Blob-based Liquid Morphing

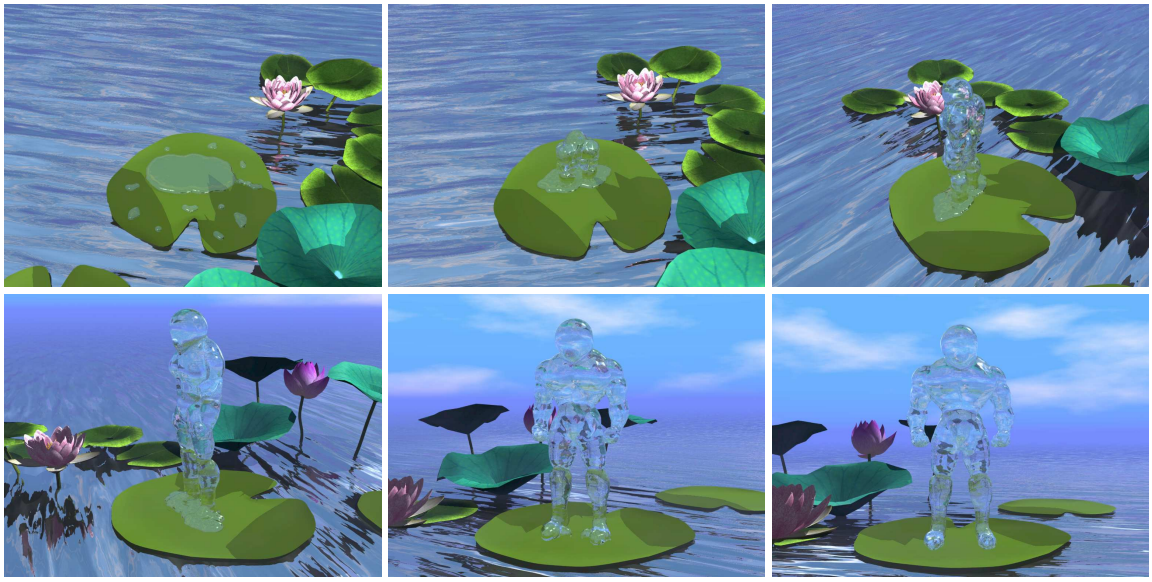

Figure 1: The six frames of liquid robot morphing.

Abstract

In this paper, we propose a novel practical method for blob-based liquid 3D morphing. Firstly, blobby objects are employed to approximate a given polygonal surface through an energy optimization procedure so that the distance between the isosurface of blobs and the given model is minimized. The primitives in the medial axis sphere-tree of a polygonal model are utilized as initial blobs — this greatly improves the robustness and efficiency of the blob-based approximation. Secondly, we establish the blob correspondences between two models by sphere cellular matching and hierarchial matching. Finally, we interpolate the parameters of the implicit representation to get the intermediate shapes. Experiments show our method can produce visually pleasing liquid morphing effects.

Keywords: 3D morphing, blobby object, medial axis sphere-tree, cellular matching, hierarchial matching

1 Introduction

The metamorphosis with liquid effect is frequently used in computer animation and entertainment industry. In some of the recent movies, there has been a liquid robot emerging from a pool of water. We would like to develop techniques for digitally reproducing similar effects. Two problems need to be addressed: 1) to find a model representation suitable for liquid morphing, 2) the morphing algorithm.

Implicit surfaces are widely used in computer graphics applications including geometric modeling, three-dimensional metamorphosis and collision detection. An implicit surface S is usually defined by a continuous scalar function $f(x)$ with $x \in \mathcal{R}^3$. The geometry of S is given by the locus of points at which the function $f(x) = 0$. An important class of implicit surfaces is the so-called blobby model [1] and its variants — metaball [2] and soft object [3]. The implicit functions of these surfaces are the sum of radial symmetric functions that have a Gaussian profile. Their general form is like

$$f(x) = -t + \sum_{i=1}^{n} \omega_i f_i(x). \qquad (1)$$

In this formula, the parameter t is the threshold of isosurface S, n is the number of blobby primitives, ω_i is the weight for the ith primitive with default value 1.0, and every single function f_i describes the profile of a blobby sphere with a particular center and radius. Due to the smooth blending property of implicit surfaces, 3D morphing can be easily performed between two given implicit surfaces with any topological structures. The blobby model is a natural representation for simulating liquid. We therefore use blobby implicit surface to represent the models under liquid morphing.

For simple objects (e.g., spheres and peanut-like objects etc), it is easy to obtain their corresponding blobby representations. But for a model with complex shape such as a human body, it is a hard and tedious work to construct blobs for the model manually. Therefore, automatic approaches are sought. Muraki [4] is the first to generate blobby objects automatically for shape reconstruction. His method incrementally adds primitives one at a time. Because this requires solving an optimization problem when selecting every primitive, the computational speed is quite slow. Tsingos et al. [5] presented a semi-automatic method for the shape reconstruction with implicit surfaces. Its efficiency is improved by the usage of a selection criterion based on "local skeleton energy". Bittar et al. [6] proposed an automatic approach for implicit surface reconstruction by calculating a medial axis of the volume data. However, their method works well only for the implicit surface with a small number of primitives. This is not enough for modelling objects with complex geometry. On the other hand, in the approach of [6], users must select proper medial axis resolution to generate primitives — inappropriate decision will lead to poor results. But selecting a good medial axis resolution is by no means an easy job, especially for novices.

There are many other works about surface reconstruction using implicit surfaces. Such as adaptively sampled distance fields [7], variational implicit surfaces [8, 9], multi-level partition of unity implicit surfaces [10], interpolating and approximating implicit surfaces using a moving least-squares formulation with constraints [11]. These methods all can get good result, however, they are not appropriate for 3D morphing with liquid effect.

The proposed method in this paper is a novel practical solution for automatically approximating polygonal meshes with blobby objects. First of all, a medial axis sphere-tree of the given object is constructed. Then, primitives in the sphere-tree are adopted as initial guesses of blobby objects, which are further refined through an energy optimization. Since the sphere-tree has already completely covered the surface S of a given object M, the isosurface defined by blobs converges rapidly to S in optimization. Compared to other earlier techniques, the approach presented here is more robust and efficient. Objects with complex topology and geometry can be automatically approximated by blobby objects in our scheme. This greatly benefits the morphing applications.

Metamorphosis (or morphing) is the process of smoothly transforming a source shape into a target shape. Implicit surface based morphing techniques directly compute the metamorphosis by interpolating the parameters of implicit functions, where the isosurface of the interpolated implicit function represents the intermediate shape in morphing. Wyvill [12, 13] first addressed the implicit smooth blending method. In [14], Pasko avoided the correspondence process by directly interpolating the implicit functions of source and target shapes. But this technique lacks control over morphing process. Galin and Akkouche [15] discussed a morphing method using Minkowski sum of skeletons. In [16] and [17], the feature correspondence issue between objects being morphed is discussed, where distance-fields are utilized. By treating the morphing process as a higher dimensional interpolation problem, Turk and O'Brien [18] presented a concise morphing approach. Treece et al. [19] proposed an algorithm for volume-based three-dimensional metamorphosis using region correspondence, and the region correspondence is guided by sphere-correspondence vectors between source and target models. Using dynamic level-set and particles, Foster et al. [20] and Enright et al. [21] simulated a realistic water effect. However, these physics-based methods are not appropriate for morphing between two models. What we need is a special 3D morphing effect that is a smooth transformation with liquid appearance.

A new 3D metamorphosis technique is developed in this paper based on blobby model representation. Our morphing algorithm borrows some idea from [22], but there are two major differences from [22]. One is that we adopt optimized blobs instead of spheres to interpolate and get blobs of the intermediate shape. Another is that we just consider one sphere and its nearest sphere instead of minimizing the sum of "distances" between two sphere sets to establish the primitive correspondence. Thus, more accurate shape can be generated during metamorphosis. Also, since our blobby objects are determined from a medial axis sphere-tree, the hierarchical information helps us to speed up the process of searching correspondences.

The major contributions of this article include an overall framework for 3D morphing with liquid effect, an automatic scheme for implicit surface reconstruction based on blobby objects, a new field function designed for blobby objects optimization, and a novel morphing algorithm based on blobby representation.

In the following section, the technology of using blobby objects to approximate polygonal meshes is presented in detail. After that, we describe the morphing algorithm with fluid effect. At the end of this paper, several results of surface approximation and 3D metamorphosis are demonstrated to illustrate the functionality of our approach.

2 Approximating polygonal meshes

We propose a new algorithm for approximating given polygonal meshes by blobby objects. First of all, points are sampled on the polygonal mesh M so that a medial axis sphere-tree Ψ can be generated from the sample points, which are called *forming points*. Based on the forming points, spheres are incrementally added into Ψ to increase the approximation accuracy. The sphere-tree Ψ is presented in a hierarchical structure, where the spheres at the most detail level are further reformed to blobs whose isosurface gives a better approximation of M. The parameters of blobs are determined through an energy optimization. An illustration of the major steps in our blobby approximation approach is shown in Figure 2.

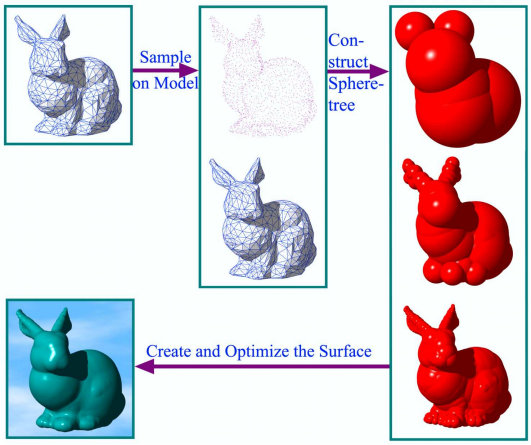

Figure 2: The process of blobby surface reconstruction.

2.1 Sphere-tree construction

The *medial axis* of an object represents its skeleton and can be defined as the centers of a set of maximally sized spheres that fill the object. It can be intuitively considered as the set of points that are the farthest inside the object, forming a skeleton that gives information on the object's topology and geometry. Thus, these points are very good candidates for becoming skeletons of blobby objects whose isosurface approximates the object's shape.

Medial axis sphere-tree is a hierarchical structure of spheres whose centers locate on the medial axis. In our method, the blobby objects approximating a given polygonal mesh are generated from the spheres in a medial axis sphere-tree of the mesh. We call these spheres as *medial-axis spheres*. A number of algorithms have been developed for the construction of medial axis sphere-trees. The sphere-tree construction algorithm employed here conducts an iteration scheme to update the medial axis approximation adaptively, tightly based on [23].

The surface of a given polygonal mesh M is represented by an arbitrarily large set P of sample points. The medial axis sphere-tree is constructed on these sample points. When computing the medial-axis spheres, the surface of M is considered to be completely covered if the full set of P are covered by spheres in Ψ. To reflect this requirement, the root of sphere-tree is a sphere s_0 enclosing all points in P. The sphere s_0 will be subdivided into a number of smaller

spheres to get a more accurate approximation. Therefore, an upper bound on the approximation error of a medial-axis sphere is need. The following measurement is used [23],

$$e = r - \|q - c\|, \tag{2}$$

where e is the approximation error, r is the radius of the sphere, c is the center of the sphere, and $q \in P$ is the surface point closest to c. Based on Eq. (2), the forming point q could be selected and introduced into the Voronoi diagram to generate more medial-axis spheres.

The hierarchy of a medial axis sphere-tree is completed through the process of adaptive medial axis update and sphere reduction. When generating the spheres in level L of Ψ from the spheres in level $(L-1)$, the Voronoi diagram in level $(L-1)$ is firstly updated by adding new forming points so that the set of spheres contains ten times the sphere number of the target. Then, the number of spheres is reduced by an expending algorithm [23]. The spheres in level L segments the sampling points held by spheres in level $(L-1)$ so that the object is divided into sub-regions with as little overlap as possible. Each region that defines the areas of the object must be covered by a set of children spheres. These spheres form the next level of hierarchy. Figure 3 shows an example of the hierarchical structure.

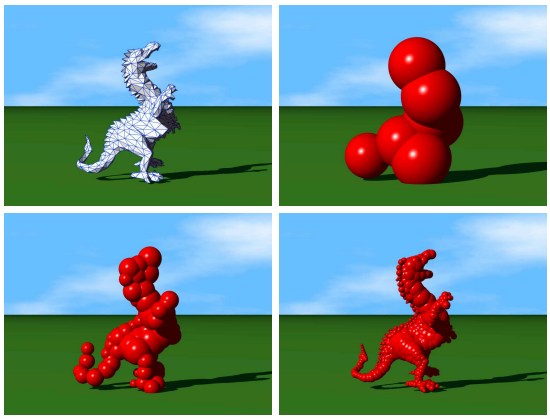

Figure 3: A dragon model and the three levels of its medial axis sphere-tree.

2.2 Blobby fitting

Through constructing the medial axis sphere-tree, we obtain the sphere representation of a given object M. The blobby model is created from the spheres by taking centers of them as skeletons with associated field function, whose isosurface approximates the surface of M. The parameters of blobby objects are then optimized to achieve a better approximation.

2.2.1 Definition of field functions

Field functions employed in a blobby model can be classified into global and local ones. The global field functions such as Gaussian functions utilized in [1, 4] become zero at infinite. But if global field functions were used, the computational cost will be quite expensive when there are many primitives included, and their fields will overlap each other globally. These factors limit the number of blobs that could be involved. Therefore, we choose local polynomial field functions in our blobby model. Local field functions (e.g., the one introduced in [2, 3]) decrease to zero at the distance of influence radius R. Based on the local influence region, a fast computing speed can be achieved. Besides, local field functions offer local controls on their defined implicit surfaces, which is extremely important for using implicit surfaces to approximate a given mesh surface. Due to this local property, when adjusting the parameters of one blob, the surface region defined by other blobs will not be influenced. As the parameters of the field function will be modified through a numerical optimization process, the number of parameters will directly affect the dimension of searching space in optimization. The fewer dimension, the faster computation could be given in the optimization. Therefore, we propose a new field function as following

$$f_i(p) = \begin{cases} (1+B_i)^2(1-\frac{r_i^2}{R_i^2})^2, & \text{if } r_i \in [0, R_i] \\ 0, & \text{elsewhere} \end{cases} \tag{3}$$

where i represents the index of a primitive, $r_i = d(p, c_i)$ returns the Euclidean distance from a given point p to the center of blob c_i, $R_i = \sqrt{e_i^2(1+B_i^{-1})}$ defines the influence region, e_i is the radius of ith media-axis sphere, and B_i is a factor to adjust the shape of f_i. This field function actually defines an isosurface using points as skeletons. In our approach, we take the spheres in the most detail level of the medial axis sphere-tree as the initial blobs of the approximation model. The centers of the medial-

axis spheres form an approximation of the object's skeleton. To accelerate the optimization procedure, the positions and radii of blobs are fixed, so we only have one parameter to be optimized for each blob.

In this paper, we set the threshold t of the implicit surface equation to 1.0. By this setting, from Eq. (3), it is not difficult to find that the implicit surface defined by blobs is the sphere itself if there is only one medial-axis sphere. When there are a number of medial-axis spheres, the implicit surface covers the surface of the union of spheres tightly. This can be considered as another benefit of using Eq. (3). Figure 4 shows the relationship between medial-axis spheres and their corresponding isosurface.

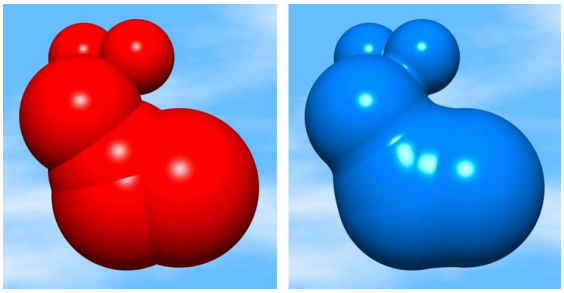

Figure 4: An example relationship between medial-axis spheres (left) and the corresponding implicit surface (right).

2.2.2 Optimization of blobby parameters

Since the medial axis sphere-tree is developed for collision detection purpose but not for surface approximation, visual artifacts will arise in the reconstructed shape if we take the medial-axis spheres as blobs and generate the implicit surface directly without optimization. This can be seen in the left of Figure 5. Therefore, we need to adjust blob parameters to achieve the best approximation.

Here we determine the parameters of blobs through an optimization procedure. The implicit surface formed by blobs is as defined in Eq.(1). Our optimization follows the principle that an ideal approximation of M given by $f(x) = 0$ is that every point on M exactly lies on $f(x) = 0$. Thus, we first sample the given object M into points; then, an objective function is defined as

$$E = \frac{1}{m}(\sum_{p}(f(p))^2) \qquad (4)$$

where p is a sample point on M and m is the number of sample points. By minimizing E, the best approximation is determined.

Since all the blobs come from medial-axis spheres, the spheres give a good estimation for the primitives' radii and positions. For the blobby model given in Eq.(3), only the parameter B_i is chosen to be adjusted during the optimization. In practice, we find that if the number of blobs is large, it is hard to compute the global optimization in a short time. To speed up, we adopt a local searching strategy to minimize E. In other words, we optimize the primitives one by one. After several iterations (usually 3), the surface $f(x) = 0$ gives a smooth approximation of M with small error. The surface shown in the right of Figure 5 is an optimized implicit surface from the spheres (level 3) in Figure 3.

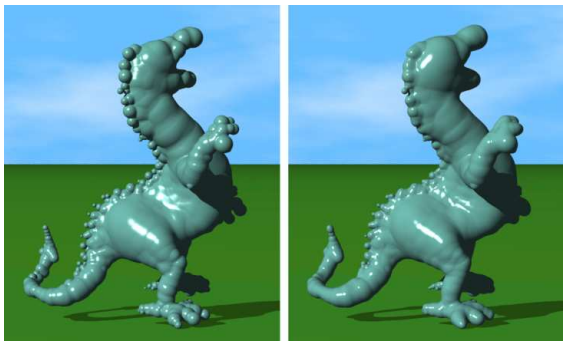

Figure 5: The implicit surface of the dragon model without optimization (left) and with optimization (right).

3 Metamorphosis

Objects with complex topology and geometry can be automatically approximated by blobby objects through our above method. This greatly benefits three-dimensional metamorphosis with liquid effect. In 3D morphing approaches, most models are initially defined as polygonal meshes. In order to solve the correspondence problem, some approaches convert polygonal models into implicit representations (e.g., the skeleton-based method [24], or the sphere-based techniques [19, 22]). Sometimes volume data can also be utilized to establish the correspondence [25]. Then, the blended surfaces in morphing are generated by extracting polygonal meshes from the interpolations of implicit surfaces (or volume data). Here we demon-

strate a morphing approach which is based on our blobby approximation of polygonal meshes.

In our blob-based surface approximation, an implicit surface is defined as an isosurface expressed by the field functions of primitives — blobs. The implicit surfaces are usually morphed by interpolating underlying primitives. Therefore, the correspondences between blobs need to be established. Several skeletal element based heuristics for matching and interpolating isosurface have been proposed ([26, 27]). They can be classified into two categories: 1) techniques that match elements according to their positions in space (so-called cellular matching), or 2) approaches that require extra information about the elements (i.e., hierarchical matching). Also, a pre-processing step is necessary to ensure that both initial and final shapes have the same number of primitives. For our blob-based morphing algorithm, the number of primitives, which are the blobs approximating given source and target polygonal meshes, is not necessary to be same. Both cellular matching and hierarchical matching can be performed automatically based on our blobby approximation technique. Here, we first establish the sphere matching in the medial axis sphere-trees of source and target objects, the blob matching is then obtained accordingly.

3.1 Cellular matching

In [22], the authors reformulated the primitive matching problem as a minimum weight bipartite graph matching problem. Here, we borrow some idea from them but develop a different scheme for sphere matching. When a 3D metamorphosis is expected between two objects M_A and M_B, the blobby approximations of them are modelled and their corresponding spheres are stored in sets Ω_A and Ω_B. First, we transform the spheres in Ω_A and Ω_B to align them into a common canonical position. Then, the matching pairs of spheres in Ω_A and Ω_B are established with the minimum matching-distance. A matching-distance $d(a, b)$ [22] between two spheres $a \in \Omega_A$ and $b \in \Omega_B$ is defined as

$$d(a,b) = \alpha((x_a - x_b)^2 + (y_a - y_b)^2 + (z_a - z_b)^2) + (e_a - e_b)^\beta \qquad (5)$$

where $(x_i, y_i, z_i), e_i$ are the center position and radius of sphere $i(i = a, b)$, and α, β are parameters to define the property of a matching-distance. By varying α, β, we can assign different weights to the size and location of the primitives. For all examples in this paper, we adopt α = 1 and β = 1.

Suppose there are m spheres in Ω_A and n spheres in Ω_B, the matching-distances between each pair of spheres in Ω_A and Ω_B are computed and stored in a distance matrix $D_{m\times n}$, where the element $d_{i,j} \in D_{m\times n}$ represents the matching-distance from the ith sphere in Ω_A to the jth sphere in Ω_B. After that, a matching matrix $C_{m\times n}$ is constructed by the following steps:

- Set every element in $C_{m\times n}$ to $zero$;
- Check every row in $D_{m\times n}$ — for the elements in the ith row of $D_{m\times n}$, if the kth one $d_{i,k}$ is minimum among all elements in the same row, $c_{i,k} \in C_{m\times n}$ is assigned to one;
- Every column in $D_{m\times n}$ is detected — if the kth element in the jth column $d_{k,j}$ is minimum among all other elements in the same column, the element $c_{k,j} \in C_{m\times n}$ is set to one.

After constructing the matching matrix, for an element $c_{i,j} \in C_{m\times n}$, if $c_{i,j} = 1$, a correspondence exists between the ith sphere in Ω_A to the jth sphere in Ω_B; otherwise, there is no link between them. Based on the matching matrix $C_{m\times n}$, every sphere in Ω_A has at least one corresponding sphere in Ω_B, and vice versa.

3.2 Hierarchical matching

Using the cellular matching, we can get the sphere matching between set Ω_A and Ω_B. However, if the number of spheres in Ω_A and Ω_B is large, the computation is pretty expensive; also, the matching results usually cannot provide correct feature correspondences. Here, the hierarchical structure in the sphere-tree Ψ is employed to overcome above problems.

Usually, the features of a given object are separated by spheres at coarse levels in Ψ. In a sphere-tree, the children spheres must cover the areas of given model that covered by their parent sphere. Based on this property, we can first relate the features between two objects by

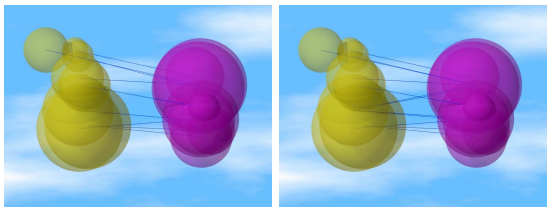

Figure 6: Comparison of primitives correspondences: left one (manually), and right one (automatically).

matching the spheres at coarse levels — either automatically or manually. Figure 6 shows a comparison about the correspondences established manually and automatically. After that, the correspondences of spheres in two matched features are further computed by the cellular matching scheme introduced above. Figure 7 illustrates a hierarchical matching between two given models represented in sphere-tree. In practical operation, the matching is always performed in two phases: the feature correspondences are firstly established between coarse level spheres of the source and target objects by automatically cellular matching — if manual adjustment is needed, just modify it; then, the children spheres in the pair of corresponding feature spheres are further matched automatically. This approach is a hybrid between global and local matching. Whether to perform the matching process automatically is given as an option to users.

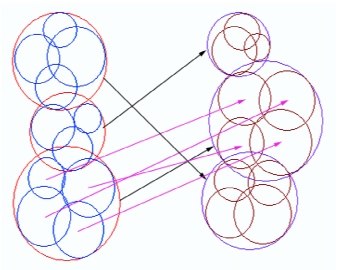

Figure 7: An example of hierarchical matching.

3.3 Interpolation of surfaces

After blob matching, we can interpolate between the matched blobs to get the transition between objects M_A and M_B. When one blob with weight ω_i corresponds to k blobs, we split the blob into k sub-blobs. The weight for each sub-blob is set to ω_i/k. We interpolate the following parameters: the centers, the radii, the weights, and the parameters B_is of each blob pair. The constraints of interpolation can be set to let the radii of spheres, or the volume of the implicit surface vary linearly. Computing the interpolation following latter constraint is expensive since it is very difficult to directly go from volume to the position of spheres (there is no general unique solution). Thus, we simply choose linear interpolation for all parameters. For the positional path for each pair of blobs during interpolation, we can either use linear path as shown in Figure 13, or use other user-defined paths (e.g., the result in Figure 1).

4 Results

The goal of this research is to develop a technique for liquid 3D morphing based on the implicit surfaces defined by blobby objects. We have implemented both the approximation algorithm and the 3D morphing with liquid effect on a PC with standard configuration (Inter PIV 2.4GHz CPU + 512MB RAM). Our algorithm has been tested on a variety of polygonal models. The results can be robustly obtained by our approach. In the following, several examples will be given, and the images are rendered by ray tracing [27].

4.1 Surface approximation by blobby objects

The surface approximation results are demonstrated in Figure 8–12. Each figure is arranged in pair, with a polygonal mesh and its corresponding blobby model. The medial axis sphere-tree of a given polygonal mesh is first constructed; then, the optimized blobby model is determined by minimizing the distance between the implicit surface defined by blobs and the sample points on the given model. To solve this minimization problem of Eq.(4), we get an approximate solution by using the downhill simplex method [28]. The models without sharp features can be approximated quite well by our approach; however, the surfaces with very sharp features are problematic for our algorithm because blobby model is not suitable for representing sharp features. In our testing examples, all major features on a given model can be well cap-

tured but the sharp features are rounded off. The running time analysis of our algorithm for the examples shown in Figure 8–12 is listed in Table 1. By our current implementation, all examples can be finished within 20 minutes.

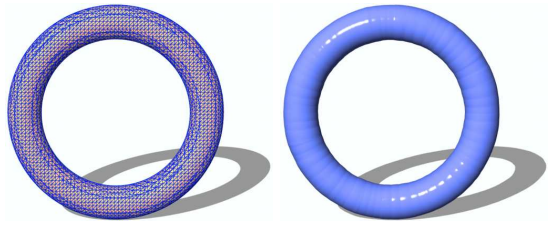

Figure 8: A torus polygonal model (left) and the implicit surface approximated by blobs (right).

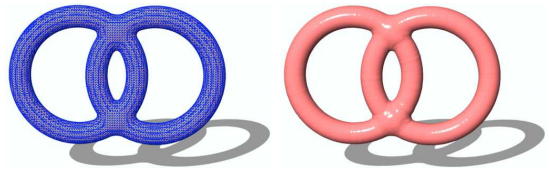

Figure 9: A two-torus polygonal model (left) and the reconstructed blobby surface (right).

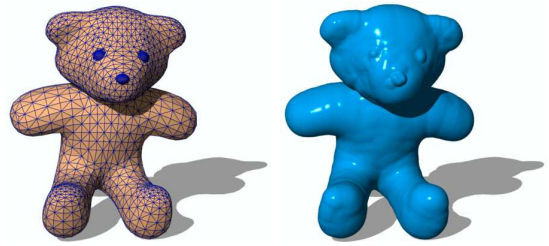

Figure 10: A bear polygon model (left) and its blobby representation (right).

4.2 3D morphing with liquid effect

The images shown in Figure 1 are six frames in the animation of "Liquid Robot Morphing", in which a lot of drops are firstly collected together and then morphed into a robot. The camera is moving during the animation. In Figure 13, a torus is liquidly morphed to a two-torus.

5 Conclusion

In this paper, a 3D morphing method with liquid effect is proposed. Firstly, we have designed an automatic algorithm for approximating polygonal meshes with blobby objects.

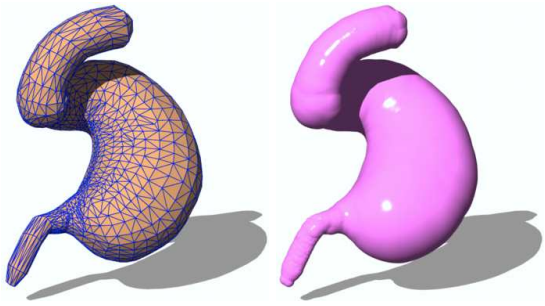

Figure 11: A stomach polygon model (left) and its blobby representation (right).

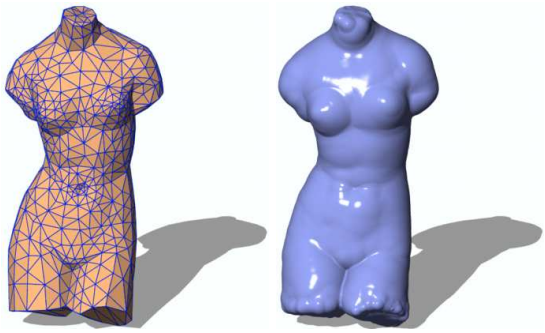

Figure 12: A Venus polygon model (left) and its blobby model (right).

The implicit representation using blobs is constructed through an energy optimization procedure so that the distance between the isosurface of blobs and the given mesh is minimized. Primitives in the medial axis sphere-tree of a given polygonal model are utilized as the initial guess of blobs — this greatly improves the robustness and efficiency of the blobby model based surface approximation. Objects with complex topology can be automatically approximated by blobby objects through our method, which greatly benefits the morphing applications. Secondly, we have developed a three-dimensional metamorphosis approach, which employs the reconstructed blobby objects of given polygonal meshes. With the help of the hierarchial structure of a sphere-tree, we establish the blob correspondences between the source and target models. The intermediate shape is obtained by linearly interpolating the parameters of blobby objects. Examples are given at the end of the paper. As a byproduct, the blobby approximation could also be utilized to compress 3D model data.

However, as mentioned before, the aliasing on sharp or thin features exists when using blobby objects to express the shape of given models. The reason is that spheres are isotropic. In our

Model	Figure	Time for sphere-tree construction (s)	Time for the optimization of blobby objects (s)	Total time(s)
Torus	7	300	19	319
Two-torus	8	912	111	1,023
Bear	9	594	83	677
Stomach	10	811	101	912
Venus	11	663	108	771

Table 1: The running time of blobby model approximation.

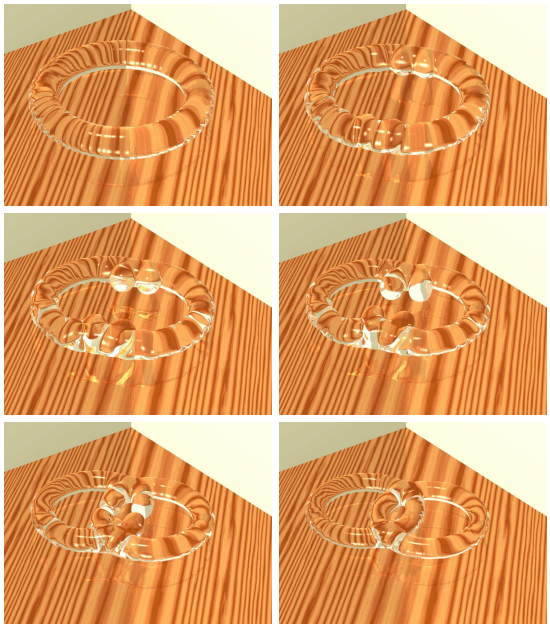

Figure 13: Liquid metamorphosis from a torus to a two-torus.

future work, anisotropic primitives, for example ellipsoids, will be considered to achieve a better approximation on the region with fine-scale features.

References

[1] J. F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, 1982.

[2] H. Nishimura, M. Hirai, T. Kawai, T. Kawata, I. Shirakawa, and K. Omura. Object modeling by distribution function and a method of image generation. *Transaction IEICE Japan*, J68-D(4):718–725, 1985.

[3] G. Wyvill, C. McPheeters, and B. Wyvill. Data structure for soft objects. *The Visual Computer*, 2:227–234, 1986.

[4] S. Muraki. Volumetric shape description of range data using blobby model. *Computer Graphics*, 25(4):227–235, July 1991.

[5] N. Tsingos, E. Bittar, and M. P. Gascuel. Semi-automatic reconstruction of implicit surfaces for medical applications. In *Computer Graphics International*, pages 3–15, June 1995.

[6] E. Bittar, N. Tsingos, and M. P. Gascuel. Automatic reconstruction of unstructured 3D data: combining a medial axis and implicit surfaces. *Computer Graphics Forum*, 14:457–468, 1995.

[7] S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones. Adaptively samples distance fields:a general representation of shape for computer graphics. In *Proc. SIGGRAPH'2000*, pages 249–254, 2000.

[8] G. Yngve and G. Turk. Robust creation of implicit surfaces from polygonal meshes. *IEEE Transactions on Visualization and Computer Graphics*, 8(4):346–359, 2002.

[9] G. Turk and J. F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855–873, October 2002.

[10] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H. P. Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22(3):463–470, 2003.

[11] C. Shen, J. F. O'Brien, and J. R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Transactions on Graphics*, 23(3):896–904, 2004.

[12] B. Wyvill, C. McPheeters, and G. Wyvill. Animating soft objects. *The Visual Computer*, 2:235–242, 1986.

[13] B. Wyvill, J. Bloomenthal, T. Beier, J. Blinn, A. Rockwood, and G. Wyvill, editors. *Modeling and Animating with Implicit Surfaces*, Dallas,USA, August 1990. SIGGRAPH'90 Course Notes No 23.

[14] A. Pasko and V. Savchenko. Constructing functionally defined surfaces. In *Implicit Surfaces'95*, pages 97–106, Grenoble, France, April 1995.

[15] E. Galin and S. Akkouche. Blob metamorphosis based on Minkovski sums. *Computer Graphics Forum*, 15(3):143–154, 1996.

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

[17] D. Cohen-Or, D. Levin, and A. Solomovici. Three-dimensional distance field metamorphosis. *ACM Transactions on Graphics*, 17(2):116–141, 1998.

[18] G. Turk and J. F. O'Brien. Shape transformation using variational implicit functions. In *Proc. SIGGRAPH'99*, pages 335–342, Los Angeles, California, August 1999.

[19] G. Treece, R. Prager, and A. Gee. Volume-based three-dimensional metamorphosis using sphere-guided region correspondence. *The Visual Computer*, 17(7):397–414, 2001.

[20] N. Foster and R. Fedkiw. Practical animation of liquids. In *Proc. SIGGRAPH'2001*, pages 23–30, 2001.

[21] D. Enright, S. Marschner, and R. Fedkiw. Animation and rendering of complex water surfaces. *ACM Transactions on Graphics*, 21(3):736–744, 2002.

[22] V. Ranjan and A. Fournier. Shape transformations using union of spheres. Technical Report TR-95-30, Imager Computer Graphics Laboratory, Department of Computer Science, University of British Columbia, 1995.

[23] G. Bradshaw and C. O'Sullivan. Adaptive medial-axis approximation for sphere-tree construction. *ACM Transactions on Graphics*, 23(1):1–26, January 2004.

[24] J. Bloomenthal and C. Lim. Skeletal methods of shape manipulation. In *Shape Modeling International '99*, pages 44–47, Aizu-Wakamatsu, Japan, 1999.

[25] M. Sramek and W.E. Kaufman. Alias-free voxelization of geometric objects. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):251–267, July 1999.

[26] B. Wyvill. Metamorphosis of implicit surfaces. In *Modeling, Visualizing and Animating with Implicit Surfaces*, Anaheim, CA, USA, August 1993. SIGGRAPH'93 Course Notes No 25.

[27] J. Bloomenthal, C. Bajaj, J. Blinn, B. Wyvill, and G. Wyvill, editors. *Introduction to Implicit Surfaces*. Morgan Kaufman Publishers, 1997.

[28] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge, 1988.