *Vertex Split Sequence*

The last step in the spherical parameterization process employs the iterative re-insertion, in the mesh structure, of all vertices removed after applying the simplification operation as presented in Section 3.2. The algorithm exploits the fact that a contraction operation is invertible. For each edge collapse, a corresponding inverse operator, called vertex split, can be defined. The objective is to re-insert a removed vertex in the mesh structure without generating triangle flips or degenerate faces. This requires a position optimization of the vertex to be inserted. In order to solve this problem we have adopted a simple, yet efficient optimization procedure, illustrated in Figure 4.

The first ring neighborhood from which the considered vertex was removed (Figure 4a) is first subdivided in order to obtain a set of potential positions (Figure 4b) where the vertex to be inserted. Each face is split after a 1-to-4 triangles scheme and 3 levels of such subdivisions are performed. A sub-set of valid possibilities (*i.e.*, position which do not lead to overlaps or degenerate triangles) is then determined (Figure 4c). In order to accomplish this task, we establish for each potential position if the new edges that would form, intersect the boundary edges defined by the first ring neighborhood.

If no intersection is produced then the position is considered valid. Among them, the vertex which provides the optimal angular distribution of the corresponding triangles is determined (Figure 4d). In order to reach this objective, we select the position which yields the maximal value of the minimal angle of the adjacent triangles. Let us note that if the mapping is an embedding prior the vertex split operation, then it should remain also valid after the insertion.

Figure 3 illustrates the entire process of mesh parameterization for the 3D TRex model.

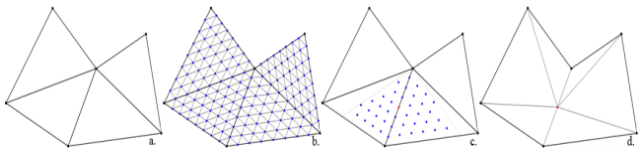

Figure 4: *Vertex insertion operation: (a) initial configuration; (b) polygon subdivision; (c) set of valid positions; (d) final retained position and the new configuration.*

3.4 Feature correspondence

After the parameterization of the two source and target models, we can directly overlay the spherical embeddings to obtain a rough correspondence, apply an arbitrary interpolation procedure and obtain a morphing sequence. However, such an approach would fail keeping aligned the relevant characteristics of the input objects in the intermediate morphing models. Thus, in order to ensure that the features are preserved during the morphing process it is necessary to re-place the user specified corresponding feature points such that they share the same position in the parameter domain. Such a re-placement requires a global deformation of the whole parametric domain, such that the corresponding meshes should be smoothly deformed and without foldovers. The process is referred to as *mesh warping*.

In order to accomplish this task we make use of radial basis functions (RBF). The RBF technique allows to displace all mesh vertices based only on the known displacement of some control points (feature vertices). The displacement is characterized by an interpolation function s, which is defined as the sum of a set of radial basis functions, as described in the following equation:

$$s(p_j) = \sum_{i=1}^{n_c} \alpha_i \phi(\|p_j - p_{ci}\|) + P(p_j) \qquad (2)$$

$$\forall j = 1, \dots, V,$$

where V is the total number of mesh vertices, n_c the number of control points, $p_{c_i} = p_{c_i}(x_{c_i}, y_{c_i}, z_{c_i})$ their spatial positions, P a given polynomial function of degree |P|, and ϕ is the given radial basis function defined with respect to the Euclidian distance (||.||).

The coefficients of the polynomial P and the coefficients $\alpha_i = [\alpha_{ix}, \alpha_{iy}, \alpha_{iz}]$ can be determined from the following interpolation conditions:

$$\forall j = 1, \dots, n_C \quad , \quad s(p_{c_i}) = d_{ci} \qquad (3)$$

$$\sum_{i=1}^{n_c} \alpha_i q(x_{ci}) = 0 \qquad (4)$$

where d_{c_i} is a vector specifying the displacement of the control vertex p_{c_i}, and q represents all polynomials with a degree equal to or less than |P|.

Based on the results in [Boer *et al.* 2007], we have adopted the CTPS C^2_a radial basis function since it ensures a good trade-off between displacement accuracy and level of distortions. The CTPS C^2_a function is expressed as:

$$\phi_r(\xi) = \phi\left(\frac{x}{r}\right) = 1 - 30\xi^2 - 10\xi^3 + 45\xi^4 - \\ -6\xi^5 - 60\xi^3 \log(\xi) \qquad (5)$$

where r represents the support radius that control the influence of a vertex displacement.

In addition, we have successively decomposed the RBF deformation in a number of steps (between 10 and 100), since

directly determining a global RBF deformation would lead to a mesh surface which would not lie on the unit sphere. In addition, such an approach avoids fold-overs and self intersections. In order to guarantee that the final embedding remains valid we propose projecting the mesh back onto the unit sphere after each step.

Additionally, in order to further reduce the distortion rate we constrain both source and target feature vertices to move in their middle position.

Figure 5 presents an example of two 3D closed models with feature vertices already specified by user, their initial spherical mappings and their final embeddings. The corresponding pairs of vertices are aligned in the parametric domain wit the help of the warping process described here-above.

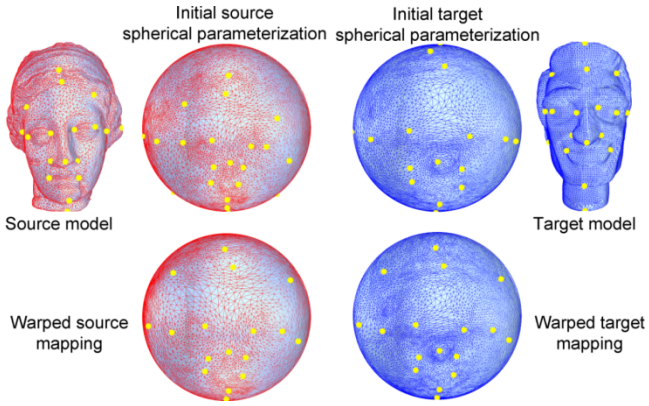

Figure 5: *Feature vertices correspondence through spherical embeddings warping (Igea-ManHead case).*

3.5 Pseudo metamesh construction

Once the two input models are parameterized in a common domain and the main features of the objects are aligned properly, the next natural step in a morphing framework is to establish a one-to-one correspondence between the models shapes.

In order to accomplish this task the method at hand is to overlap the two embeddings and to build a new mesh structure, called metamesh, which can represent both models. The most frequently used approach exploited by numerous morphing methods [Kanai *et al.*1998], [Urtasun *et al.* 2004], [Alexa 2000], [Lee and Huang 2003], [Zhu and Pang 2009] consists of merging the two topologies into a single one by inserting edges of one model into the structure of the other. However, such an approach significantly increases the number of mesh triangles and is very challenging due to numerical instabilities when computing intersections between source and target edges.

In order to overcome this limitation, we introduce a simple yet efficient technique to create a pseudo supermesh which avoids tracking the edge intersections. In addition, our method reduces drastically the number of vertices normally needed in a supermesh structure. We call our structure pseudo-metamesh since it is not created in the classical manner based on edge intersection, and also it only approximates the two models shapes. We initialize first the supermesh structure with the one of the target parameterization. Then, for each source parametric vertex, we establish the supermesh triangle in which it can be projected by employing a ray-triangle intersection test. Once we determine the face in which a source vertex lies, we split the triangle with the help of a 1-to-4 scheme, as illustrated in Figure 6.

The process is repeated until all source vertices are processed such that each triangle in the subdivided mesh includes a unique vertex of the source mesh. Obviously, the obtained pseudo-metamesh does not have any more a valid triangular mesh structure since, after a triangle subdivision, the adjacent faces are not triangles anymore. Thus, a mesh retriangulation is required. At the end, the final retriangulated pseudo-metamesh contains only the target vertices and the new vertices obtained by split operations.

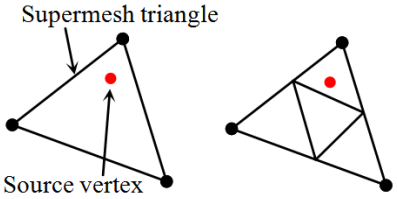

Figure 6: *1-to-4 subdivision scheme.*

The retriangulation process can be easily accomplished if we store the elements of the metamesh structure in appropriate lists which are updated accordingly after each triangle split. We consider $V\{p_1, \dots, p_{N_V}\}$ and $F\{f_1, \dots, f_{N_F}\}$ the metamesh list of vertices and faces respectively, where N_V and N_F denote the initial numbers of vertices and faces, after the initialization of the metamesh with the target model. For each vertex p_i, the list of adjacent faces $F_{i/adj}$ is known.

In order to illustrate the retriangulation process, let us consider the example in Figure 7. The triangle f_k defined by vertices p_A, p_B, p_C (Figure 7.a) is split after 1-to-4 subdivision scheme resulting three new faces $f_{N_F+1}, f_{N_F+2}, f_{N_F+3}$, which are added in the F list. The most inner triangle obtained after the subdivision process will take the place of the initial triangle in the F list. Also three new vertices, $p_{N_V+1}, p_{N_V+2}, p_{N_V+3}$, are added in the V list.

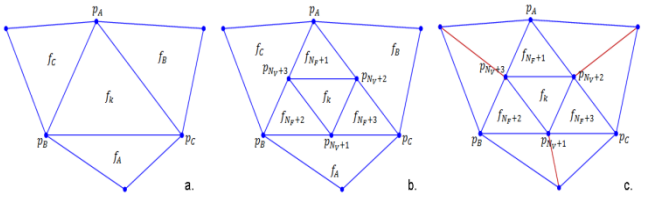

Figure 7: *Mesh retriangulation: (a) the pseudo-metamesh before the subdivision; (b) the pseudo-metamesh obtained after the 1-to-4 subdivision scheme; (c) retriangulated pseudo-metamesh.*

The lists of the adjacent faces is updated as follows:

- p_A replaces the adjacent face f_k with f_{N_F+1};
- p_B replaces the adjacent face f_k with f_{N_F+2};
- p_C replaces the adjacent face f_k with f_{N_F+3};
- the new vertex p_{N_V+1} add in the $F_{N_V+1/adj}$ list the following triangles: $f_{N_F+2}, f_k, f_{N_F+3}$;
- the new vertex p_{N_V+2} add in the $F_{N_V+2/adj}$ list the following triangles: $f_{N_F+3}, f_k, f_{N_F+1}$;
- the new vertex p_{N_V+3} add in the $F_{N_V+3/adj}$ list the following triangles: $f_{N_F+1}, f_k, f_{N_F+2}$;

Let us note that the triangles f_A, f_B, f_C are not added in the lists of adjacent faces of the new vertices.

The list of faces is traversed and the "*triangles*" with more than 3 vertices are detected. In Figure 7, we establish, for example, that face f_C is not well defined and the vertex p_{N_V+3} split the edge $e(p_A, p_B)$ in two halves. This is simply established by analyzing the lists of faces adjacent to vertex p_{N_V+3}. If two vertices of a triangle have only one common adjacent face then that face must be retriangulated. In our example, p_A and p_B have only the face f_C adjacent. Thus, f_C will be split into two triangles and the metamesh lists updated accordingly. The two new faces are verified if should be further split. In this manner, the final retriangulated pseudo-metamesh contains only the target vertices and the new vertices obtained by triangle split operations.

The next step aims to establish the 3D positions of these vertices relatively to the source and target shape. The 3D position of the new vertices relatively to the target shape can be easily established since we know that after each split operation, the new vertices are inserted at the middle of an existing edge. The 3D positions of all pseudo metamesh vertices relatively to the source shape can be computed employing a point-in-triangle test and using the barycentric coordinates (α, β, γ). Considering that a parametric metamesh vertex p^{H_M} lies in the source spherical domain (H^S) on a triangle $f^{H_S}(p_i^{H_S}, p_j^{H_S}, p_k^{H_S})$ then its position p^S relative to the original face $f^S(p_i^S, p_j^S, p_k^S)$ on the source shape can be computed using the barycentric coordinates:

$$p^S = \alpha p_i^S + \beta p_j^S + \gamma p_k^S \quad (6)$$

Figure 8 illustrates an example of pseudo-metamesh obtained with the proposed algorithm. It can be observed that the mesh structure remains simple without too many vertices and in the feature regions the metamesh is adaptively remeshed accordingly with the input models in order to construct a connectivity well-adapted for both source and target models.

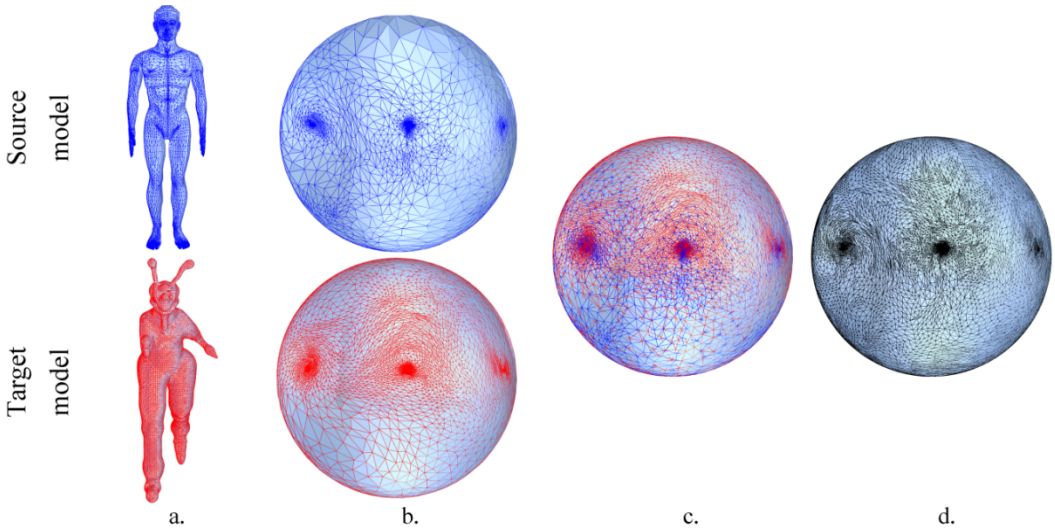

Figure 8: *Pseudo metameshes: (a) original models; (b) spherical parameterization; (c) overlaid maps; (d) final pseudo metamesh.*

Table 1 presents the characteristics of some pseudo-metameshes obtained, in terms of number of vertices and triangles compared with the original models. Note that in most of the cases the pseudo metamesh number of vertices does not exceed the sum of the source and target vertices, which is quite a remarkable result.

Table 1: *Pseudo metamesh characteristics*

	Model	V	F	Pseudo Metamesh	
				V	F
Source	Man	14603	29202	34796	69588
Target	Alien	16267	32530		
Source	Head1	17358	34712	22467	44930
Target	Head2	7896	15788		
Source	Dino	16996	33988	31082	62080
Target	Horse	19851	39698		
Source	Cow	11610	23216	13386	26768
Target	TRex	2832	5660		
Source	Igea	15002	30000	24789	49574
Target	Head1	17358	34712		

3.6 Interpolation

Since the metamesh structure is able to approximate both the source and target models, the in-between shapes at various time steps of the morphing transformation can be now calculated by finding proper paths for each vertex connecting the initial position p_i^S to the final position p_i^T in the metamesh. As in the majority of morphing applications [Kanai *et al.* 2000], [Alexa 2000], [Ahn *et al.* 2004], [Athanasiadis *et al.* 2012], we employ here the linear interpolation, described by the following equation:

$$p_i^t = p_i^S + t(p_i^T - p_i^S) \quad (7)$$

where p_i^t is the position of the i^{th} vertex of the metamesh at the moment of time t . p_i^S and p_i^T are the extreme vertex positions at the moment $t = 0$, and $t = 1$ respectively.

Even if the linear interpolation can lead in some cases to self-intersections of the mesh due to large differences between source ad target models, we have adopted this type of interpolation because of its simplicity and possibility to operate in real time. However, more sophisticated interpolation schemes, such as the

ones introduced in [Alexa 2001], [Hu *et al.* 2007] can be considered in this framework.

4. Experimental Results

A prototype application that allows the user to interactively operate with 3D models and control the morphing process has been developed. The intuitive interface permits user to select the correspondent vertices in the two models or to save the processed meshes at any time. Figure 9 shows different views of the user interface layout. Inside a view, the left window display the source model, while the right one displays the target object. Once the computation of the metamesh is completed, this is displayed on the left part.

Figures 10-14 illustrates some examples of metamorphosis between different 3D models obtained using our algorithm. The considered subset of objects consists of 3D closed genus-0 manifold models with various complexities and shapes. All the models are freely available over the Internet and are part of the Princeton and MPEG-7 databases. We can observe that in all cases the resulting morphing sequences ensure a gradual and visually pleasant transition between source and target models. In addition, the pseudo metamesh proposed is able to adapt to both source and target shapes.

5. Conclusions and Perspectives

In this paper, we proposed a novel morphing method that permits smooth and natural transformations between 3D closed genus-0 meshes. The technique involves a minimum user intervention through a dedicated GUI to specify only some vertices of correspondence.

A spherical mapping has been considered as intermediary step, followed by a feature matching that employs a mesh warping scheme based on radial basis function. Furthermore, we have introduced a simple, but efficient technique of mesh merging to create a pseudo supermesh structure that approximates well both source and target 3D shapes.

Perspectives of future work concern: (a) the interpolation mechanism: linear interpolation should be replaced with other alternatives in order to avoid self-intersection at the level of in-between models; (b) the re-triangulation of the pseudo supermesh which leaves room for optimization.

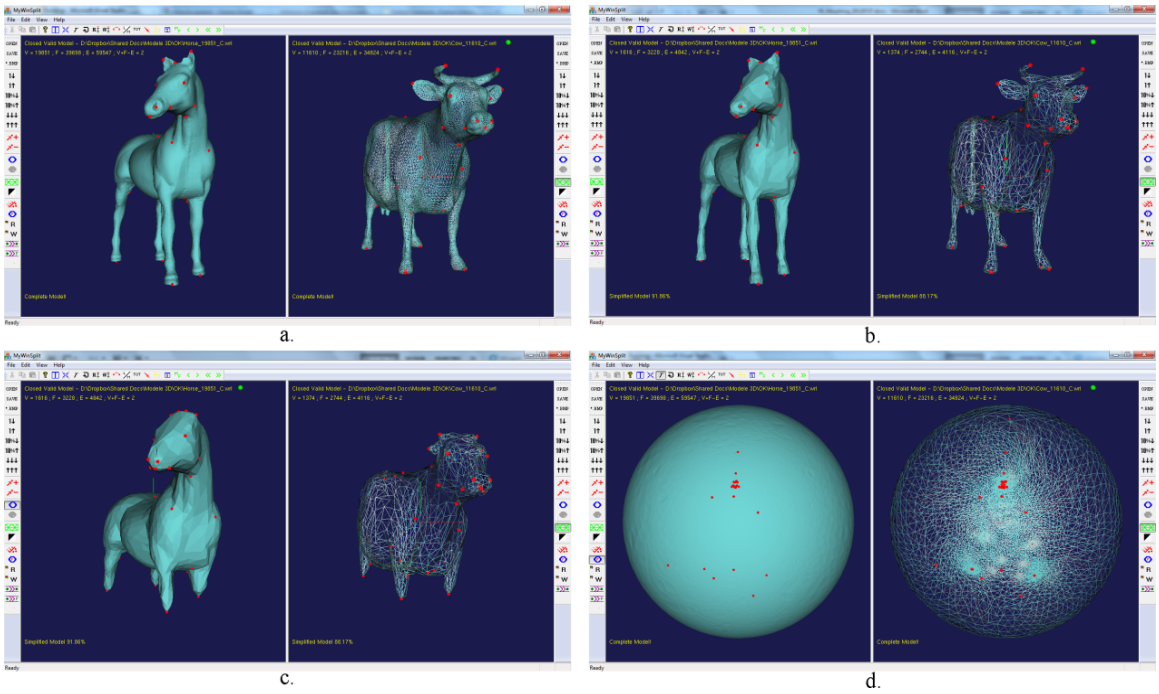

Figure 9 : *Graphical user interface: (a) view with the input models; (b) view during mesh simplification; (c) view during parameterization; (d) view with the final spherical embeddings*

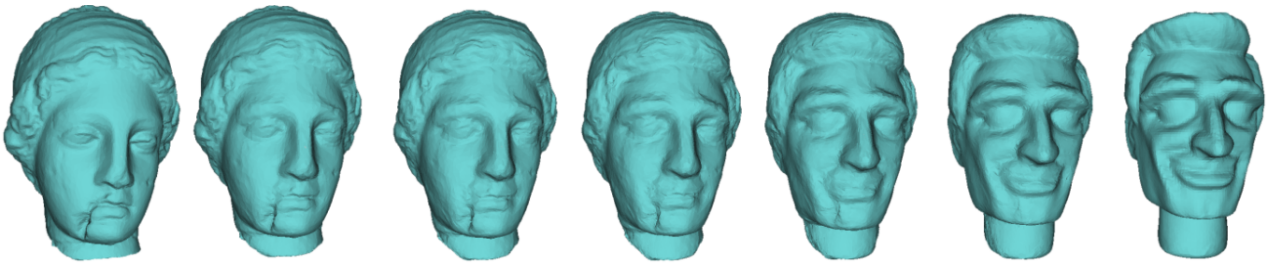

Figure 10: *Morphing between Igea and ManHead models.*

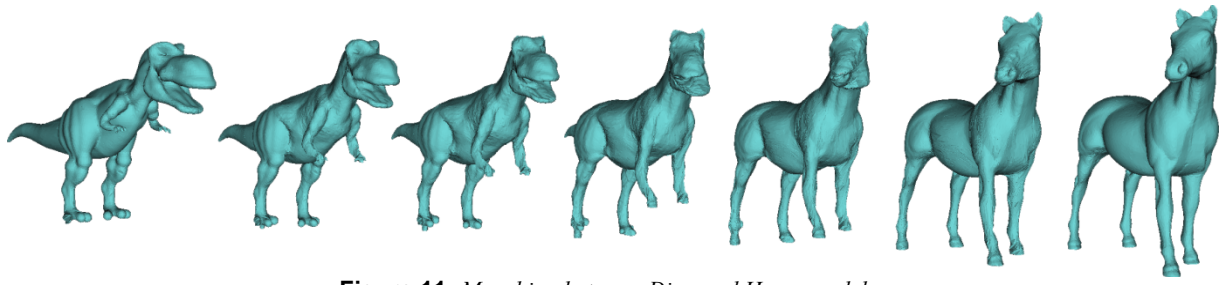

Figure 11: *Morphing between Dino and Horse models.*

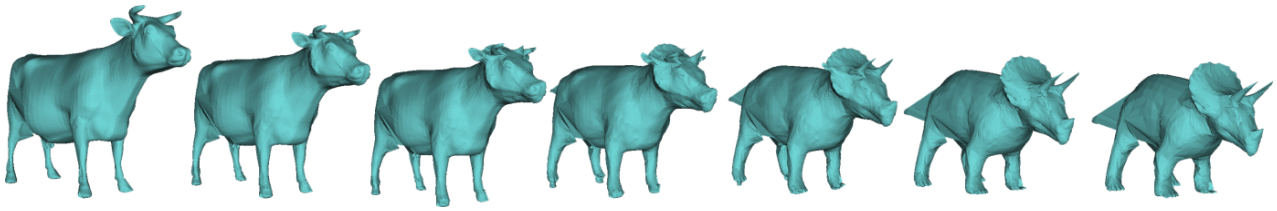

Figure 12: *Morphing between Cow and TRex models.*

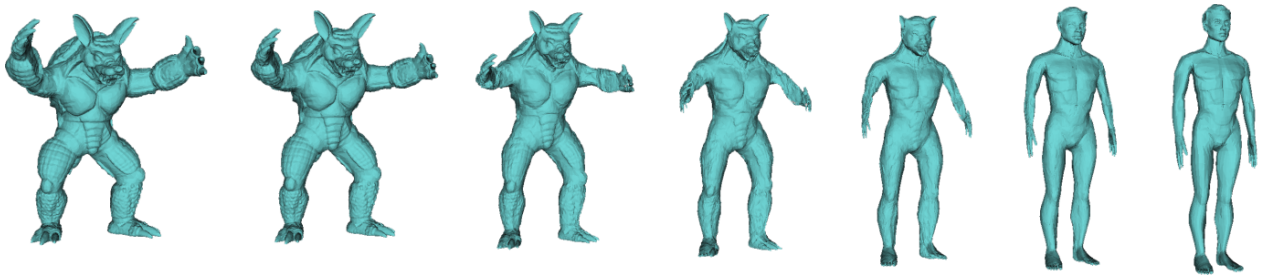

Figure 13: *Morphing between Armadillo and Man models*

Figure 14: *Morphing between DinoSkeleton and Dino.*

References

AHN, M., LEE, S., AND SEIDEL, H. 2004. Connectivity transformation for mesh metamorphosis. *In SGP '04: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, New York, USA, 75–82.

ALEXA, M., 2000. Merging polyhedral shapes with scattered features. *The Visual Computer*, vol. 16, 26-37.

ALEXA, M. 2001. Local control for mesh morphing. *Proceedings of Shape Modeling International*, 209-215.

ATHANASIADIS, T., FUDOS, I., NIKOU, C., AND STAMATI, V. 2012. Feature-based 3D morphing based on geometrically constrained spherical parameterization. *Computer Aided Geometry Description*, vol. 29, 2-17.

BEN-CHEN, M., GOTSMAN, C., AND BUNIN, G. 2008. Conformal flattening by curvature prescription and metric scaling. Computer Graphics Forum, vol. 27, no. 2, 449-458.

BOER, A., SCHOOT, M. S., AND BIJL, H. 2007. Mesh deformation based on radial basis function interpolation. *Computers & Structures*, vol. 85, 784-795.

FLOATER, M. S., 1997. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design,* vol. 14(3), 231-250.

GARLAND, M., AND HECKBERT, P. S., 1997. Surface Simplification Using Quadric Error Metrics. *In 24th Annual Conference on Computer Graphics and Interactive*, 209-216.

GUSKOV, I., VIDIMCE, K., SWELDENS, W., AND SCHRÖDER, P., 2000. Normal meshes. *In Computer Graphics (Proc. SIGGRAPH2000)*, ACM Press, New York, 95–102.

HOPPE, H. 1993. Mesh Optimization. *In Proceedings of ACM SIGGRAPH*, 19-26.

HU, J., LIU, L., AND WANG, G. 2007. Dual Laplacian morphing for triangular meshes. *Computer Animation and Virtual Worlds*, vol. 18(4/5), 271-277.

JOLLIFFE, I. T. 2002. Principal component analysis. *2nd edition, Springer*, New York.

KANAI, T., SUZUKI, H., AND KIMURA, T. 1998. Three-dimensional geometric metamorphosis based on harmonic maps. *The Visual Computer*, vol. 14, 166–176.

KANAI, T., SUZUKI, H., AND KIMURA, F. 2000. Metamorphosis of Arbitrary Triangular Meshes. *IEEE Computer Graphics and Applications*, 62-75.

LEE, A. W.F., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D., 1998. MAPS: Multiresolution Adaptive Parameterization of Surfaces. *Computer Graphics, SIGGRAPH '98 Proceedings*, 95–104.

LEE, A., DOBKIN, D., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution Mesh Morphing. *Proceedings of SIGGRAPH 99*, 343-350.

LEE, T. Y., AND HUANG, P. H. 2003. Fast and intuitive metamorphosis of 3d polyhedral models using SMCC mesh merging scheme. *IEEE Trans. Visualizaton Computer Graphics*, vol. 9(1), 85–98.

MICHIKAWA, T., KANAI, T., FUJITA M., AND CHIYOKURA, H., 2001 Multiresolution interpolation meshes. *In 9th Pacific Conference on Computer Graphics and Applications, IEEE*, 60–69.

MOCANU, B., AND ZAHARIA, T. 2011. Direct spherical parameterization of 3D triangular meshes using local flattening operations, *7th International Symposium on Visual Computing, ISVC-2011*, Part I, LNCS 6938, Las Vegas, 611–622.

PRAUN, E., AND HOPPE, H. 2003. Spherical parameterization and remeshing. *ACM Transactions on Graphics*, vol. 22, no. 3.

SHEFFER, A., AND KRAEVOY, V. 2004. Pyramid Coordinates for Morphing and Deformation. *Proc. 3D Data Processing, Visualization and Transmission Conference (3DPVT)*, 68-75.

STAMATI, V. AND FUDOS, I., 2007. A Feature-Based Approach to Re-engineering Objects of Freeform Design by Exploiting Point Cloud Morphology. *In Proceedings of SPM 2007: ACM Symposium on Solid and Physical Modeling*, Beijing China, 347-353.

TURK, G. 1992. Re-tiling polygonal surfaces. *In Edwin E. Catmull, editor, ACM Computer Graphics (SIGGRAPH '92 Proceedings)*, vol. 26, 55–64.

URTASUN, R., SALZMANN, M. AND FUA, P. 2004. 3D Morphing without user interaction. *EPFL Technical report*.

WU, H. Y., PAN, C., YANG, Q., AND MA, S., 2007. Consistent correspondence between arbitrary manifold surfaces". *In ICCV*, 1–8.

YU, J. B., AND CHUANG, J. H. 2003. Consistent mesh parameterizations and its application in mesh morphing. *Proc. Computer Graphics Workshop*, Hualian.

ZHU, Z. J., AND PANG, M. Y. 2009. Morphing 3D Mesh Models Based on Spherical Parameterization. *International Conference on Multimedia Information Networking and Security*, 309–313.