Three-Dimensional Distance Field Metamorphosis

Daniel Cohen-Or, David Levin and Amira Solomovici
School of Mathematical Sciences
Tel-Aviv University, Ramat-Aviv 69978, Israel

Abstract

Given two or more objects of general topology, intermediate objects are constructed by a distance field metamorphosis. In the presented method the interpolation of the distance field is guided by a warp function which is controlled by a set of corresponding anchor points.

Some rules for defining a smooth least-distorting warp function are given. To reduce the distortion of the intermediate shapes, the warp function is decomposed into a rigid rotational part and an elastic part. The distance field interpolation method is modified so that the interpolation is done in correlation with the warp function.

The method provides the animator with a technique, which can be used to create a set of models forming a smooth transition between pairs of a given sequence of keyframe models. The advantage of the new approach is that it is capable of morphing between objects having a different topological genus, and where no correspondence between the geometric primitives of the models needs to be established. The desired correspondence is defined by an animator in terms of a relatively small number of anchor points.

1 Introduction

Surface metamorphosis is the continuous evolution of a surface from a source surface, through intermediate surfaces, into a target surface. This object-space process of generating intermediate 3D models is to be distinguished from image-space metamorphosis [4, 35]. Figure 1 shows a sequence of images from an animation movie showing a metamorphosis of a source 3D model (on the upper left) into a target 3D model (on the bottom right). Note that the camera roams slightly during the animation and the model casts a shadow that evolves according to the shape of the 3D model. Such view-dependent effects are impossible with image-space metamorphosis.

This paper describes a method that allows the user to create a series of objects which forms a smooth transition in-between keyframe objects. The method presented is explicitly designed for surfaces of solid objects, that is, the boundary, in any dimension, between the inside of the solid and the outside. However, the method is presented for the metamorphosis of surfaces of 3D solid objects.

The problem of blending two surfaces, even polyhedral ones, is not simple. In order to blend two polyhedral models, most techniques require establishing a full correspondence between their structures [18]. However, a correspondence alone does not guarantee a smooth transition from the source model to the target model, and the *vertices' paths* have to avoid troublesome situations such as self-intersections. Two-dimensional shape blending techniques that deal with the vertices' path problem are not easy to extend to 3D ones [12, 29, 31, 33]. Another class of methods, the image metamorphosis techniques [4, 35], has rather straightforward extensions to voxel-space [15, 13, 21]. However, the exact location of the surface of the voxel-based intermediate objects is not explicitly defined.

The problem of 2D shape blending can also be considered as one of body reconstruction from cross-sections [30]. Some of the reconstruction methods use bivariate interpolation after finding a correspondence between the contours of the cross-sections. When the cross-sections are not dense, they may exhibit significant differences in the geometry and topology of the boundary contours. A method that easily handles that issue is the Distance Field Interpolation (DFI) presented and analyzed in [22]. DFI is a general set-valued interpolation method for the reconstruction of an n-dimensional model from a sequence of its $(n-1)$-dimensional cross-sections. It uses a univariate interpolation of the distance field, and performs well for cross-sections of different

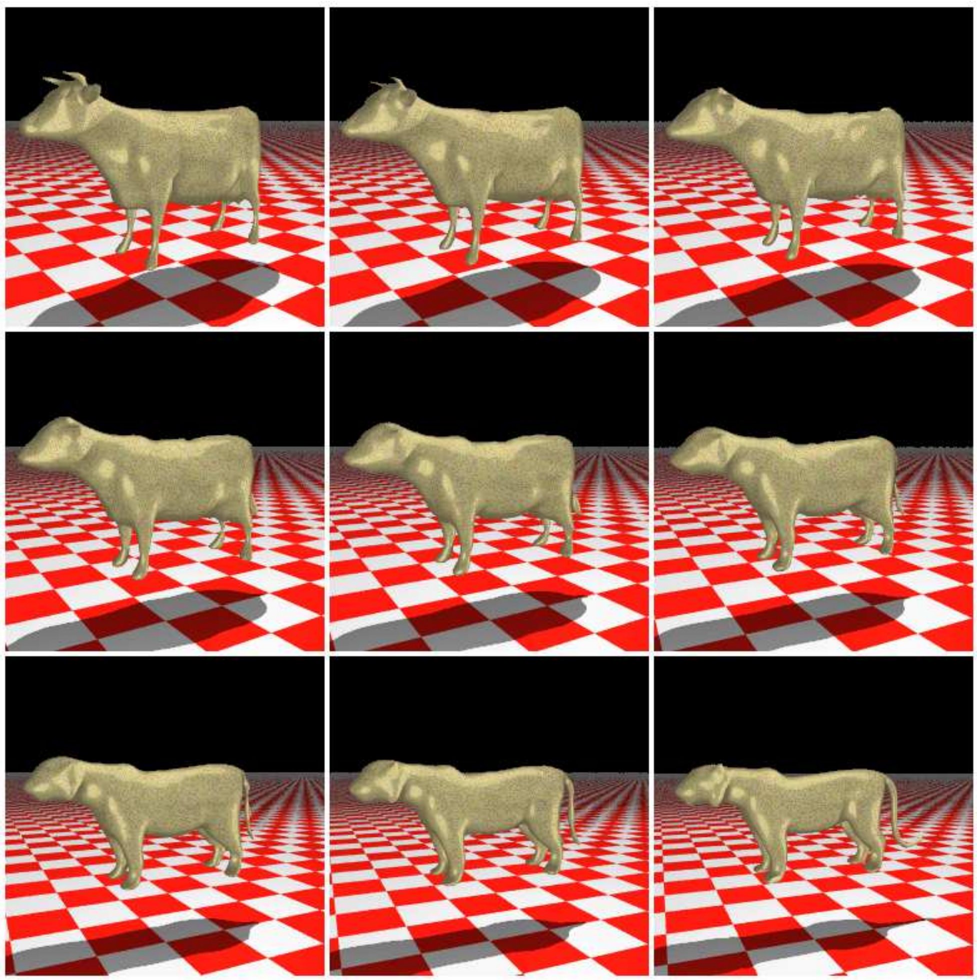


Figure 1: *3D morphing between a cow and a tiger. Note that the camera roams during the animation and the model casts a shadow that evolves according to the shape of the 3D model.*

topological genus in any dimension. The DFI method has been adapted successfully in biomedical reconstruction applications [14].

The metamorphosis presented in this paper is based on the DFI method. However, it is extended in the sense that the interpolation, and thus the blended surface, follows a warp transformation guided by means of a user-defined control. The control is defined by a point-to-point correspondence between prescribed anchor points on the given intermediate object. To avoid large deviations of the in-between shapes from the two original shapes, we decomposed the warp transformation W into a *rigid* (rotation and translation) transformation and an *elastic* transformation, which are separately interpolated. The rigid-elastic decomposition of the warp function, and its particular interpolation are so chosen to minimize the distortion of the intermediate surfaces. The rigid transformation is used to rotate and translate the source object to match the coarse features of the target object, while the finer features of the object are evolved by the elastic part. In [8] we have shown how these principles can be applied to surface reconstruction from 2D cross-sections.

In Section 2 we present the different terms used in the field of morphing, and survey related work. Section 3 describes the DFI method. The warp function, its decomposition and parameterization, are introduced in Section 4. In Sections 5 and 6 we describe some implementation details and discuss the results in Section 7. We conclude with a summary in Section 8.

2 Background - from Warps to Metamorphosis

The problem of transition from one geometrical model to another can be formally stated as follows: Given two models, a source S and a target T, construct a set of transformations $\{W_t \mid t \in [0,1]\}$, such that $W_0 = I$ is the identity transformation, i.e., $W_0(S) = S$, while $W_1(S) = T$. Intermediate models or *in-between* "blends" of S and T are defined by $W_t(S)$, $t \in (0,1)$.

It is important to distinguish between methods operating on a discrete representation, where the transformation is applied to the pixels of an image [4, 20, 35] (image-based methods), or to a voxel representation [13, 15, 21], and methods operating on a combinatorial representation (e.g., polygons and polyhedra), [12, 18, 29, 31, 33, 16].

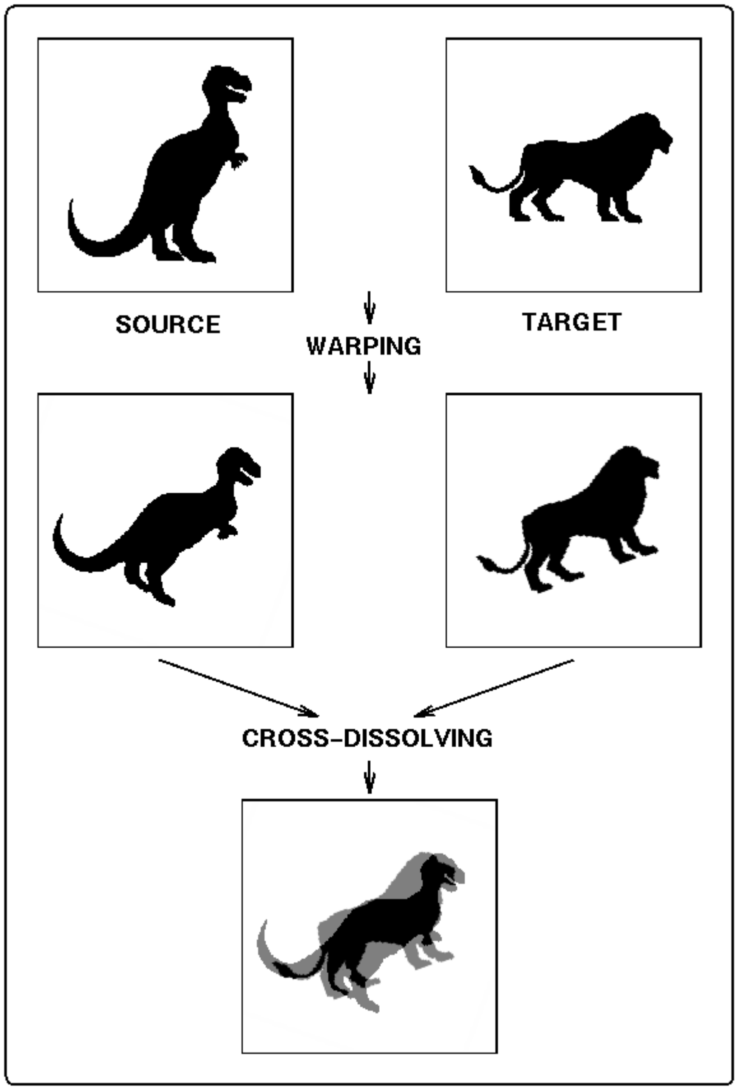


Figure 2: *The image-based morphing process.*

The term *morphing* is used for all methods which gradually and continuously deform S into T, while producing the in-between models. The morphing process is usually composed of *warping* and *interpolation*. Since morphing involves two objects, we perform the warping operation first, to obtain an approximated alignment so that the two shapes will be relatively similar, and then we blend the two warped objects into one. This interpolation process in image-based techniques is often referred to as *cross-dissolving*, where the values of corresponding pixels are blended, yielding the effect of fading-in of the target image with a fade-out of the source image. Figure 2 illustrates the image-based morphing, where cross-dissolving is used to blend the gray-level values of corresponding pixels.

Warping can be generally described as a mapping or a transformation which can be applied either in image space [1, 26, 35] or object space [28, 34, 6]. This mapping essentially distorts the object it is applied to, according to some predetermined constraints, which are usually specified by the user. Such user control over the warping is crucial in obtaining satisfactory results, and can be achieved, for instance, through specifying a predetermined mapping for a set of points (*point-to-point correspondence*).

It is important to point out that, in practice, warping (and hence morphing) requires a good level of user specification to achieve the desired results. All forms of specification (e.g., point-to-point correspondence) are based on the idea of defining the transformation by manipulating only a finite and hopefully small, number of parameters. From these specifications, mathematical techniques should be used to compute the transformation for any point in the warping domain.

Shape-blending algorithms usually refer to techniques operating on a combinatorial representation. Most of the work done here has dealt with polygonal or polyhedral objects [18, 29, 31, 33, 24, 19, 9, 10, 6, 16]. The general method of these algorithms is to displace the vertices, edges and faces of S over time to coincide in position with the vertices, edges and faces of T. This usually requires solving the *vertex correspondence* problem, which determines which vertex travels to which vertex during the deformation process. However, establishing suitable correspondence is difficult, and does not guarantee good behavior of the intermediate models. In addition, a shape blending technique needs to control the *vertex paths*, i.e., the paths along which the vertices travel between their corresponding positions, to avoid self-intersections or major changes in intrinsic geometric properties, such as angles and lengths [12, 18, 29, 31, 33].

Research effort in the metamorphosis field initially focused on the 2D problem. For 2D polygonal shapes, Sederberg and Greenwood [30] proposed a solution to the vertex correspondence problem, while the vertex path problem was dealt with in [29]. Other 2D algorithms which deal with the vertex path of polygonal shapes were proposed by Shapira and Rappoport [31] who used a connecting skeleton to control the interior of the shape as well as its boundary, and by Goldstein and Gotsman [12] who used a multiresolution representation of simple polygons. As for image space techniques, Beier and Neely [4] suggested an algorithm based on fields of influence surrounding two-dimensional control primitives. A treatise of image warping methods can be found in Wolberg's book [35].

Less work has been done in 3D morphing. Chen and Parent [7] briefly addressed an extension of a transformation for piecewise linear 2D contours to 3D objects represented by a set of planar contours. Another shape transformation algorithm was proposed by Kent et al. [18], where the correspondence problem for polyhedral objects was solved by merging their topological structures (see also [16]). Kaul and Rossignac [17] computed the Minkowski sum of scaled versions of the models, and then used it to obtain a smooth transition between the models. Lerios et al. [21] extended the work of Beier and Neely [4] where fields of influence of three-dimensional primitives were used to warp a voxel-space. The voxel-based model was then rendered by volume rendering techniques.

Recently, Sun et al. [33] presented a solution to the vertex path problem of polyhedral models, which is an extension of the intrinsic shape transformation scheme suggested in [29] for 2D polygons. Rather than considering a polyhedron as a set of independent vertices or faces, their algorithm treats a polyhedron as a graph representing the interrelation between vertices, where intrinsic shape parameters, such as dihedral angles and edge lengths, are used for interpolation.

Other 3D metamorphosis algorithms deal with sampled or volumetric representation of the objects. Hughes [15] proposed transforming the object into the Fourier domain, and then interpolating the low-frequencies over time while the high-frequencies are slowly added in, thus minimizing the object distortion caused by its high-frequency components. A method which uses the wavelet transform instead of the Fourier transform, was proposed by He et al. [13]. This method enables a more accurate transition of the iso-surfaces, due to the better localization of the wavelet transform in the spatial domain.

Payne and Toga [25] a distance-field volumetric representation of 3D biomedical objects. Their technique consists of cross-dissolving the distance values of each voxel and then reconstructing the intermediate surfaces out of the intermediate distance-field. However, no warping or correspondence was established between the source and the target object. This lack of control over the morphing process often yields poor results. We present a method that uses the distance-field volumetric representation of 3-dimensional objects, but in addition, establishes a point-to-point warp function between the two models, that can be seen as a way of enforcing topological correspondence and geometrical properties.

3 Distance Field Interpolation

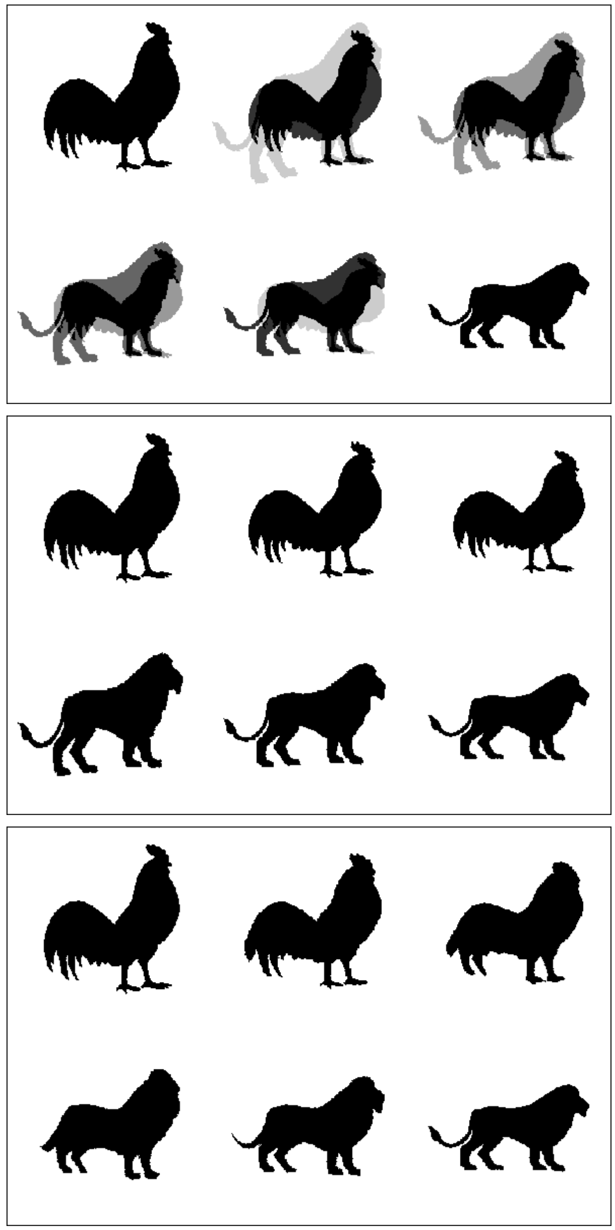
Reconstruction of 3D objects from cross-sections is of major importance in many applications, particularly in biomedical imaging. One class of reconstruction methods builds a polygonal surface by assuming the cross-sections consist of closed polygonal contours [3]. Another class of methods uses a volumetric approach. The entire volume (3D voxel array) is reconstructed from the gray levels of the cross-sections. The missing data is reconstructed by some interpolation of the gray values of the cross-sections. The surface is then defined and reconstructed by iso-surfacing methods. These methods do not yield smooth spatial transformations, as can be seen in Figure 3. Figure 3(a) shows that sections of the source object dissolve instead of spatially transforming into the target object, as in Figure 3(c).

The method of DFI presented in [22] achieves better reconstruction of the surface by interpolating distance values rather than gray values. This method has been successfully adapted in biomedical reconstruction applications (see e.g. [14], [25]).

The DFI method can be described in terms of n-dimensional surface reconstruction from $(n-1)$-dimensional cross-sections ($n > 1$). Consider a solid object $\Omega \subset I\!\!R^n$, discretized or not. Given a finite set of $(n-1)$-dimensional cross-sections of the object,

$$\Omega_{t_j} = \{x = (x^1, \ldots, x^{n-1}) \mid (x^1, \ldots, x^{n-1}, t_j) \in \Omega\} \ , \quad t_0 < t_1 \ldots < t_M \ , \quad (1)$$

we would like to reconstruct Ω. Throughout the paper we use lower indices for numbering entities and upper indices to denote the components of a



(a)

(b)

(c)

Figure 3: *Comparing gray-level interpolation with warping to the DFI with warping. (a) Results of the gray-level interpolation before thresholding, and (b) after thresholding. (c) Results of the DFI technique. Note the discontinuity between the third and the fourth objects in (b).*

vector. Let us define the signed distance fields at the levels $t_0, t_1 \dots t_M$. For $x = (x^1 \dots x^{n-1})$

$$D_{t_j}(x) = \begin{cases} -dist(x, \partial\Omega_{t_j}) & \text{if } x \in \Omega_{t_j} \\ dist(x, \partial\Omega_{t_j}) & \text{otherwise} \end{cases}, \tag{2}$$

where $\partial\Omega_{t_j}$ denotes the boundary of Ω_{t_j}, and *dist* denotes the Euclidean distance in $I\!R^{n-1}$. Now, using univariate interpolation (with respect to the parameter t) we interpolate, between the cross-sections, the distance values of points having the same first $n-1$ coordinates. The resulting interpolant approximates the $n-1$ Euclidean distance between $y = (x^1, \dots, x^{n-1}, t)$ and $\partial\Omega_t = \partial\Omega \mid_{y^n = t}$.

Once this approximated distance field is available, the surface of the object can be determined by the zero points of the distance field, or, in the discretized version, by the boundary between the positive and negative valued lattice points, while the volume itself (its *interior*) is defined as the set of all negative valued points.

Formally, the DFI method defines an approximated domain $\widetilde{\Omega} \approx \Omega$ as follows: For $y = (x^1, \dots, x^{n-1}, t)$ we first define the interpolant $d_x(t)$ ($x = (x^1 \dots x^{n-1})$) by univariate interpolation of the values $\{D_{t_j}(x)\}_{j=0}^{M}$. Now,

$$\widetilde{\Omega} = \{y \mid d_x(t) \leq 0 \} . \tag{3}$$

For more details on the choice of the interpolation method, and for approximation rate analysis, see [22].

Clearly, this method can be used for morphing, where the problem of matching and interpolating the deformations of $(n-1)$-dimensional objects can be viewed as an n-dimensional shape reconstruction. The DFI method works well provided that "corresponding parts" of the two objects are properly aligned. Otherwise, some parts may unexpectedly disappear and reappear. The approach taken in this work to overcome this restriction is the combination of DFI with a proper warp transformation.

4 Warp-Guided DFI Morphing

Let us consider for simplicity the blending of two objects, $S = \Omega_0$ and $T = \Omega_1$. DFI gives us a method of blending by interpolating (linearly) the distance functions D_0 and D_1, as described in Section 3. We call this blending process, no-warp DFI blending. No-warp DFI blending is quite restricted, and may produce unsatisfactory results (see the example in Figure 4). To demonstrate this, consider the case when T is just a rotation of S, and S is just a thin rod. No-warp DFI blending at $t = \frac{1}{2}$ gives either a very small object, or an empty object, which is not the naturally-expected result. On the other hand, no-warp DFI blending works nicely if S is a thin rod and T is just the rod S without its middle third.

Consider now the possibility that T is obtained by a deformation (warp) of S, i.e., $T = W_1(S)$. Further, let us assume that T gradually evolves from S, through a continuum of objects $W_t(S)$, where $\{W_t\}_{t\in[0,1]}$ are smooth transformations, smoothly changing with t, and $W_0 \equiv I$ is the identity transformation. In such a case the blending can be defined by the warp itself; this is pure-warp blending. Pure-warp blending is quite restricted, as it does not allow changes in the genus of the objects.

We now hybridize the two different methods into a more powerful tool [4, 20]. The first step of the hybrid method is to find a smooth warp $\{W_t\}_{t\in[0,1]}$ such that $W_1(S) \approx T$ in some chosen sense. Then the DFI method is applied, now guided by the warp $\{W_t\}$. Note that the warp operates in $\mathbb{R}^{n-1}$ and the interpolation of the distance field is performed along the $n-th$ dimension. As stated above, we first find a smooth warp $\{W_t\}_{t\in[0,1]}$ such that $W_1(\Omega_0)$ approximates Ω_1 as well as possible. In addition, we use the two signed distance functions D_0 and D_1 as defined by Eq. (2). For $x = (x^1, \ldots, x^{n-1})$ we then interpolate the values $D_0(W_0(x))$ and $D_1(W_1(x))$ with respect to the parameter t, denoting the interpolant $d_x(t)$. The approximated domain $\tilde{\Omega}$ defined by the hybrid warped DFI procedure is

$$\tilde{\Omega} = \{y = (W_t(x), t) \;\mid\; d_x(t) \leq 0\ \} . \tag{4}$$

4.1 Obtaining the Approximated Warp W_t

We define the warp W_t so that N points $\{p_{0,i}\}_{i=1}^N$ in the source domain are mapped to corresponding points $\{p_{1,i}\}_{i=1}^N$ in the target domain. Later we refer to these points as *anchor points*. The warp is to be determined so that

$$W_1(p_{0,i}) = p_{1,i}\ , \quad 1 \leq i \leq N\ . \tag{5}$$

The warp function maps the source (level zero) anchor points to their corresponding target (level one) points, while other points in the source domain

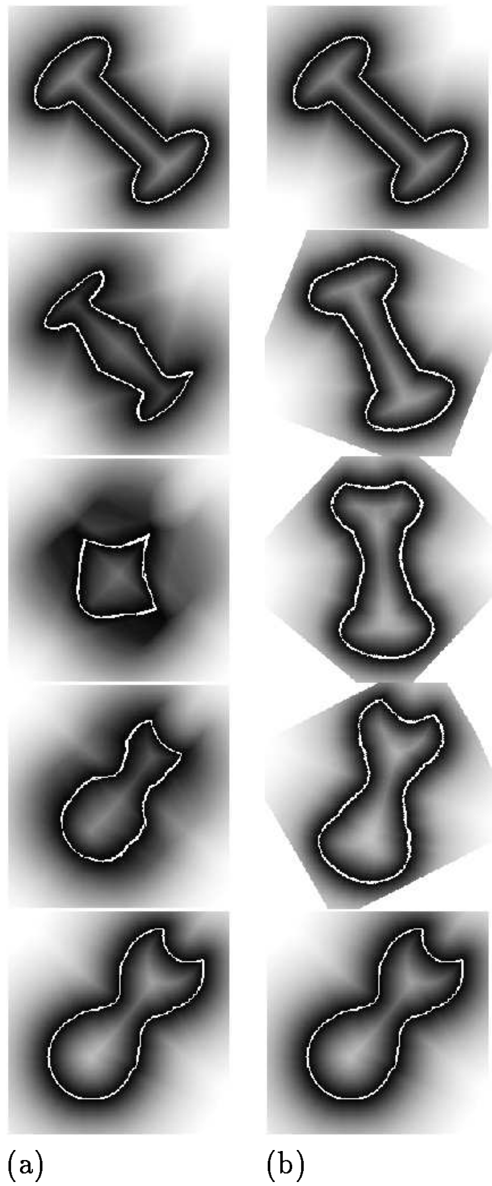
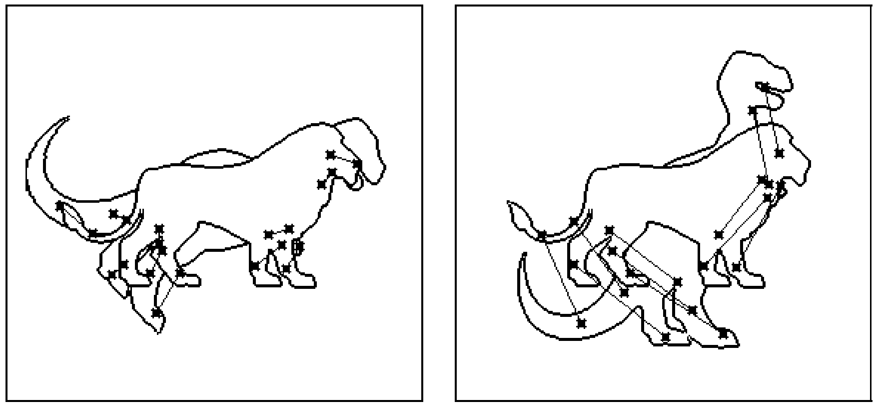


Figure 4: *Comparing the morphing results using the DFI method. (a) Without any warp, and (b) using a proper warp. The objects are defined by the white contours.*



(a) (b)

Figure 5: *The rigid part minimizes the work of the elastic part. (a) The blending with the rigid transformation, and (b), without it.*

comply with those constraints. Defining the warp by anchor points is attractive because it is a concept naturally applicable in any dimension. From the user's point of view, this method is intuitive and very easy to handle. It also has the advantage of being robust and very stable in the sense that the results are not very sensitive to small perturbations in the anchor points.

We now define the warp transformation based on anchor points, starting with the case $n = 3$, i.e., the cross-sections are in $\mathbb{R}^2$. To achieve least distortion of the in-between objects, the warp transformation W_t is decomposed into a *rigid* part and an *elastic* part. To explain the motivation here, let us view a very simple case: Consider the blending of two objects in $\mathbb{R}^2$, $S = \Omega_0$ and $T = \Omega_1$, where $T = R_\theta S$ and R_θ is a rotation in angle θ. There are two obvious options for defining a warp W_t. One is the *linear interpolation* warp $W_t = (1-t)I + tR_\theta$. The other is the *linear rotation* warp $W_t = R_{t\theta}$. Both warps vary smoothly from the identity I at $t = 0$ to R_θ at $t = 1$. However, the second option is the preferable one since it is an isometry for any t, i.e., non-distorting. Now consider the case where T is obtained from S by a rotation R_θ, a translation c, and an *elastic* transformation E, i.e., $T = E(R_\theta S + c)$. In the following we will choose the rotation R_θ and the translation c so that E is as close as possible to I, in some sense (see Figure 5). The warp W_t is then defined by

$$W_t(x) = ((1-t)I + tE)(R_{t\theta}x + tc) \ , \ x \in \mathbb{R}^2 \ . \tag{6}$$

We will describe how to obtain the rotation, the translation and the elastic transformation, and will consider the main case $n = 4$, namely, the morphing of 3D objects.

4.2 The Rigid Transformation in $\mathbb{R}^3$

To start with, we are given two ordered lists of points which, taken together, define pairs of anchor points in $\mathbb{R}^3$, $\{p_{0,i}\}_{i=1}^N$ and $\{p_{1,i}\}_{i=1}^N$. The rigid part of the transformation is defined by the rotation R and the translation c, which minimize

$$Q = \sum_{i=1}^{N} \|Rp_{0,i} + c - p_{1,i}\|^2 , \tag{7}$$

where $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^3$. Imagine that the transformed points $\{Rp_{0,i} + c\}$ are connected by identical elastic springs to the corresponding points $\{p_{1,i}\}$. The form Q represents the elastic energy of this system, and the rigid transformation which minimizes Q, brings Ω_0 to an equilibrium position in the springs' system.

The above least-squares fitting problem of the two sets of anchor points in $\mathbb{R}^3$ is solved using an explicit algorithm, which involves the *singular value decomposition* (SVD) of a 3×3 matrix [2]. As a result, we obtain a 3×3 rotation matrix R and a translation vector c. In order to interpolate the rotation defined by the matrix R, we use the *Quaternion* approach [32], thus obtaining the intermediate rotation matrices R_t, for $t \in (0, 1)$.

Finally, the warp transformation in the 3D case is defined by

$$W_t(x) = ((1-t)I + tE)(R_t x + tc) \ , \ x \in \mathbb{R}^3 \ . \tag{8}$$

4.3 The Elastic Warp in $\mathbb{R}^3$

Once we have computed the rigid transformation, we move on to calculating the elastic transformation. In accordance with Equations (6) and (9), we look for a transformation E, in general, a non-linear transformation, $E : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, such that

$$E(Rp_{0,i} + c) = p_{1,i} \ , \ 1 \leq i \leq N \ . \tag{9}$$

This is a multivariate scattered data interpolation problem, which we suggest solving by using *Radial Basis Functions*, abbreviated RBF [11, 1]. In $\mathbb{R}^3$ this means solving three interpolation problems in $\mathbb{R}^3$, for each component of the vector equation (9).

Radial basis functions have proven to be an effective tool in multivariate interpolation problems of scattered data. Given scattered points $\{x_i\} \subset \mathbb{R}^d$ and corresponding data values $\{F_i\} \subset \mathbb{R}$, we look for an interpolatory function $S(x)$ of the form

$$S(x) = \sum_{i=1}^{N} a_i g\left(\| x - x_i \|\right), \tag{10}$$

such that $S(x_i) = F_i$, $1 \leq i \leq N$. Here $\| \cdot \|$ denotes the usual Euclidean norm on $\mathbb{R}^d$ and $g : \mathbb{R}^+ \to \mathbb{R}$. A function of this form is usually referred to as a *pure radial sum*. Using radial functions reflects the fact that the scattered data has no preferred orientation [11]. Other multivariate approximations, generalizing the univariate splines, use augmented radial sums, where the sum in Eq. (10) is augmented by a low degree polynomial. Our discussion is mainly concerned with the 3-dimensional case ($d = 3$). We define the elastic transformation $E : \mathbb{R}^3 \to \mathbb{R}^3$ as:

$$E(q) = Z(q) + L(q) \tag{11}$$

where L is an affine transformation and Z is of the form:

$$Z(q) = (S_1(q), S_2(q), S_3(q)) \tag{12}$$

where $S_k(q) = \sum_{i=1}^{N} a_i^k g(\|q - q_i\|)$, for $1 \leq k \leq 3$, $q = (q^1, q^2, q^3)^T \in \mathbb{R}^3$, $g : \mathbb{R}^+ \to \mathbb{R}$ is a univariate function, and $\{q_i\}$ for $i = 1, 2, \ldots, N$ are the source anchor points *after* the rigid transformation has been applied to them.

The above definition of the transformation E includes a linear part L. This linear part is added because pure radial sums may yield poor approximation of the transformation for points away from the anchor points. Moreover, the linear part is natural here since we would like to exactly reconstruct those transformations E which are affine, and especially the identity transformation.

A proper choice of g is also important. Choosing g as a function with a fast decay, for example, the Gaussian $g(r) = exp(-r^2/\sigma^2)$ with a small σ, results in a finer local influence of the radial part, which can be used

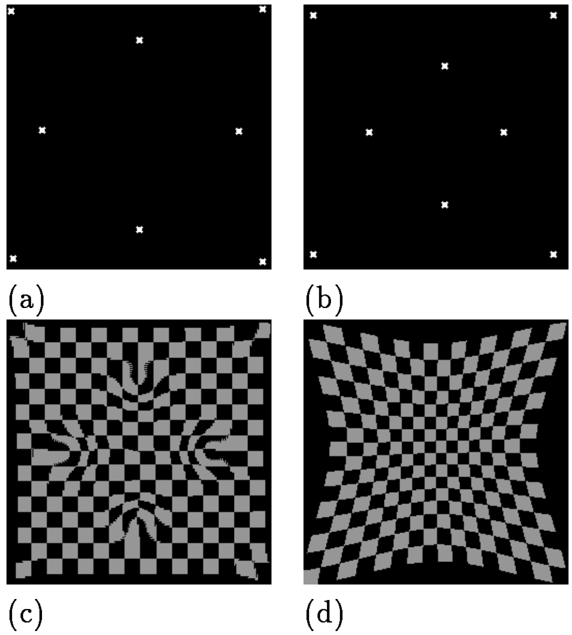


Figure 6: *The different effects when using a g function with a fast decay (Figure (c)) and with a global effect (Figure (d)). The source and target anchor points are shown in Figures (a) and (b), respectively.*

to obtain different effects for various parts of the object. Other choices are $g(r) = r^2 \log(r)$ in $I\!R^2$ and $g(r) = r^3$ in $I\!R^3$, which have global effect [11, 26]. The different effects in $I\!R^2$ are demonstrated in Figure 6. In the following we describe the computation of the elastic warp in $I\!R^3$.

After computing the rigid part, namely, the rotation R and the translation c, the elastic mapping E is defined by the following N interpolation conditions:

$$E(Rp_{0,i} + c) \equiv E(q_i) = p_{1,i} \equiv y_i \quad , \quad i = 1, \ldots, N, \tag{13}$$

where $q_i = (q_i^1, q_i^2, q_i^3)^T$, and $y_i = (y_i^1, y_i^2, y_i^3)^T$. This interpolation problem is always solvable if we use an augmented radial approximation (with a properly chosen g) of the form

$$E(q) = \sum_{i=1}^{N} a_i g\left(\|q - q_i\|\right) + Aq + \alpha_4, \tag{14}$$

where $A = (\alpha_1, \alpha_2, \alpha_3)^T$, $a_i \in I\!R^3$, $1 \le i \le N$, $\alpha_\ell = (\alpha_\ell^1, \alpha_\ell^2, \alpha_\ell^3)^T \in I\!R^3$, $1 \le \ell \le 4$.

Thus E is determined by $N+4$ coefficients in $\mathbb{R}^3$. The computation of those coefficients involves the solution of three square linear systems of size $N+4$ each, where N conditions are derived by the interpolation requirements, as defined in (13), and the additional compatibility conditions are

$$\sum_{i=1}^{N} a_i^k = \sum_{i=1}^{N} a_i^k q_i^1 = \sum_{i=1}^{N} a_i^k q_i^2 = \sum_{i=1}^{N} a_i^k q_i^3 = 0 \ , \ k = 1, 2, 3 \ .$$

These conditions guarantee that the transformation is affine reducible, i.e., the transformation is purely affine whenever possible. The system of equations for the vectors of unknowns $u_k = (a_1^k, ..., a_N^k)^T$ and $v_k = (\alpha_1^k, \alpha_2^k, \alpha_3^k, \alpha_4^k)^T$, k=1,2,3, is

$$\begin{cases} Gu_k + Hv_k = b_k \\ H^T u_k = 0 \end{cases} , \tag{15}$$

where $b_k = (y_1^k, ..., y_N^k)^T$, $G = \{ \ g(\|q_i - q_j\|) \ \}_{i,j=1}^N$ and H is an $N \times 4$ matrix with an i-th row $\{q_i^1, q_i^2, q_i^3 \ , 1\}$, $1 \leq i \leq N$.

Now that the warp transformation W_t is fully defined, the warped DFI approximation can be obtained by applying the definition of Eq. (4). In the following section we describe the application of the method in a discrete voxel-based environment.

5 The Signed 3D Distance Transform

Given a geometric object, its signed 3D distance field is generated by first rasterizing the object into a binary discrete volumetric representation. The binary volume, which represents the solid object, consists of binary-valued voxels, which are classified either as feature or non-feature voxels. Feature voxels are defined as those which either intersect the surface of the object or are contained within it. Assume now that the feature voxels are contained in a discrete bounding box partitioned into voxels at a certain resolution.

Given a binary 3D discrete volume, the 3D distance transformation converts the volume into a distance field as defined in Eq. (2). We extend the 2D discrete distance transform [5] to 3D. The basic idea is that the global operation of calculating the distances can be approximated by propagating local distances. These methods are most efficient, and are easily extended to 3D.

5	4	5
4	3	4
5	4	5

4	3	4
3	0	3
4	3	4

5	4	5
4	3	4
5	4	5

Figure 7: *The 3 layers of the cubic Chamfer distance mask.*

The basic process of computing the distances of the non-feature voxels to the nearest feature voxel, is done as follows. The original volume is initialized, such that feature voxels are assigned a zero value and non-feature voxels are assigned infinity. We use a discrete distance called Chamfer distance [5] which approximates the Euclidean distance. It uses a $3 \times 3 \times 3$ cubic mask (illustrated in Figure 7), where the local distances are scaled to avoid floating point arithmetic.

The distance transform process consists of two successive scannings of the volume, a forward scan and a backward scan. The 3D Chamfer distance mask is partitioned into two half masks. The first half, denoted Forward Mask (FM), contains the distances of the voxels that are positioned in the mask "before" the central voxel in the scanning direction. The second half, denoted Backward Mask (BM), contains the other voxels in the mask. Both halves contain the central voxel.

At each step the mask is placed over a voxel $v_0 = (x_0, y_0, z_0)$ and its value, $V(v_0)$, is replaced by the following computation:

Forward Scan: top to bottom, back to front, left to right ,

$$V(v_0) = \min_{(i,j,k) \in FM} \{V(x_0 + i, y_0 + j, z_0 + k) + d(i,j,k)\} \ .$$

Backward Scan: bottom to top, front to back, right to left ,

$$V(v_0) = \min_{(i,j,k) \in BM} \{V(x_0 + i, y_0 + j, z_0 + k) + d(i,j,k)\} \ .$$

Here (i,j,k) is the position in the mask (the center being $(0,0,0)$), and $d(i,j,k)$ is the local distance from the mask center. The process described assigns to each non-feature voxel the distance to its nearest feature voxel, while the feature voxels are assigned the value zero.

However, for our application, this basic computation of the distance transform needs to be extended to evaluate the distances of the feature voxels as well as the non-feature voxels, as defined by Eq. (2). To match this definition, we switch the roles of the feature and the non-feature voxels, and repeat the distance transform to compute a second distance field volume, containing the distances of the feature voxels to the nearest non-feature voxel. All the positive distance values in the second distance field are negated, and the two discrete distance fields are combined (by discarding the zero-valued distances in both volumes), to yield the representation of Eq. (2).

The source and the target objects to be morphed are represented as discrete Distance Field volumes (DF-volumes). Constructing the intermediate object essentially requires generating its discrete DF-volume, out of which its surface can be extracted [23].

6 The Discrete Morphing Procedure

In this section we present the morphing procedure used for generating the intermediate DF-volume in a voxel-based representation. The formal definition of the warped DFI method is given in Eq. (4). However, the implementation of this formula in a discrete environment is not straightforward, and involves an additional discretization approximation. For clarity, let us consider the procedure for the case of two 3D objects, $S = \Omega_0$ and $T = \Omega_1$.

As explained in Section 2, the morphing procedure combines two simultaneous processes: warping and cross-dissolving. The warp mapping is determined by the mapping of the anchor points, while the cross-dissolving is performed by interpolating the distance values of corresponding voxels (see Figure 8). We combine two backward mappings, as described below with respect to Figure 9, in order to perform this process.

Let D_0 and D_1 be the two DF-volumes of the source and the target objects, respectively, and let us denote the DF-volume of the desired intermediate object D_t . We assume that all these DF-volumes and the anchor points are contained within a finite set X of $\mathbb{R}^3$.

Constructing the intermediate DF-volume D_t requires the evaluation of the correct distance value to be associated with each of the voxels at level t. The first step of this process is to apply the warp transformation W_t, as defined in Eq. (11), to the source anchor points, in order to find the location of the anchor points in the intermediate level. These points are denoted

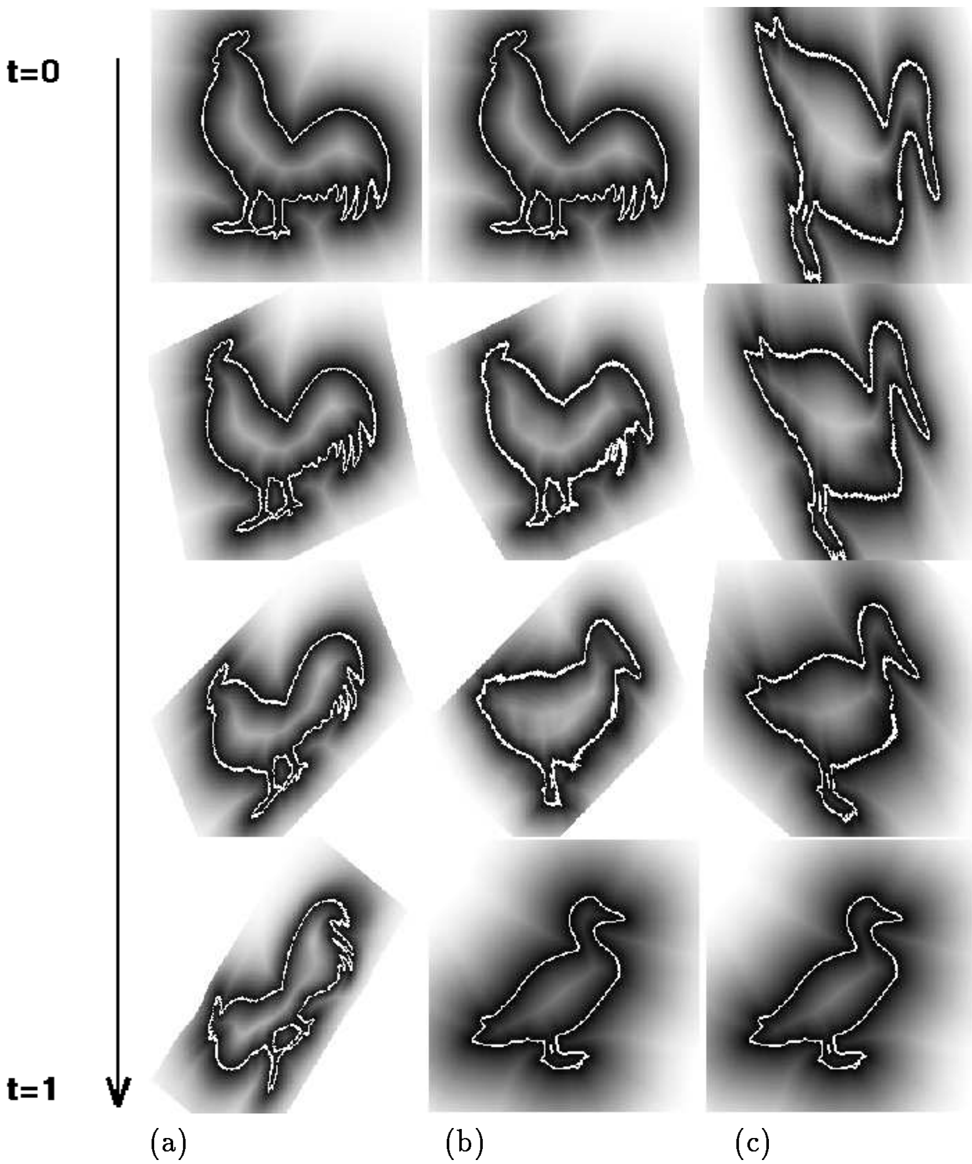


Figure 8: *Warp-Guided DFI. (a) The source Distance Field warped towards the target. (c) The target Distance Field warped towards the source. (b) The intermediate interpolated Distance Field. The contours are highlighted in white.*

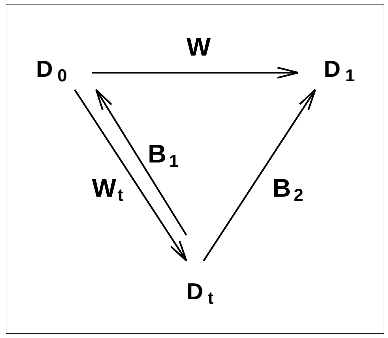


Figure 9: *The transformations involved in the generation of D_t. W_t maps the anchor points and B_1 and B_1 backmap the distance values.*

$\{m_i \equiv W_t(p_{0,i})\}_{i=1}^N$. In the next step, for any voxel at level t we would like to find the corresponding voxels at levels 0 and 1. This is done using the two backward mappings B_1 and B_2, such that B_1 , $B_2 : X \to \mathbb{R}^3$, and

$$B_1(m_i) = p_{0,i} \ , \quad B_2(m_i) = p_{1,i} \ , \quad i = 1, \ldots, N. \tag{16}$$

It is important to note here that while B_1 is the inverse of W_t at the anchor points, this does *not* necessarily hold for other points. Finding B_1 and B_2 is done by the process described in the previous section.

Once B_1 and B_2 are available, D_t is generated as follows: For each $v \in X$, the distance value $D_t(v)$, is evaluated by

$$D_t(v) = (1-t)D_0(B_1(v)) + tD_1(B_2(v)), \tag{17}$$

where the discrete distance functions D_0 and D_1 are extended to X by trilinear interpolation. Hence, the two distance values of the corresponding voxels at levels 0 and 1 are interpolated linearly. For some voxels $v \in X$, it may occur that either $B_1(v)$ or $B_2(v)$ is not in X, in which case we assign to this point an approximated distance value, by adding the distance from this point to the distance value of its nearest voxel in X. In order to speed up the computation, the trilinear interpolation can be applied only to distance values which are close to zero, since this is where the surface is located, and avoided elsewhere.

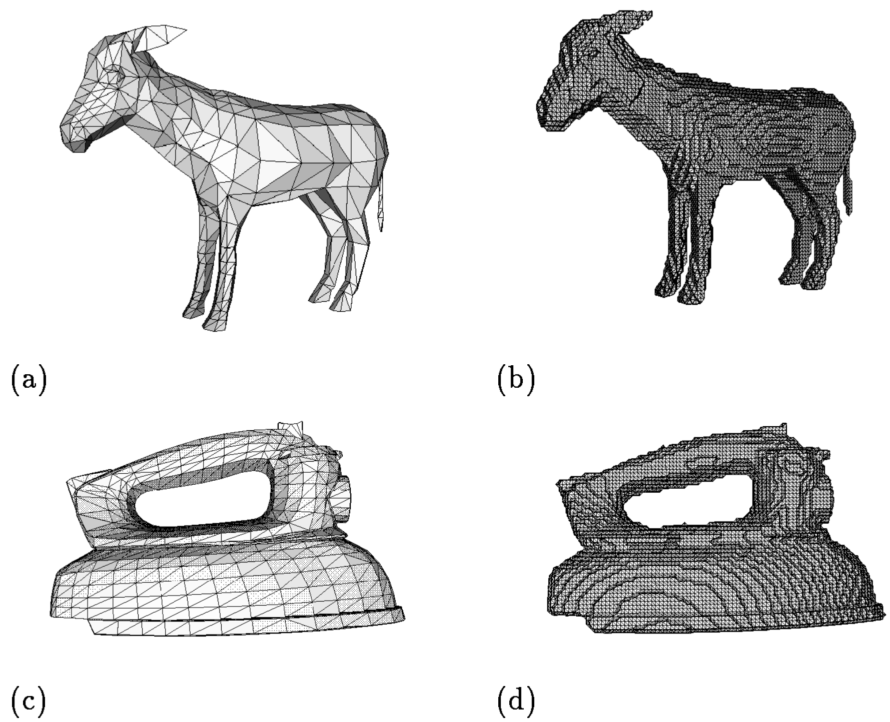


Figure 10: *Surface reconstruction results. (a) The original Donkey (874 triangles). (b) The model voxelized at a resolution of* 100^3 *(26764 triangles) . (c) The original Iron (2600 triangles). (d) The model voxelized at a resolution of* 100^3 *(42482 triangles) .*

7 Results

We have tested the algorithm on several 3D examples, as shown in Figures 1, 12 and 13. Two choices of radial basis functions for computing the elastic warp were examined. The Gaussian function $g(r) = exp(-r^2/\sigma^2)$ is used in Figures 12 and 13, while the $g(r) = r^3$ basis function is used for the morphing example given in Figure 1. For more details on the choice of the radial functions, see [1]. For a comparison of run-times using radial basis functions transformations, see [26].

Figure 10 shows two results of the voxelization and the surface reconstruction of 3D models. Note that the large number of triangles in the reconstructed surfaces depends directly on the resolution used, regardless of the complexity of the original object representation, and as the resolution grows the main problem becomes the storage space consumption. However, a large number of redundant polygons in regions of low curvature can be removed by decimation and simplification processes (e.g., [27]). Once the voxelized DF-volume is computed, the time cost for creating an intermediate volume is a function of the resolution, the number of anchor points and the choice of the radial basis functions.

We have implemented the 3D algorithm on an SGI R4400 machine. Using an unoptimized C code, it took approximately 40 minutes to create an intermediate 200^3 volume, with 20 anchor points and global supported radial functions. All the models seen in the examples were originally represented as polyhedra, and the voxelization was performed in a 200^3 resolution. The RayShade ray-tracer was used to render the images.

Figure 1 shows a 3D morphing sequence between the surfaces of a cow and a tiger. We used $g(r) = r^3$ as the radial function, and controlled the morphing with 18 anchor points, located on the legs, head, tail and along the sides of the animals.

The morphing in Figure 12, between the surfaces of a Triceratops and an iron, is controlled by 16 anchor points using the Gaussian as the radial function with $3\sigma = 20$. Note that only the iron object has a hole.

Figure 13 shows a 3D morphing sequence between two key objects that have a different number of connectivity components. The source is composed of two objects, a mushroom and an iron, while the target is a teapot. The morphing is controlled using 11 anchor points, located on the handle of the teapot and the iron, on the spout and along the sides of the teapot. We used the Gaussian as the radial function with $3\sigma = 20$. It should be emphasized that the relative positions of these three models is identical to that shown in the sequence, so that the rotation is performed by the rigid transformation.

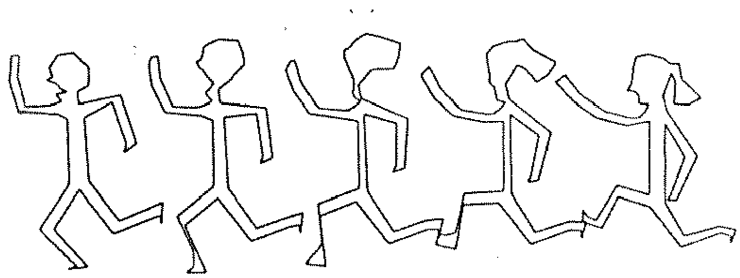
We visually compared the results of our algorithm with the results of Sederberg et al.[29], Shapira and Rappoport [31] and Goldstein and Gotsman [12] , on the same input. (The first three results were scanned from [12].) The results are given in Figure 11, where the anchor points used to create our morphing are marked on the key figures. The thin-plate spline function $g(r) = r^2 \log r$ has been used as the radial function. It should be pointed out, that the first three methods treat the objects as polygons and are thus forced to create a proper correspondence between their geometric features (e.g., vertices), whereas our method treats the objects as (discretized) contours. That means that in our technique the representation of the two objects may differ and need not be restricted to polygons, since no explicit correspondence between the representation's features needs to be established.

8 Conclusions

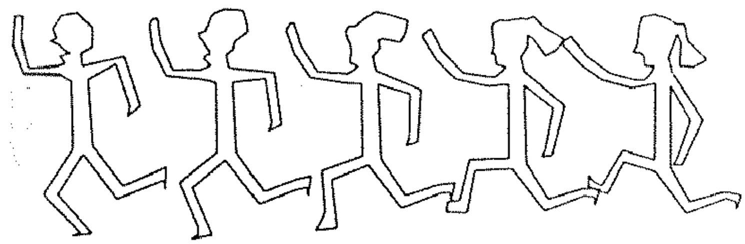
The paper presents an object-space metamorphosis method between two (or more) surfaces of solid objects with general topology, using an extended distance field interpolation. The metamorphosis of the surface is guided by corresponding control points which define a warp function. The warp function is decomposed into a rigid part and an elastic part in order to reduce the distortion of intermediate models. The rigid transformation serves as an approximation to the overall warp transformation, which yields better control over the path of the in-between shape throughout the metamorphosis.

The method presented here allows an intuitive and simple manual intervention of the animator who can control the animation process by prescribing a small number of anchor point. The advantage of this approach is that no correspondence between the models' geometric primitives (e.g., vertices) needs to be established. Moreover, the input objects do not necessarily needs to have the same representation or the same topological genus, and can have a different number of connectivity components (see [9]).

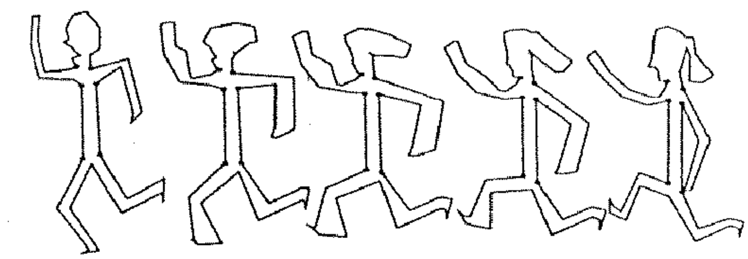
Although there is no explicit definition of what the in-between objects



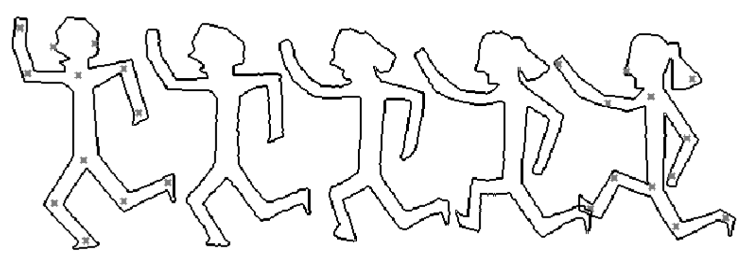
(a)



(b)



(c)



(d)

Figure 11: *Comparative results of morphing algorithms. (a) Sederberg et al. (b) Shapira and Rappoport. (c) Goldstein and Gotsman. (d) DF Morphing. The anchor points are marked on the two key figures.*

should look like, the sequence of shapes should look "natural" to the viewer. There is a variety of ideas of what the sequence of in-between objects should satisfy to appear "natural". However, it is rather easy to agree that a morph sequence should satisfy the following criteria to yield a pleasing morph.

- The volume and the boundary surface area of the objects should change smoothly and monotonically.
- The boundary surface of the objects should retain the smoothness of the original objects.
- Features common to both source and target objects (e.g., head or legs), should be preserved during the process.

Generally speaking, these criteria aim at treating the objects as rigidly as possible and at avoiding redundant global and local deformations. A morph sequence satisfying the above criteria probably yields a good metamorphosis, but it still remains difficult to obtain an objective comparison between different sequences, resulting either from different algorithms or from the same algorithm with different parameters.

However, the quality of the morph is very much subject to a proper warp. That is, it works well provided that "corresponding parts" of the two objects are properly aligned by the warp. Otherwise, some parts may unexpectedly disappear and reappear, exactly as can happen in pure DFI (with no warp). The fact that the warp is guided by the user, is probably the weak part of the technique. However, a high fidelity, eye-pleasing, automatic alignment of arbitrary objects is still a major challenge.

Acknowledgments

The authors would like to thank Pixel Inc., especially Haggai Goldfarb and Boaz Tal, for providing us with the 3D models. Special thanks to Yael Milo for her moral support and encouragement. Oded Cohen and Eran Rosenberg helped us to produce the video movies. Gershon Elber and Nur Arad provided us with many helpful comments on early drafts.

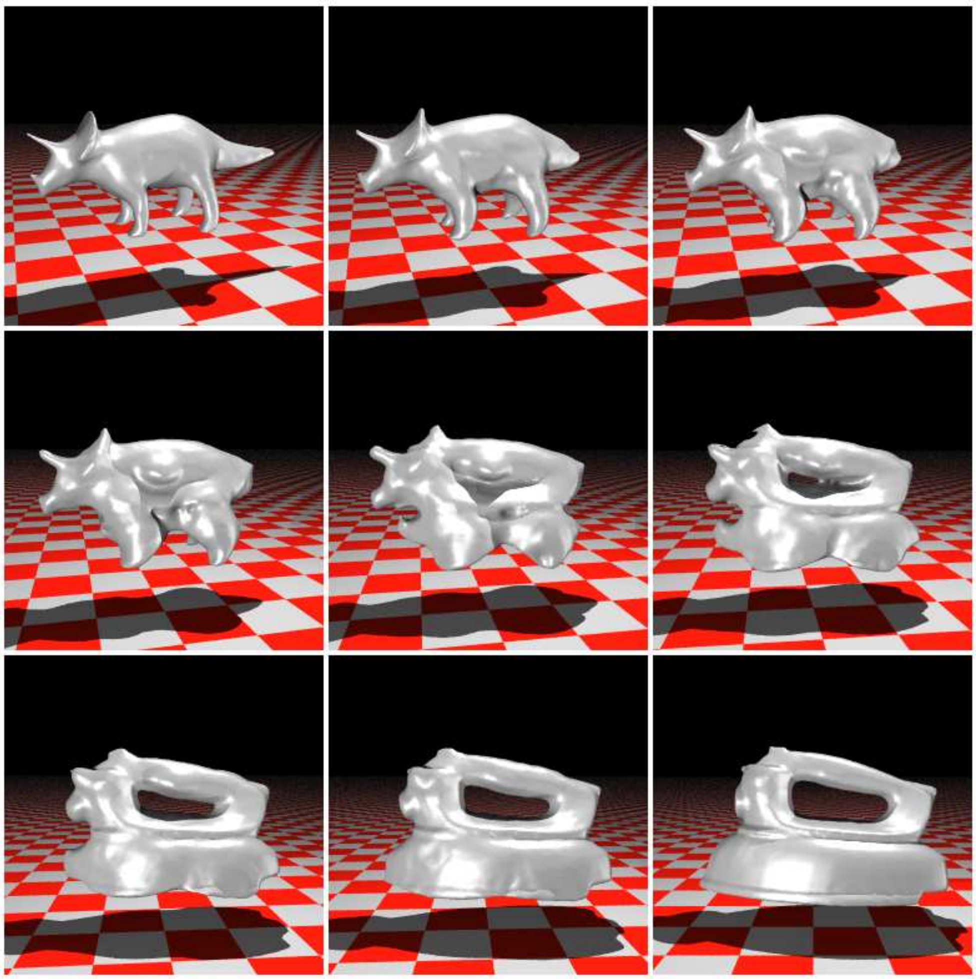


Figure 12: *3D morphing between a Triceratops and an iron. Note that only the iron object has a hole.*

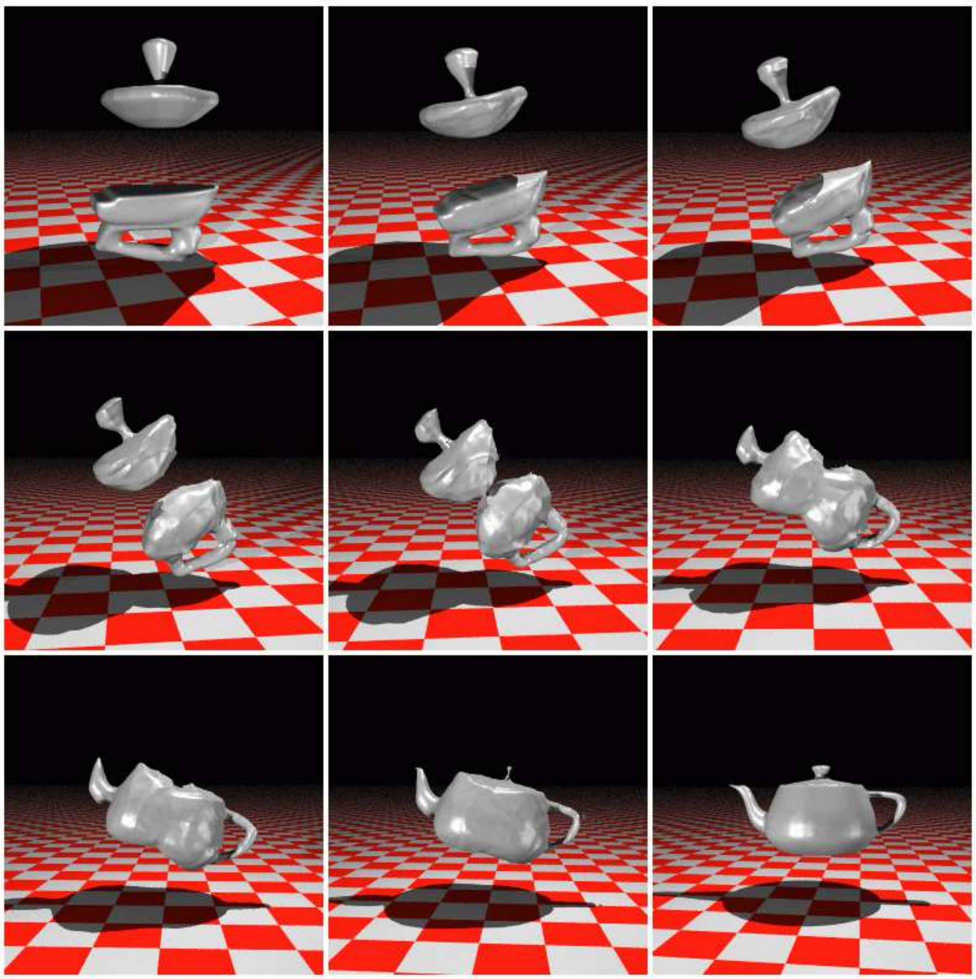


Figure 13: *3D morphing between a mushroom, an iron and a teapot. Note that the source and the target objects have a different topological genus.*

References

[1] N. Arad and D. Reisfeld. Image warping using few anchor points and radial functions. *Computer Graphics Forum*, 14(1):35–46, 1995.

[2] K.S. Arun, T.S. Huang, and S.D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, September 1987.

[3] G. Barequet and M. Sharir. Piecewise linear interpolation between polygonal slices. *Proc. 10th Ann. ACM Symp. on Computational Geometry, Stony Brook, NY*, pages 93–102, 1994. Full version to appear in: *Computer Vision, Graphics and Image Processing: Image Understanding*, vol. 63 (1), January 1996.

[4] T. Beier and S. Neely. Feature-based image metamorphosis. In *Proceedings of SIGGRAPH '92*, volume 26(2), pages 35–42. ACM, June 1992.

[5] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371, June 1986.

[6] E. Carmel and D. Cohen-Or. Warp-guided object-space morphing. *The Visual Computer*, to appear in 1997.

[7] E. Chen and R. Parent. Shape averaging and its applications to industrial design. *IEEE Computer Graphics and Applications*, 9(1):47–54, January 1989.

[8] D. Cohen-Or, D. Levin, and A. Solomovici. Contour blending using warp-guided distance field interpolation. In R. Yagel and G.M. Neilson, editors, *Proceedings of Visualization '96*, pages 165–172. IEEE Computer Society, October 1996.

[9] D. DeCarlo and J. Gallier. Topological evolution of surfaces. In *Graphics Interface'96*, pages 194–203, 1996.

[10] P. Decaudin. Geometric deformation by merging a 3d-object with a simple shape. In *Graphics Interface'96*, pages 55–60, 1996.

[11] N. Dyn. Interpolation and approximation by radial and related functions. In L.L. Schumaker C.K. Chui and J.D. Ward, editors, *Approximation Theory VI*, pages 211–234, 1987.

[12] E. Goldstein and C. Gotsman. Polygon morphing using a multiresolution representation. In *Graphics Interface '95*, pages 247–254, 1995.

[13] T. He, S.Wang, and A. Kaufman. Wavelet-based volume morphing. In D. Bergeron and A. Kaufman, editors, *Proceedings of Visualization '94*, pages 85–91. IEEE Computer Society and ACM SIGGRAPH, October 1994.

[14] G.T. Herman, J. Zheng, and C.A. Bucholtz. Shape-based interpolation. *IEEE Computer Graphics and Applications*, 12:69–79, May 1992.

[15] J.F. Hughes. Scheduled Fourier volume morphing. *Computer Graphics*, 26(2):43–46, July 1992.

[16] T. Kanai, H. Suzuki, and F. Kimura. 3d geometric metamorphosis based on harmonic maps. In *Proceedings of Pacific Graphics '97*, pages 97–104, October 1997.

[17] A. Kaul and J. Rossignac. Solid-interpolation deformations: Construction and animation of pips. In *Proceedings of EUROGRAPHICS'91*, pages 493–505, September 1991.

[18] J.R. Kent, W.E. Carlson, and R.E. Parent. Shape transformation for polyhedral objects. *Computer Graphics*, 26(2):47–54, July 1992.

[19] F. Lazarus and A. Verroust. Feature-based shape transformation for polyhedral objects. In *Proceedings of the Fifth Eurographics Workshop on Animation and Simulation*, 1994.

[20] S.Y. Lee, K.Y. Chwa, S.Y. Shin, and G. Wolberg. Image metamorphosis using snakes and free form deformations. In *Proceedings of SIGGRAPH '95*, pages 439–448. ACM, August 1995.

[21] A. Lerios, C. D. Garfinkle, and M. Levoy. Feature-based volume metamorphosis. In *Proceedings of SIGGRAPH '95*, pages 449–456. ACM, August 1995.

[22] D. Levin. Multidimensional reconstruction by set-valued approximation. *IMA J.Numerical Analysis*, 6:173–184, 1986.

[23] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of SIGGRAPH '87*, volume 21(4), pages 163–169. ACM, August 1987.

[24] R.E.. Parent. Shape transformation by boundary representation interpolation: a recursive approach to establishing face correspondences. *The Journal of Visualization and Computer Animation*, 3:219–239, 1992.

[25] B.A. Payne and A.W. Toga. Distance field manipulation of surface models. *IEEE Computer Graphics and Applications*, 12(1):65–71, January 1992.

[26] D. Ruprecht and H. Muller. Image warping with scattered data interpolation. *IEEE Computer Graphics and Applications*, pages 37–43, March 1995.

[27] W. Schroeder, J. Zarge, and W. Lorensen. Decimation of triangle meshes. *Computer Graphics*, 26(2):65–70, July 1992.

[28] T. Sederberg and S. Parry. Free-form deformations of solid geometric models. In *Proceedings of SIGGRAPH '86*, volume 20(4), pages 151–160. ACM, August 1986.

[29] T.W. Sederberg, P. Gao, G. Wang, and H. Mu. 2-D shape blending: An intrinsic solution to the vertex path problem. In *Proceedings of SIGGRAPH '93*, volume 27, pages 15–18. ACM, August 1993.

[30] T.W. Sederberg and E. Greenwood. A physically based approach to 2d shape blending. In *Proceedings of SIGGRAPH '92*, volume 26(2), pages 25–34. ACM, June 1992.

[31] M. Shapira and A. Rappoport. Shape blending using the star-skeleton representation. *IEEE Computer Graphics and Applications*, 15:44–50, March 1995.

[32] K. Shoemake. Animating rotation with Quaternion curves. In *Proceedings of SIGGRAPH '85*, volume 19(3), pages 245–254. ACM, July 1985.

[33] Y.M. Sun, W. Wang, and F.Y.L. Chin. Interpolating polyhedral models using intrinsic shape parameters. In S.Y. Shin and T.L. Kunii, editors, *Proceedings of the Third Pacific Conference on Computer Graphics and Applications, Pacific Graphics '95*, pages 133–147. World Scientific, 1995.

[34] T.J. True and J.F. Hughes. Volume warping. In *Proceedings of Visualization '92*, pages 308–315. IEEE Computer Society and ACM SIGGRAPH, 1992.

[35] G. Wolberg. *Digital Image Warping.* IEEE Computer Society Press, Washington, DC, 1990.