

Implicit modeling by metamorphosis of 2D shapes

Július Parulek*

Faculty of Mathematics, Physics and Informatics
Comenius University
Mlynská Dolina, Bratislava, Slovakia

Miloš Šrámek†

Austrian Academy of Sciences
Donau-City-Strasse 1
A-1220 Vienna, Austria

Abstract

Nowadays, there is a lack of existing practical modeling tools suitable for specification of free-form implicit shapes. In this paper, we propose a new interpolation technique supplemented by interactive tools aimed at shape modeling by feature preserving metamorphosis of two-dimensional implicit shapes. The metamorphosis is controlled by global parameters and a set of correspondence vectors that are drawn directly to the scene. The underlying environment is the XISL package for definition and manipulation of implicit objects, which offers additional modeling and rendering possibilities.

CR Categories: I.3.5 [COMPUTER GRAPHICS]: Computational Geometry and Object Modeling—Curve, surface, solid and object representations;

Keywords: implicit surfaces, metamorphosis, geometrical modeling

1 Introduction

Recent progress in computer graphics has put much effort in exploring shape transformation techniques. Considering initial and destination geometric shapes, the problem is to find a suitable method that creates continuous transformation (metamorphosis) between them. Metamorphoses can be thought as temporal (animations) and spatial.

Several approaches were proposed for creation of a 2D metamorphosis of polygonal shapes [Sederberg and Greenwood 1992; Shapira and Rappoport 1995; Cohen-Or et al. 1996]. A detailed survey on 3D metamorphosis can be found in [Lazarus and Verroust 1998]. However these techniques assume certain preconditions: equivalent topology, polygonal shape representation, shape alignment and overlap. The techniques of metamorphosis of implicit shapes were addressed in several works and can be classified in two main categories.

First, methods of metamorphosis were proposed for the so-called soft-objects [Wyvill and Wyvill 1986] which are build on a set of skeletal primitives. Here, a weighting function is attached to each primitive. In the process of surface in-betweening the weighting function, in case of the source object, progressively increases from zero to one and, in case of the destination object, decreases from one to zero. The metamorphosis of soft-objects was studied later by Galin and Akkouche [Galin and Akkouche 1996]. Their approach was based on creation of a bijective graph of correspondence between carrier skeletons and then, using Minkowski sum, distance and field functions of intermediate shapes were characterized. Techniques based on soft objects together with CSG operations were generalized in the advanced modeling concept, which was elaborated as the BlobTree system [Wyvill et al. 1999]. Metamorphosis of the BlobTree was studied by Galin et al. [Galin et al. 2000]. The whole transformation is characterized by a generic BlobTree that provides a correspondence between nodes of the

input trees. Its instantiations interpolate the initial and the final shapes. The specification of the tree correspondence provides a fine control of the final metamorphosis.

The second class of metamorphosis methods is based on functional representation of objects [Pasko et al. 1995]. In function representation defining functions have no special limitations. It combines numerous different models like skeleton based implicit surfaces, set-theoretical solids, sweeps, volumetric objects, parametric and procedural models. *Savchenko and Pasko* proposed general framework of metamorphosis based on extended space mapping [Savchenko and Pasko 1998] (discussed later in Section 2). Further, bounded blending techniques were applied for metamorphosis of implicit shapes [Pasko et al. 2002]. The authors use an additional bounding implicit solid in which the metamorphosis takes place. As the bounding solid the authors propose to use intersection of half-spaces placed in parallel near the locations of original implicit shapes. The blending material is then added only within this intersection and creates a smooth transition between the given shapes. *Pasko et al.* also propose to apply Minkowski sums to implement metamorphosis [Pasko et al. 2003]. First, the Minkowski sum is created for two real valued functions representing the initial shapes. Second, using Minkowski sum operator, the metamorphosis is achieved by interpolating of coordinates of input shapes. *Schmitt et al.* introduced controlled metamorphosis between two implicit shapes [Schmitt et al. 2005]. Their approach is based on the creation of non-overlapping space partitions, which control metamorphosis by specific time schedule. The time schedule and special global and local dynamic variables control the behavior of metamorphosis in each partition.

In this paper we propose a new object modeling technique by spatial metamorphosis between a set of 2D implicit shapes which are organized in a system of parallel planes. In the paper remainder, the term *implicit shape* stands for a 2D implicit shape. The main motivation is to create tools, which provide for developing of an arbitrary metamorphosis interactively using only a small amount of parameters. Note, the aim is to create a 3D object; i.e. not an animation. Nevertheless, the 2D animation can be also thought as a result. In this case, the central transformation axis can be considered as a time parameter.

The remainder of the paper is organized as follows: In Subsection 1.1 we introduce the XISL modeling tool. The Section 2 describes modeling techniques based on extended space mapping and a basic method for metamorphosis of objects based on functional representation. The theoretical background of our method is introduced in Section 3. The Sections 4 and 5 involve description and demonstration of our modeling environment. Application of the presented method to current research is described in Section 6. The future work is introduced in Section 7 and we conclude in Section 8.

1.1 XISL tool

In our work, we define implicit models using the XISL language (XML based scripting of Implicit Surfaces [Parulek et al. 2006a]) and tools, which assist developers in construction of arbitrary im-

*e-mail: julius.parulek@savba.sk

†e-mail: milos.sramek@oeaw.ac.at

PLICIT models. These models are described in declarative text files by means of the extensible markup language (XML). Each implicit function class (a primitive, an operation, etc.) is defined by its appropriate XISL tag(s). This ensures the requirement of clear notation of complex implicits. The XISL implicits are defined by the inequality

$$f(\mathbf{x}) \geq 0, \quad (1)$$

where $\mathbf{x} = (x_1, x_2, x_3) \in E^3$. This function is also called functional representation of objects [Pasko et al. 1995] and provides for implementation of various forms of implicits.

In XISL, each implicit object is represented via an n -ary hierarchical tree, leafs of which stand for arbitrary implicit primitives and inner nodes stand for unary, set-theoretic, blending and interpolation operations.

The whole XISL package is written in C++, includes the individual classes of implicit surfaces and provides high-level C++ API for parsing of XISL tags, voxelization and polygonization. Several similar modeling systems based on implicits were developed [Adzhiev et al. 1999; Wyvill et al. 1999], nevertheless, XISL is a compact package, which provides for extensibility and works well on both Windows and Linux systems.

2 Related work

Savchenko and Pasko presented a mathematical framework of shape transformations by the form of extended space mappings [Savchenko and Pasko 1998]. When a given function f is defined in n -dimensional space E^n , the space E^{n+1} is then called an extended space and can be constructed by adding one more dimension. Consider $P_n = (x_1, \dots, x_n)$ is a point in E^n , so-called hypersurface S is defined by the function $f(x_1, \dots, x_n) = c$ and $P_{n+1} = (x_1, \dots, x_n, c)$ is a point in E^{n+1} . The geometric object G is then defined as

$$G = \{P_n : P_{n+1} \in S, c \geq 0\}. \quad (2)$$

The authors presented various methods of transformations of geometric objects defined by the extended space mappings $\Phi : E^{n+1} \rightarrow E^{n+1}$ applied to hypersurfaces S . In general, the mapping $\Phi(P_{n+1}) = \Phi(P_n, c) = (P'_n, c')$ is composed of two parts ϕ_1 and ϕ_2

$$\Phi(P_{n+1}) = \Phi(P_n, c) = (\phi_1, \phi_2) = (P'_n, c'), \quad (3)$$

where

$$\begin{aligned} \phi_1(P_n, c) &= P'_n \\ \phi_2(P_n, c) &= c' \\ f(P_n) &= c \end{aligned} \quad (4)$$

The Eqs. 4 can be rewritten in the compact form

$$c' = \phi_2(\phi_1^{-1}(P'_n), f(\phi_1^{-1}(P'_n))). \quad (5)$$

According to the possible selection of ϕ_1 and ϕ_2 , authors classified mappings in three main categories. First, function mappings, which are defined as

$$c' = \phi_2(P'_n, f(P'_n)) \quad (6)$$

and incorporate the operations of offsetting, metamorphosis, etc.; i.e. operations that affect only function values. Second, geometric coordinate space mappings are defined by

$$c' = f(\phi_1^{-1}(P'_n)). \quad (7)$$

These mappings take into account only modification of coordinates, which can be used for various deformations. Third, combined mappings use both functions ϕ_1 and ϕ_2 , therefore its general definition is in Eq. 5.

The following Eq. 8 represents an example of the combined form of extended space mapping

$$c' = f(g(P'_n)) - o, \quad (8)$$

where the shape defined by the function f is deformed by the twist mapping $g(\cdot)$ and the function offset is changed by the scalar o (Fig. 1). According to [Savchenko and Pasko 1998], the combined

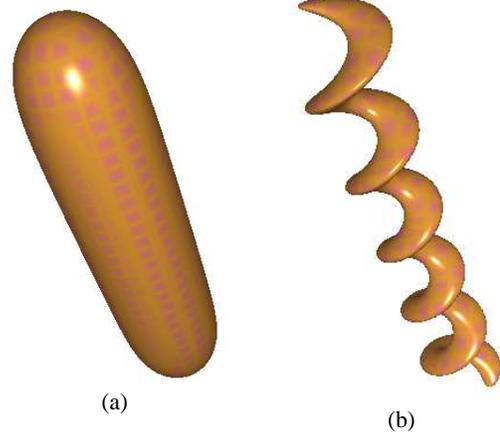


Figure 1: (a) The original object. (b) The object is deformed by the combined mapping composed of twisting and offsetting operations.

mapping was used for metamorphosis between two input shapes, which also provides for controlling of feature correspondences between the input shapes.

2.1 Linear interpolation

Consider two 2D implicit shapes, with the corresponding defining functions f_1 and f_2 , located in parallel planes at z_1 and z_2 z -positions. In functional representation the basic method for defining smooth transition between the two given real valued functions f_1 and f_2 is linear interpolation (Fig. 2). It can be viewed as function mapping class, where the resultant function is composed of the functions f_1 and f_2 :

$$c' = \phi_2(\mathbf{x}, f_1(\mathbf{x}), f_2(\mathbf{x})) = (1-t)f_1(\mathbf{x}) + tf_2(\mathbf{x}), \quad (9)$$

where $\mathbf{x} = (x, y)$ is a point in Euclidean space E^2 and t is a parameter of interpolation estimated from a current z -position, e.g. defined as $t = (z - z_1) / (z_2 - z_1)$. Nevertheless, utilization of the Eq. 9 assumes that the shapes defined by f_1 and f_2 overlap. For instance, by applying translation to the shape defined by f_1 , which shifts f_1 outwards of overlapping, the resultant metamorphosis produces gaps (Fig. 3).

To ensure gap-free metamorphosis also for the case of non-overlapping shapes, we proposed a new extended interpolation method supplemented by an interactive editing application, which together enable to define the required smooth transformation.

3 Extended interpolation

Our method modifies the approach proposed in [Savchenko and Pasko 1998], which was based on the combined space mapping approach. We will explain the new approach on a model defined by transformation between two implicit shapes with their defining functions f_1 and f_2 . Assume that each shape contains

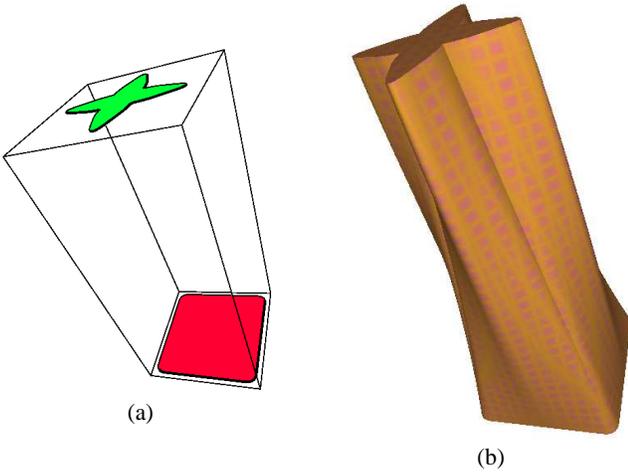


Figure 2: (a) Given two 2D overlapping implicit shapes placed in parallel. (b) A metamorphosis can be defined as a linear interpolation between their defining functions.

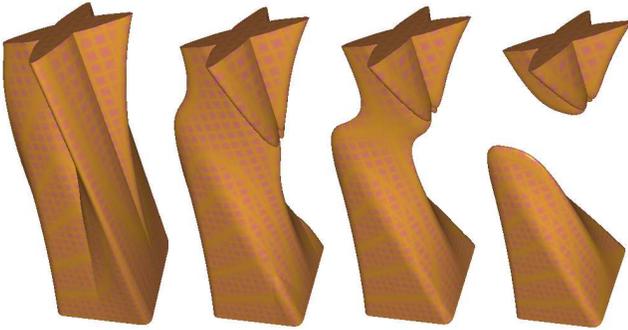


Figure 3: Incremental translation applied to the shape defined by the function f_1 . A metamorphosis, defined as linear interpolation, in the case of non-overlapping shapes creates the gap.

a set of control points, $P = \{p_1, \dots, p_n\}$ for the function f_1 and $Q = \{q_1, \dots, q_n\}$ for the function f_2 . These points are interactively defined by a user, which enables him/her to specify the desired shape. Moreover, vectors of correspondence are established between these points, $C_P = \{q_1 - p_1, \dots, q_n - p_n\}$ and $C_Q = \{p_1 - q_1, \dots, p_n - q_n\}$. The set C_P is attached to the set P , and the set C_Q is attached to the set Q . Now, using geometric space mapping notation, we create two weighting displacement functions ϕ^P and ϕ^Q , which represent the transformations of the given shapes in the directions defined by the vectors C_P and C_Q . The weighting displacement functions are defined by

$$\begin{aligned} \mathbf{x}' &= \phi^P(\mathbf{x}) = \mathbf{x} + h_1(t)d_1(\mathbf{x}) \\ \mathbf{x}' &= \phi^Q(\mathbf{x}) = \mathbf{x} + h_2(t)d_2(\mathbf{x}), \end{aligned} \quad (10)$$

where $h_1(t)$, $h_2(t)$ represent weighting proportions within the interval $\langle 0, 1 \rangle$, and $d_1(x)$, $d_2(x)$ represent interpolation of control points given by directions C_P and C_Q . To interpolate the displacement d_1, d_2 we adopt volume splines—the so-called thin-plate function [Duchon 1977; Savchenko and Pasko 1998]. The advantage is that this spline has the minimum bending energy from all functions which interpolate scattered data. In this case, the interpo-

lation function $d_1(\mathbf{x})$ is defined by

$$d_1(\mathbf{x}) = \sum_{i=1}^n \lambda_i g(\mathbf{x}, p_i) + \lambda_{n+1} + \lambda_{n+2}x + \lambda_{n+3}y, \quad (11)$$

where λ_i are spline coefficients and $g(\mathbf{x}, p_i) = |\mathbf{x} - p_i|^2 \ln(|\mathbf{x} - p_i|)$. To estimate the spline coefficients λ_i , we expand Eq. 11 to a $(n+3) \times (n+3)$ matrix

$$\begin{bmatrix} g_{ij} & P \\ P^T & M \end{bmatrix} \begin{bmatrix} \Lambda \\ \Lambda \end{bmatrix} = \begin{bmatrix} C \\ N \end{bmatrix}, \quad (12)$$

where g_{ij} is $n \times n$ matrix, $g_{ij} = g(p_i, p_j)$, for $0 \leq i, j \leq n$, M is 4×4 zero matrix, $\Lambda = [\lambda_1, \dots, \lambda_n, \lambda_{n+1}, \lambda_{n+2}, \lambda_{n+3}]^T$, $C = [q_1 - p_1, \dots, q_n - p_n]^T$, $N = [0, 0, 0, 0]^T$ and

$$P = \begin{bmatrix} p_1^x & p_1^y & 1 \\ p_2^x & p_2^y & 1 \\ \vdots & \vdots & \vdots \\ p_n^x & p_n^y & 1 \end{bmatrix}.$$

The system is solved by the symmetric LU decomposition. The function $d_2(\mathbf{x})$ is estimated in the same manner, but instead of the sets P and C_P , the sets Q and C_Q are used.

To create smooth transformation, the linear interpolation (Eq. 9) is modified by the displacement functions (Eqs. 10). Now, the resultant combined mapping, originally proposed in [Savchenko and Pasko 1998], would be defined as

$$c' = w_1(t)f_1(\phi^P(\mathbf{x})) + w_2(t)f_2(\mathbf{x}), \quad (13)$$

where t is the parameter of interpolation and the weighting functions w_1, w_2 define the resultant metamorphosis.

In our work, in order to provide displacement in both directions, we extend the Eq. 13 by the second displacement function ϕ^Q from the Eqs. 10. The mapping is then defined by

$$c' = w_1(t)f_1(\phi^P(\mathbf{x})) + w_2(t)f_2(\phi^Q(\mathbf{x})). \quad (14)$$

The advantage is that the control points are moved in both directions, which provides better better preservation of features (Fig. 4).

However, using Eq. 14 gaps can still appear. Therefore, to fill the possible gaps, we extend this formula by an additional blending term w_3 :

$$c' = w_1(t)f_1(\phi^P(\mathbf{x})) + w_2(t)f_2(\phi^Q(\mathbf{x})) + aw_3(t), \quad (15)$$

where the parameter a represents the amount of the blending material and the weighting function $w_3(t)$ controls its distribution in the interpolation direction.

3.1 Weighting functions

The weighting functions h_1, h_2, w_1, w_2 and w_3 from Eqs. 10 and 15 can be specified by an operator in an interactive process by observing of the resultant transformation.

In our case, h_1 and h_2 , i.e. functions that control the displacement of control points, are defined as

$$h_1(t) = (1 - t^a)^b \quad (16)$$

$$h_2(t) = 1 - h_1(t), \quad (17)$$

where the parameters a, b modify the slope and curvature of the transition (Fig. 5). In order to interpolate the given shapes, the both

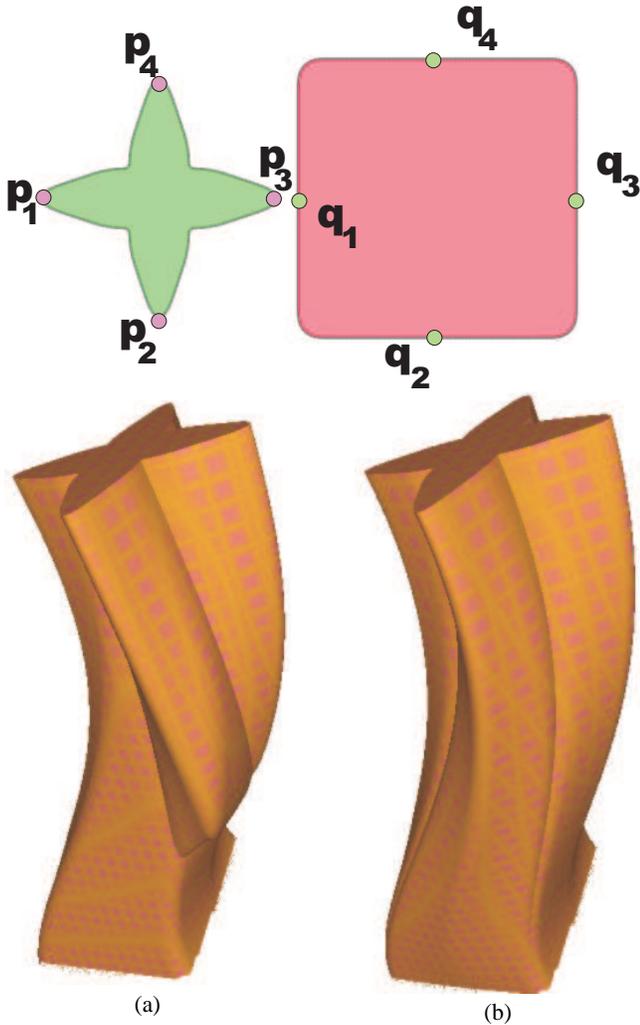


Figure 4: Top: A given set of control points for both input shapes. (a) If using Eq. 13, the top shape features vanish in the middle part of the metamorphose. (b) Using our method (Eq. 14), these features are preserved better.

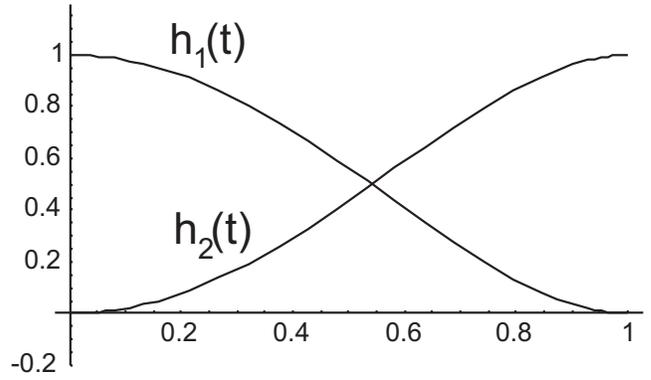


Figure 5: Weighting functions h_1 and h_2 .

functions h_1 and h_2 equal zero at the initial shape's surface positions.

The weighting functions w_1 and w_2 affect the final transformation. For simplicity, these functions are defined as linear interpolation $w_1(t) = (1-t)$ and $w_2(t) = t$.

With respect to our experience, the gaps can originate mainly in the middle of the transformation and therefore we define the weighting function $w_3(t)$ by

$$w_3(t) = (1 - (2t - 1)^c)^d, \quad (18)$$

where the parameters c , d modify the slope and the curvature (Fig. 6). Note that function w_3 equals zero at initial positions of shapes again and therefore it does not corrupt the metamorphosis.

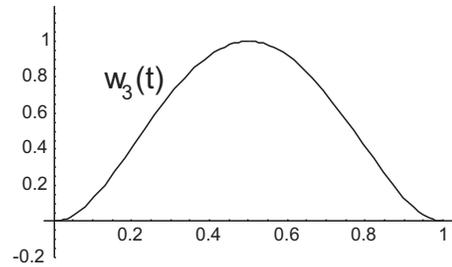


Figure 6: The weighting function w_3 has the maximum in the middle.

4 Interactive modeling

We developed a graphical application which enables to load arbitrary two-dimensional implicit shapes defined in the XISL language and, using the aforementioned extended interpolation, enables to create the required three-dimensional smooth interpolation of the given shapes. Similarly to F-rep, XISL shapes can be defined in numerous ways. For instance, the presented shapes were created by the method of implicit curved polygons [Pasko et al. 1996].

The incorporated shapes are given in parallel planes (Fig. 7). In this case the shapes do not overlap, and therefore, using the simple linear interpolation (Eq. 9), the resultant 3D implicit object is not interconnected (Fig. 8). Using simple interaction, we draw several line segments, which represent the correspondence vectors between the input shapes (Fig. 9). Now, by the means of the extended

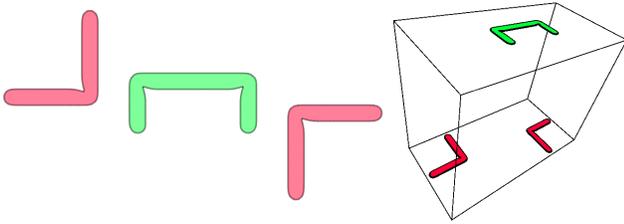


Figure 7: Left: The incorporated shapes that belong to the same plane are drawn in the same color. Right: 3D view of the shape positions.

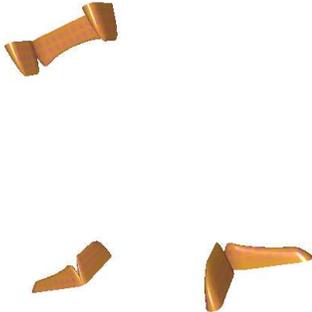


Figure 8: The resultant transformation obtained by the means of linear interpolation does not provide for smooth transition.

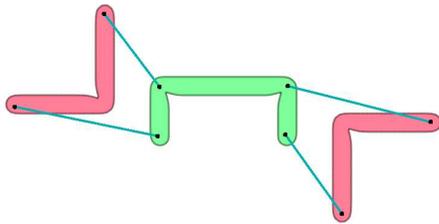


Figure 9: Vectors of correspondence are specified by the operator by direct drawing of line segments.

interpolation, the resultant function will smoothly interconnect the given shapes (Fig. 10). However, we sometimes still observe gaps or cracks in the final model. Therefore, by means of increasing the blending parameter a from Eq. 15 these cracks can be filled. The consequence of increasing this parameter is depicted in Fig. 11a. Nevertheless, the drastically increasing of this parameter can deform the resultant shape and brings unwanted bulges (Fig. 11b).

5 Examples

Let us now consider initial shapes as those depicted in Fig. 12. Specification of the correspondence vectors and of the blending term not only enables us to get rid of the unwanted cracks, but also enables to specify the final shape according to the demands of the user. The interactive application provides for drawing line segments easily and, moreover, enables to inspect the model by means of surface polygonization and rendering and to store the resultant object in a file in the XISL language (Fig. 13). Here, this approach is ex-



Figure 10: Interpolation by means of the vectors of correspondence specified in Fig. 9 using Eq. 14. Note that the undesirable cracks are still visible.

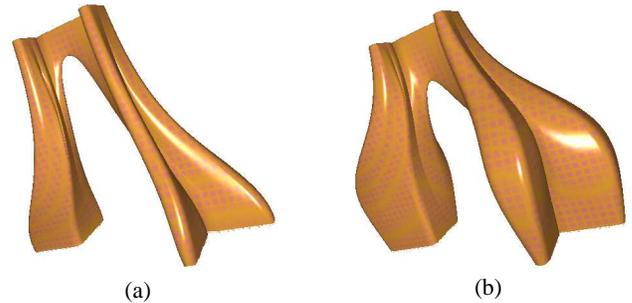


Figure 11: Increasing the blending parameter a (Eq. 15) is used to fill the cracks (a), nevertheless very high values of a may result in unwanted bulges (b).

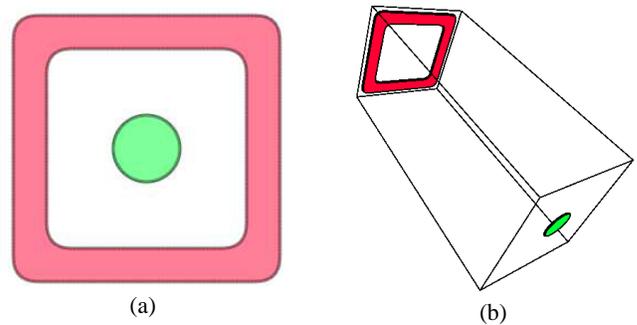


Figure 12: The initial implicit shapes.

emplified by creation of several configurations of correspondence, which lead to different shape of the final object (Fig. 14). We can observe high sensitivity of the resultant transformation shape to the given configuration of correspondences. Additional modification of the resultant shapes can be achieved by adjusting the parameters a and b in the weighting functions h_1 and h_2 . Moreover, to present the flexibility of implicit representation, we apply the twist operation on the resultant transformations (Fig. 15). An additional useful feature resides in the possibility to interpolate through any number of shape levels. Of course, the user has to draw the required correspondence vectors between each pair of levels (Fig. 16). However, creation of smooth transitions between each pair of levels is still a problem.

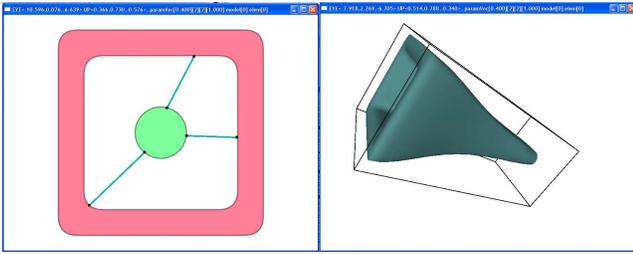


Figure 13: Application screen shots demonstrate creation of vectors of correspondence and polygonized model preview.

6 Application to current research

The presented technique was used for geometrical modeling of muscle cells [Parulek et al. 2006b]. In order to create the complex system of interconnected tubes of one of the basic muscle cell structures, sarcoplasmic reticulum, we applied the transformation method to a set of scattered spherical implicit shapes. However, the number of appropriate vectors of correspondence would be in this case too high for interactive editing. Therefore, we proposed an automatic method residing in that two sets of control points P and Q are created on both input shapes. With respect to the fact that all included shapes are spherical (convex and without holes), these points correspond to centers of gravity for each shape. Note, these sets can contain different number of control points; i.e. $|P| \neq |Q|$. The second step represents the process of establishing correspondence vectors C_P and C_Q between the P and Q sets. In this step, the main problem is to find a way for finding suitable correspondences. Currently, we simplify the solution by finding for each point $p \in P$ the nearest control point $q \in Q$. However, before that, we compute a bounding box enclosing the contours in each plane and transform coordinates of points to the interval $\langle 0, 1 \rangle$ within the bounding box (Fig. 17). Now, for each point in P exists the vector $q - p$. The same method is performed on the set Q . All sets are then processed by the proposed extended interpolation method, bringing acceptable results (Fig. 18).

This method is suitable not only for modeling purposes, but also for reconstruction of biological objects from contour data sets. The problem can be stated as finding suitable implicit surface approximation from contours. Several works dealing with this problem [Savchenko et al. 1995; Turk and O'Brien 2002; Akkouche and Galin 2004] were published. Geometric algorithms that directly address the creation of triangulated surface interpolating the contours face various problems; e.g. correspondence, tiling, branching, etc. Techniques based on implicit surfaces tackle these problems. However, in case of complex and non-overlapping shapes, even implicit representation does not solve these problems and additional information has to be included; e.g. by means of the correspondence vectors introduced in this paper.

7 Future work

In the case of a large number of shapes it would be quite time consuming to interactively draw the desired line segments representing the correspondence vectors. Therefore, we have already started to move towards an automation since, specially in a system of simple shapes as is the case of muscle cells, it is possible to define the basic correspondence in an automatic way. However, in the case of complex shapes this topic requires further studies. One possibility resides in utilization of the space time blending method that does not need specification of correspondences [Pasko et al. 2004]. More-

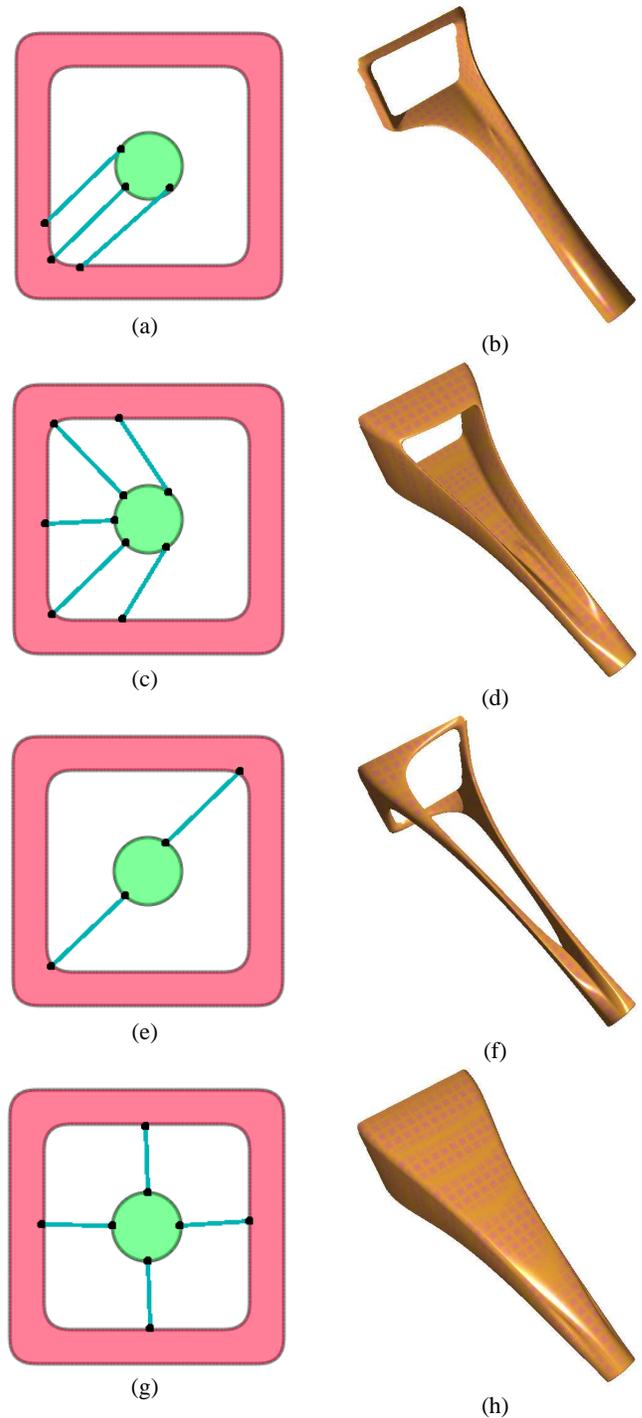


Figure 14: The left column represents possible configurations that represent the vectors of correspondence. The corresponding resultant models are depicted on the right.

over, the bounded blending technique, proposed by the same authors, can be used to define the blending term in (15), which would provide for limited influence of the blending material.

It is also possible to use the method of correspondence vectors as an animation tool. For each vector of correspondence, which represents the initial vector, a user can add a destination vector. The in-

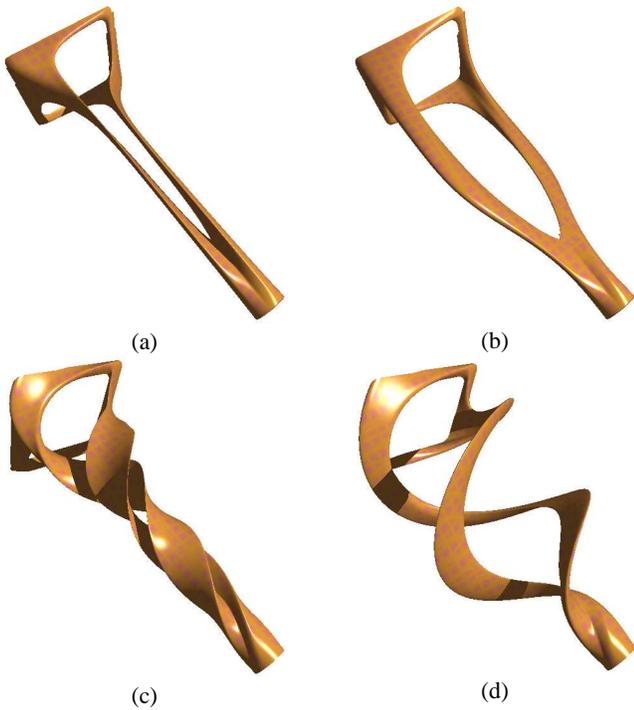


Figure 15: (a) An example of the varying weighting functions, where we set $a = 2$, $b = 12$. (b) The parameters are the same as in (a), but the functions h_1 and h_2 were swapped. (c),(d) Addition operation of twist applied to (a),(b).

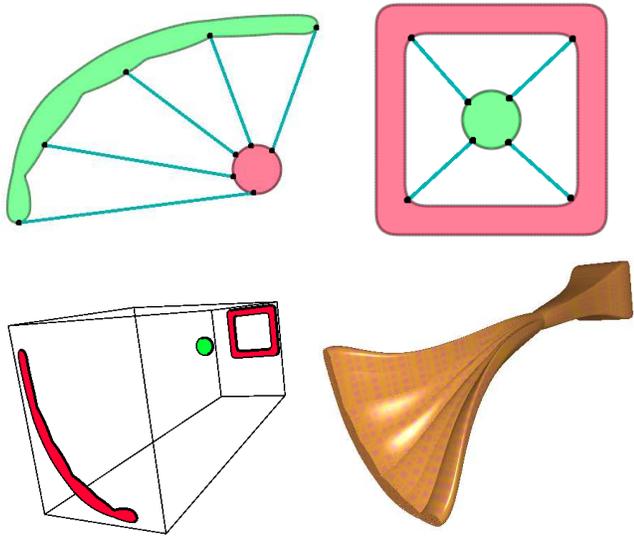


Figure 16: The technique provides for creation of metamorphoses through any number of levels.

betweening process based on these vectors produces an animation, however, pre-computation has to be done for each vector location. A speed up can be achieved by computation of displacement functions by interpolation between displacement functions in the source and the destination phase.

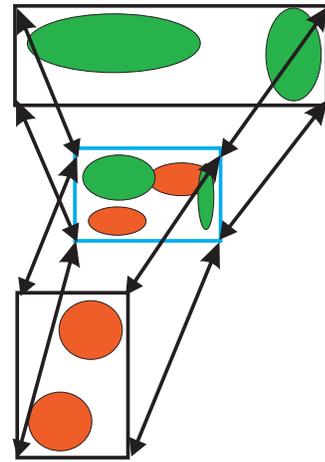


Figure 17: The coordinates of both input shapes are clamped to the unit bounding box.

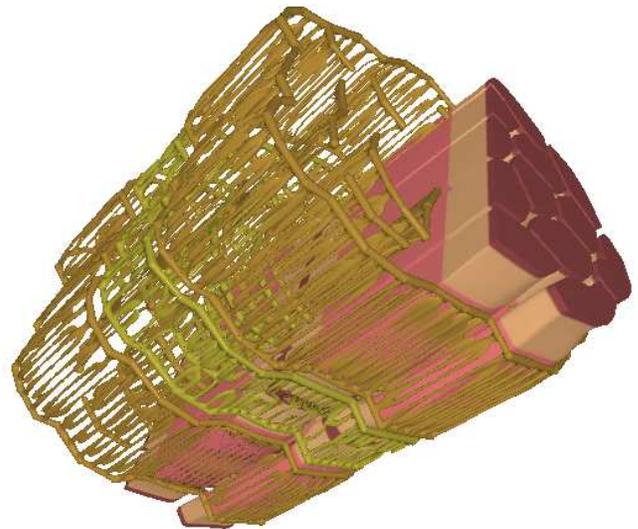


Figure 18: A section of a muscle cell model showing sarcoplasmic reticulum (thin structures) and myofibrils (thick structures). The system of tubes that represents the reticulum is built from the set of implicit shapes distributed in parallel planes.

Currently, the weighting functions are limited only to those, which already exist in the XISL system. However, a simple editor could make possible creation of custom weighting functions as, for example, spline curves.

The other improvement could be achieved by attaching correspondence curves to the control points, instead of straight correspondence vectors.

8 Conclusion

We proposed a method and a tool for object modeling by smooth space metamorphosis between sets of arbitrary two-dimensional implicit shapes. The method is based on the extended space mapping and enables to represent smooth metamorphosis between both

overlapping and non-overlapping shapes. The space-time blending approach proposed in [Pasko et al. 2004] deals with the same problem, however, their method is primarily oriented to development of animations, which do not necessarily require smooth metamorphoses. On the contrary, the resultant 3D model produced by our method should be smooth everywhere, and our method suits well for these purposes. The presented graphical tool enables to specify the final shape interactively by sketching vectors of correspondence, to preview polygonized model surfaces and to store the result by means of the XISL language, which can be used for further modeling and high-quality rendering¹.

9 Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-20-056105.

References

- ADZHIEV, V., CARTWRIGHT, R., FAUSETT, E., OSSIPOV, A., PASKO, A., AND SAVCHENKO, V. 1999. Hyperfun project: A framework for collaborative multidimensional f-rep modeling. In *Proc. of the Implicit Surfaces '99 EUROGRAPHICS/ACM SIGGRAPH Workshop*, 59–69.
- AKKOUICHE, S., AND GALIN, E. 2004. Implicit surface reconstruction from contours. *Vis. Comput.* 20, 6, 392–401.
- COHEN-OR, D., LEVIN, D., AND SOLOMOVICI, A. 1996. Contour blending using warp-guided distance field interpolation. In *VIS '96: Proceedings of the 7th conference on Visualization '96*, IEEE Computer Society Press, Los Alamitos, CA, USA, 165–ff.
- DUCHON, J. 1977. *Splines minimizing rotation-invariant seminorms in Sobolev spaces*. Springer-Verlag, ed. Walter Schempp and Karl Zeller, Berlin-Heidelberg.
- GALIN, E., AND AKKOUICHE, S. 1996. Blob metamorphosis based on minkowski sums. *Comput. Graph. Forum* 15, 3, 143–154.
- GALIN, E., LECLERCQ, A., AND AKKOUICHE, S. 2000. Morphing the blobtree. *Comput. Graph. Forum* 19, 4, 257–270.
- LAZARUS, F., AND VERRON, A. 1998. 3d metamorphosis: a survey. vol. 14, 373–389.
- PARULEK, J., NOVOTNÝ, P., AND ŠRÁMEK, M. 2006. XISL—a development tool for construction of implicit surfaces. In *SCCG '06: Proceedings of the 22nd spring conference on Computer graphics*, Comenius University, Bratislava, 128–135.
- PARULEK, J., ŠRÁMEK, M., NOVOTOVÁ, M., AND ZAHRADNÍK, I. 2006. Computer modeling of muscle cells: Generation of complex muscle cell ultra-structure. *Imaging & Microscopy* 8, 2 (June), 58–59.
- PASKO, A. A., ADZHIEV, V., SOURIN, A., AND SAVCHENKO, V. V. 1995. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer* 11, 8, 429–446.
- PASKO, A., SAVCHENKO, A., AND SAVCHENKO, V. 1996. Polygon-to-function conversion for sweeping. In *The Eurographics/SIGGRAPH Workshop on Implicit Surfaces*, J.Hart, K. van Overveld (Eds.), 163–171.
- PASKO, G., PASKO, A., IKEDA, M., AND KUNII, T. 2002. Bounded blending operations. In *SMI '02: Proceedings of the Shape Modeling International 2002 (SMI'02)*, IEEE Computer Society, 95.
- PASKO, A., OKUNEV, O., AND SAVCHENKO, V. 2003. Minkowski sums of point sets defined by inequalities. *Computers and Mathematics with Applications* 45, 10/11, 1479–1487.
- PASKO, G., PASKO, A., AND KUNII, T. 2004. Space-time blending: Research articles. *Comput. Animat. Virtual Worlds* 15, 2, 109–121.
- SAVCHENKO, V., AND PASKO, A. 1998. Transformation of functionally defined shapes by extended space mappings. *The Visual Computer* 14, 5-6, 257–270.
- SAVCHENKO, V. V., PASKO, A. A., OKUNEV, O. G., AND KUNII, T. L. 1995. Function representation of solids reconstructed from scattered surface points and contours. *Computer Graphics Forum* 14, 4, 181–188.
- SCHMITT, B., PASKO, A., AND FAYOLLE, P.-A. 2005. Local metamorphosis of functionally defined shapes. In *GRAPHITE '05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM Press, New York, NY, USA, 355–361.
- SEDERBERG, T. W., AND GREENWOOD, E. 1992. A physically based approach to 2-D shape blending. E. E. Catmull, Ed., vol. 26, 25–34.
- SHAPIRA, M., AND RAPPOPORT, A. 1995. Shape blending using the star-skeleton representation. *IEEE Comput. Graph. Appl.* 15, 2, 44–50.
- TURK, G., AND O'BRIEN, J. F. 2002. Modelling with implicit surfaces that interpolate. *ACM Trans. Graph.* 21, 4, 855–873.
- WYVILL, B., AND WYVILL, G. 1986. Using soft objects in computer generated animation.
- WYVILL, B., GUY, A., AND GALIN, E. 1999. Extending the csg tree (warping, blending and boolean operations in an implicit surface modeling system). *Computer Graphics Forum* 18, 2, 149–158.

¹<http://www.sccg.sk/~parulek/xisl/>