Layered Deformable Models with Implicit Surfaces

Marie-Paule Cani-Gascuel

iMAGIS/GRAVIR-IMAG[1]
BP53, 38041 Grenoble Cedex 9 - France,

e-mail: `Marie-Paule.Cani@imag.fr`

Abstract

Several challenging problems in the animation of deformable objects can be solved by combining existing models with implicit surfaces. The latter are used as an extra layer which coats any kind of structure that moves and deforms over time. Implicit surfaces detect collisions, model local deformations, and transmit response forces to the inner structures. They also control the variations of the object's volume. We present several applications from the animation of organic shapes to the simulation of soft inelastic substances that can separate into smaller pieces or fuse into larger ones.

Résumé

Plusieurs problèmes difficiles dans le domaine de l'animation d'objets déformables peuvent être résolus en combinant les modèles existants avec des surfaces implicites. Utilisées pour habiller une structure en mouvement, ces dernières détectent les collisions, modélisent les déformations locales qui en résultent, et transmettent les forces de réponse à la structure interne. Elles peuvent également contrôler les variations de volume de l'objet à animer. Nous présentons plusieurs applications, allant de l'animation de formes organiques à la simulation de substances hautement déformables.

1 Introduction

Animating objects that deform over time is one of the main challenges of Computer Animation. Physically-based models provide physically-based models provide very good tools for automating the animation of inanimate objects [27, 26, 22, 31]. They may also be used in combination with other techniques for animating complex organic shapes such as animals or human figures [17, 5, 14]. In all cases, the capability of surfaces to respond to collisions with other objects in a scene by deforming locally in the contact region is an important aspect of realism. Controlling the variations of an object's volume is very important too: most inanimate material maintain their volume to a constant value [23], while animating local changes of volume is needed in character animation [5].

As emphasized in [5], a layered construction is an efficient tool for creating complex models for animation. It often provides the user with parameters that are more intuitive and easier to use. Layered constructions were used for instance in [26, 5, 11, 17, 13]. In most cases, an inner layer controls large scale behavior (global motion and large scale deformations) while layers modeling flesh and skin provide the deformations of the surface to be rendered. Ideally, this surface should be used instead of the inner structure for detecting and processing collisions with other objects. Due to the complexity of most surface models, this can scarcely be done. For instance, in [17, 13], the inner structure animates the control points of a spline surface modeling the "skin" of the model, but the equation of the surface is not used for collision detection. Similarly, the flesh and skin layers should be used instead of the inner structure for providing control on the object's volume.

Implicit surfaces generated by skeletons such as blobs, metaballs and soft objects [3] are especially well suited to a layered construction. Defined as iso-surfaces of scalar fields that decrease with the distance to a geometric skeleton, they immediately follow the motion of the latter. There is no restriction on the motion of the skeleton:

- In character animation, the skeletal elements are defined with respect to an articulated structure made of connected rigid links. The implicit surface that fleshes it builds convincing organic shapes [25, 30] that may be coated with a parametric "skin" (such as a B-spline which control points are projected at each time step onto the isosurface) to ease texture mapping and rendering.

- If an unstructured skeleton such as a particle system is used, the number of connected components of the implicit surface will change according to the time varying distance between particles. This gives an easy way of modeling highly deformable substances subject to topological changes, such as viscous liquids [18, 28, 29].

However the implicit surface is used as a purely geometric layer in all the works referenced above. It does not

detect collisions, and no local deformation is generated in this case, although the flesh layer should be "soft".

This paper shows that implicit surfaces can provide an *active* deformable layer that offers control over an object's volume and handles collisions, including the precise modeling of surfaces of contact with obstacles. Implicit surfaces are used as an extra layer that coats any kind of structure that moves and deforms over time. While the base structure computes large scale behavior, the implicit layer models both "flesh" and "skin": it generates local deformations during collisions, and can be used to maintain the object's volume to a specified value.

Section 2 introduces the layered approach we use. Sections 3 and 4 presents two benefits of using an implicit layer: the precise modeling of contact between deformable bodies, and the control of the object's volume. These techniques are illustrated in Sections 5 and 6 by two different applications: the design of organic shapes such as simplified characters and the animation of soft inelastic substances capable of separation and fusion.

2 Layered Models Using Implicit Surfaces
2.1 Implicit surfaces generated by skeletons
An implicit surface [3] generated by a set of skeletons s_i $(i = 1 \ldots n)$ with associated "field functions" f_i is defined, at the isovalue c, by:

$$\{P \in \mathbb{R}^3 \mid f(P) = c\} \text{ where } f(P) = \sum_{i=1}^{n} f_i(P)$$

It surrounds a solid whose points satisfy $(f(P) \geq c)$, which may have several disconnected components. Normal vectors are directed along the field's gradient. The skeletons s_i can be any geometric primitive admitting a well defined distance function: points, curves, parametric surfaces, simple volumes, etc. The field contributions f_i are decreasing functions of the distance to the associated skeleton. Most field functions associate a restricted scope of influence to each skeleton in order to provide local control of the surface and to optimize computations.

2.2 Embedding implicit surfaces into a layered construction
Implicit surfaces generated by skeletons can be easily embedded into a layered construction. In our framework, the user defines a deformable object by specifying:

1. A "base structure" with an associated animation algorithm. For instance, this structure could be a rigid solid, an articulated structure made of several such solids, a mass/spring network or a particle system animated with physically-based animation techniques. It may also be a hierarchy of local frames animated with either direct or inverse kinematics.

2. An implicit layer, that "coats" the base structure. This layer is built by defining the skeletons that generate the implicit surface in local frames of the base structure.

During an animation, the implicit layer immediately follows the motion and deformations of the base structure, coating it with a smooth surface that can be used for rendering. The topology of this surface may change over time, since the relative motion of skeletons may yield separation or blending.

2.3 Animation algorithm
As stressed before, the coherence between the representation of an object and the model used for collision processing is very important for generating convincing motion. Instead of using a purely geometric definition for the implicit layer, we use the implicit deformable model defined in [12]. This model, which will be reviewed in the next section, defines a correspondence between applied forces and deformations that approximates elastic behavior. It can therefore be used for collision processing.

The general scheme for animating the layered model develops as follows:

1. Animate the base structure (for instance, if it is a physically-based model, integrate its equations of motion according to externally-applied forces).

2. Compute the implicit layer from the new positions of the skeletons that generate it and from user-defined constraints on the oject's volume.

3. Use the implicit layer for detecting collisions, for modeling contacts between colliding objects (through the generation of local deformations), and for computing response forces.

4. Add these forces to the set of external actions applied to the base structure at the next time step.

If the base structure is not a physically-based model the response forces computed by the implicit layer will not influence subsequent large-scale motion. The layered model is still interesting since local deformations of the object's surface will be automatically generated.

3 Deformation of Surfaces under Collisions
The capability of surfaces to respond to collisions with other objects by deforming locally in the contact region is an important aspect of realism, whether the animation

is calculated by a physics-based model or not [16]. We expect an object that is compressed by a collision to "inflate" locally around the contact region to compensate the volume loss due to compression. Simulating such an effect during collisions using finite elements is an expensive process [14]. Implicit surfaces offer an alternative solution: collision detection is performed efficiently due to a simple inside/outside test associated with the implicit formulation of objects, precise contact surface is created between two colliding objects and the amount of surface compression imposed during a collision can be used to calculate the forces to be applied to the underlying skeleton in the following animation step. Finally, assuming that local deformations around the contact region are a purely visual effect without influence on the motion, a geometric model can be used to calculate them.

3.1 Modeling collisions

Implicit surfaces are particularly advantageous for collision detection: a given point P is inside an implicit object if and only if $f(P) \geq c$. This results in a simple detection of interpenetration with obstacles, which is achieved by performing the inside/outside test for a given implicit object on the discretization points of an obstacle.
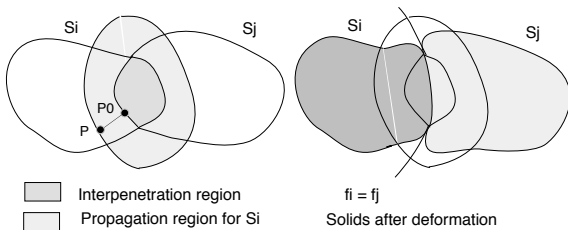

Figure 1: Deformations during a collision between two deformable objects.

Let f_1 and f_2 be the field functions of the two objects. When an interpenetration is detected:

1. Negative compressing fields equal to $c - f_2$ and $c - f_1$ are respectively added to the field functions f_1 and f_2 in the *interpenetration region* (see figure 1). It results in a local compression of the two objects and the creation of a contact surface defined by the equation $f_1 = f_2$.

2. At the same time, positive dilating fields are applied in the *propagation region* defined around the interpenetration region. These terms create local object dilation that appears around the contact surface. They are calculated so that the C^1 continuity of the deformed surface is preserved.

3. Reaction forces R_i are calculated on the contact surface, reflecting the elastic characteristics of the materials in the direction normal to the contact surface (see [12, 4, 3]):

$$R_1 = (f_2 - c)N_1 = -(f_1 - c)N_2 = -R_2$$

3.2 Local deformations in the propagation area

Since only the contact forces calculated in step 3 influence the motion of objects, a purely geometric method can be used to calculate the local deformations in step 2.

Nevertheless, the C^1 continuity constraint is difficult to satisfy. A first solution to this problem was described in [12]. However, computing the field value at a point P required the search of the closest point P_0 of the interpenetration region, which resulted into extremely expensive computations. We present a simple and efficient method for generating the same type of local deformation [20].

Modeling compression in the interpenetration region (step 1 above) is relatively intuitive since the field of each object is used to hollow out the others. We are looking for a similarly simple formulation for the dilation term.

To ensure the C^1 continuity of the deformed surfaces, their normals have to vary continuously, in particular on the border of the interpenetration region. Thus, at all points P_0 lying on this border, the gradient of the dilation field has to be equal to the gradient of the compression field. Moreover, the dilation field and its derivatives have to be zero at the outside border of the propagation region, where the dilated part joins the undeformed object.

We have chosen to express the dilation term to be added to f_1 in the propagation region as:

$$b_1(f_2(P))$$

where b_1 is a function defined on the interval $[c_1, c]$ that satisfies:

$$b_1(c) = 0 \quad b_1'(c) = -1 \quad b_1(c_1) = 0 \quad b_1'(c_1) = 0$$

The gradient of the compressing field function is equal to $-\nabla f_2(P_0)$. Thus, the tangent continuity constraint is satisfied at the border of the interpenetration region since:

$$\begin{aligned} \nabla(b_1(f_2))(P_0) &= b_1'(f_2(P_0))\nabla f_2(P_0) \\ &= b_1'(c)\nabla f_2(P_0) = -\nabla f_2(P_0) \end{aligned}$$

The parameter c_1, associated with the object 1, controls the extent of the propagation region which is limited by the isosurface $f_2(P) = c_1$ of the object 2. By definition of b_1, the value and derivatives of the dilation field are zero at this limit. In practice, the function shown in figure 2 is used for b_1. It is controlled by its maximal height h at a given parameter value $m \in (c_1, c)$ and is defined as
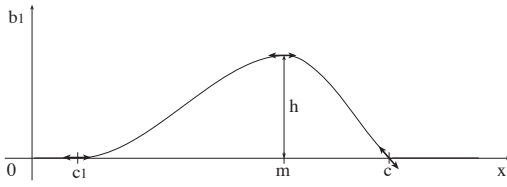
Figure 2: Function modeling dilation.

a union of two cubic functions defined over the intervals $[c_1, m]$ and $[m, c]$.

The advantage of this formulation is its low cost: the modification of the field function at a given point P in the propagation region only requires an evaluation of $f_2(P)$ (already known to determine the region P belongs to) and its combination with the function b_1 given above. There is no longer the need to find the closest point on the border of the interpenetration region. The method can therefore be used during an interactive animation process.

3.3   Anisotropic surface deformation

The method presented above assumes that surface dilation around the interpenetration region is uniform. However, this is only valid for frontal collisions. In the general case, we expect a higher bump extending for a short distance in the part of the surface in front of the arriving object, and a lower bump extending for a longer distance behind it (see Figure 3b).

This effect can be achieved with our approach by varying the parameter c_1 that defines the size of the propagation region according to the relative velocity of the objects:

$$c_1(P) = \frac{1}{2} c_0 \left(1 + \frac{D_2 \cdot \nabla f_1(P)}{\|D_2\| \|\nabla f_1(P)\|} \right)$$

where D_2 is the velocity vector of object 2 with respect to object 1, and c_0 is a parameter specified for object 1. The dissymmetrical surface dilation that is generated preserves C^1 continuity of the objects and visually emphasizes their relative motions.
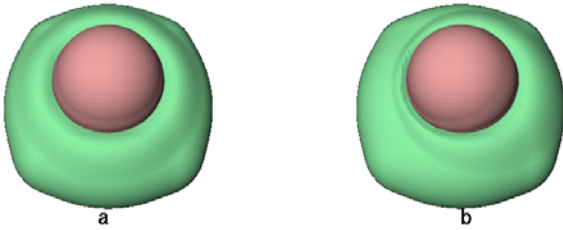

Figure 3: A rigid sphere falls into a deformable block: (a) vertically (b) from the right (dissymmetric bump).

4   Control of an Object's Volume

Unwanted volume variations are exacerbated in implicit surface animation. They are produced by the field blending process during the relative motion of the skeletons, and they may be particularly annoying when the object undergoes separation or fusion.

This section presents a simple solution for controlling the volume of objects defined by implicit surfaces, which is applicable to any field function and isovalue [8, 4]. We first present the method in terms of maintaining an object's volume to a constant value. Then we explain how to extend it to the general case where volume variations are animated by the user.

4.1   Local volume variations

Suppose the volume of an implicit object has been modified, as in Figure 4 (a), by the relative motion of some skeletons of the implicit layer. A solution for avoiding the variation consists of adjusting the strength of the field functions so that the volume keeps its initial value. However, these adjustments should not be done in areas where the object has not been deformed. As a consequence, volume variations should be detected and treated locally.
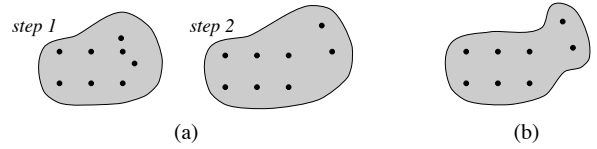

Figure 4: (a) Volume variations of an implicit surface generated by point skeletons between two steps of animation. (b) Volume controlled locally in step 2.

We define the *local volume* V_i associated with a skeleton s_i as the volume of the skeleton's *territory* $\mathcal{T}_i$:

$$\mathcal{T}_i = \{ P \in \mathbb{R}^3 \mid (f(P) \geq c) \text{ and } (\forall j \ f_i(P) \geq f_j(P)) \}$$

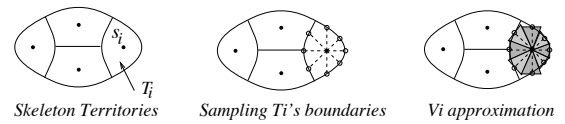Skeleton Territories   Sampling Ti's boundaries   Vi approximation

Figure 5: Approximation of local volumes.

We use the sampling method proposed in [9] for approximating values of local volumes. Each skeleton s_i contributing to the implicit layer emits a set of sample points in directions that are fixed in its local coordinate system, and are well distributed around it. Sample points stop when they reach either the iso-surface or the border of $\mathcal{T}_i$. Then, the local volume V_i is approximated as

the sum of small pyramidal volumes defined around each sample point sent by s_i (see Figure 5):

$$V_i = b_i \sum_{P \in \mathcal{P}_i} d(P, s_i)^3$$

where $\mathcal{P}_i$ is the set of sample points sent by a skeleton s_i, d is the distance function, and b_i only depends on the angular distribution of samples for s_i.

4.2 Volume control

We control local volume variations by associating a proportional-derivative controller to each skeleton. This controller can be seen as a black box that, given the current local volume $V_{i,t}$ and the value $V_{i,0}$ to reach or to maintain, outputs an adequate adjustment of the field function f_i of this particular skeleton.

For our application, the way to modify f_i must be chosen carefully since the norm of f_i's gradient gives the object local stiffness (see the expression of response forces in Section 3). In order to adjust the volume of skeleton territories without modifying the object's physical properties, we combine the original field function with a translation $\epsilon_{i,t}$. At each time step, the field originally defined by the decreasing function of the distance $f_i(P) = F_i(d(P, s_i))$ is replaced by:

$$f_{i,t} = F_i \left(d(P, s_i) - \epsilon_{i,t} \right).$$

Since we need regular shape variations, we control the time derivative $\dot{\epsilon}_{i,t}$ of the translation parameter rather than its value. The inputs of the controller are then the normalized volume variation $\Delta_{i,t}$ and its time derivative:

$$\Delta_{i,t} = \frac{V_{i,t} - V_{i,0}}{V_{i,0}} \qquad \dot{\Delta}_{i,t} = \frac{V_{i,t} - V_{i,t-dt}}{V_{i,0}\, dt}$$

Its output is: $\dot{\epsilon}_{i,t} = \alpha \Delta_{i,t} + \beta \dot{\Delta}_{i,t}$ where α and β are appropriately chosen parameters.

This method, defined for maintaining a constant volume during an animation, can be extended in order to impose specific volume variations that may be locally specified by the user: the target volumes $V_{i,0}$ simply have to be changed over time. These capabilities are useful in a broad range of applications, some of which are presented in the next two sections.

5 Animation of Organic Shapes

Implicit surfaces are particularly well suited to modeling organic forms [19, 2, 24] since they generate a smooth surface around skeletons of arbitrary geometry and topology. We give two examples of animations of such forms based on our layered formalism.

5.1 Coating characters with soft flesh

Characters such as human figures or animals are usually animated using an articulated structure ie. a hierarchy of rigid links. Adding a layer modeling flesh and skin, that follows the rigid skeleton's motion but generate local surface deformations will greatly improve the visual quality of the animation. In the terminology introduced in Section 2, a character can be modeled using:

- Base structure: an articulated structure animated with key frames, inverse kinematics, or through the use of physically-based animation.

- Implicit layer: each skeleton contributing to the implicit surface is defined in one of the local frames of the base structure.

An extra flesh layer may be added between the base structure and the implicit layer, by defining some mass-points connected by damped springs to elements of the base structure. Using these extra mass-points as some of the skeletons contributing to the implicit layer will add more life to the character [21].

Characters are animated through the general animation algorithm presented in Section 2. Local control of volume variations can be used for animating "muscles" as in [5]. However, the problem of avoiding unwanted blending effects has to be solved.

Avoiding unwanted blending effects

Using implicit surfaces for character animation requires the use of a *blending graph*, that specifies the manner of combining the contributions f_i from different parts of a character's skeleton [21, 4]. For example, an arm should be blended with the body at the shoulder while a collision should be detected between a hand and the body. A solution consists in defining a neighboring graph between the different skeletons, and stating that a skeleton's field only blends with contributions from neighboring skeletons. To compute the field value at point P:

1. Compute all the field contributions at point P,

2. Select the predominant contribution from those of groups of skeletons that blend together,

3. Return this value without summing the other field contributions.

This algorithm avoids surface discontinuity during the controlled blending process. However, the method can not guarantee $\mathcal{C}^1$ continuity everywhere [4].

During the animation, collisions are processed between parts of the implicit surface that do not blend together, using the algorithm described in Section 3 (see Figure 6).
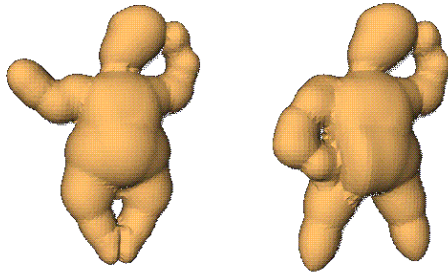
Figure 6: Deformations of a simple character.

Ray-traced lips   Side view

Structure used   Animation snapshot

Figure 7: Animating lips with implicit surfaces

5.2 Talking Lips

Lips animation is an important element in the generation of natural looking synthetic actors. Large scale deformations of lips should be synchronized with speech. Local deformations should be generated when they interact with each other or with other objects such as the teeth or a cigarette. The use of implicit surfaces within our layered framework yields a simple solution to this problem. The geometric modeling of realistic lips shapes can be obtained from acquired data by using reconstruction algorithms such as those in [1, 10]. Then, the model is animated using the following structure [15] (see Figure 7):
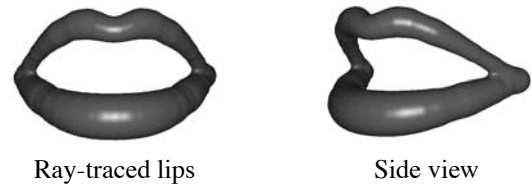
- Base structure: for each lip, a set of control points located along a "skeleton curve" and animated with key-frames, ensuring synchronization with speech.

- Additional flesh layer: mass-points linked to control points through damped springs of zero rest length.

- Implicit layer: implicit surface whose skeletons are the mass-points of the flesh layer.

When the implicit layer deforms locally due to intercollisions between the lips or to contacts with other objects, response forces are transmitted to the additional flesh layer. The latter's motion thus slightly differs from the prescribed motion of the base structure, adding more life to the animation.
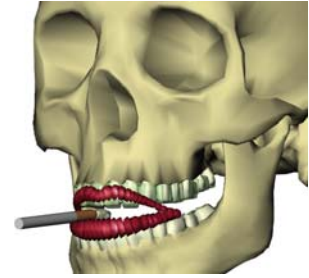
6 Animation of Soft Substances

The soft substance model we present here, first introduced in [8], is a good example for illustrating the capability of implicit surfaces to undergo separation and fusion. It is built from the layered construction by using:

- Base structure: a particle system such as those in [18, 29, 6]. Interactions between particles are modeled by attraction/repulsion forces combined with friction forces.

- Implicit layer: an implicit surface defined by point skeletons located on each particle. We use field contributions with relatively large thickness and radius of influence in order to give a smooth aspect to the simulated material even if only a few particles are used. Local volume controllers are associated with each skeleton in order to prevent volume variation.

Animation is computed from the algorithm in Section 2. The next paragraphs explain how we handle separation and fusion of the substance.

Modeling separation

When the particle system moves and deforms, a piece of substance may separate into several components, due to the relative motion of point skeletons defining the surface. However, if these disconnected chunks come back close to each other, they will blend rather than collide, since they are considered to be parts of the same surface. This artifact is related to the unwanted blending problem referred earlier. However, the problem is more complicated here since the blending properties between the set of point skeletons must change during the animation.

Our solution is based on the computation of a time varying blending graph. At each animation step, the current blending graph is stored as a list of neighbors, so called "blending list", associated with each point skeleton. Processing unwanted blending is done by reducing blending lists each time the implicit surface breaks into disconnected components that must not blend anymore.

Modeling fusion under compression

Blocks of soft substance such as clay or dough merge under compression forces that exceed a specified threshold. This behavior can easily be simulated with our model by defining a *fusion threshold* for the substance. Each time compression forces along a contact surface between two skeleton territories exceed the fusion threshold, blending lists are merged. At the next time step, fields from the two pieces will then locally blend in this area, while collisions will still be computed between the rest of the components as illustrated in Figure 8.

Figure 8: Progressive fusion under compression.

For handling substances that immediately merge after a collision such as in Figure 9, the fusion threshold is set to zero. Here, volume preservation is essential. Otherwise, a very large and sudden increase of volume would be produced between the two last frames.
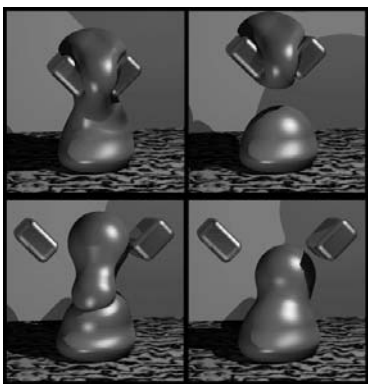

Figure 9: Soft substance grabbed away by pliers.

7 Conclusion

This paper has presented a general framework for building layered deformable models with implicit surfaces. Particularly well suited to a layered construction, implicit surfaces can be used for coating any base structure that moves over time. It defines a smooth surface around it that can be used for rendering, and offers simple yet precise processing of collision and contact. Moreover, control of deformed objects' volume is possible, even when the objects undergo significant changes such as separation or fusion.

We have illustrated this framework by detailing two very different applications: the animation of organics shapes, for which the skeleton keeps a well defined structure when animated, and the simulation of soft substances performing separation and fusion, for which the skeleton – a set of particles – is totally unstructured.

The first application could lead to interesting developments in the character animation area. Our layered framework offers a compact way of modeling both geometry and the physical characteristics of simplified human figures. Local adjustments of volume through time can be used for generating more expressive animations. Lastly, the capability of processing collisions and contacts between a character and other objects of the scene would be an essential benefit of our approach.

For the second application (animation of soft substances) we are currently experimenting with an adaptive algorithm based on particle division/fusion for optimizing the animation of the base structure modeled with smoothed particles [6]. In this framework, we developed a new approach, called "active implicit surface", for modeling the implicit layer [7].

References

[1] Eric Bittar, Nicolas Tsingos, and Marie-Paule Gascuel. Automatic reconstruction of unstructured 3d data: Combining medial axis and implicit surfaces. *Computer Graphics Forum*, 14(3):457–468, September 1995. Proceedings of Eurographics'95, Maastricht, the Netherlands.

[2] Jules Bloomenthal. Skeletal design of natural forms. *PHD Thesis*, The University of Calgary, January 1995.

[3] Jules Bloomenthal, editor. *Introduction to Implicit Surfaces*. Morgan Kaufmann, July 1997.

[4] Marie-Paule Cani-Gascuel and Mathieu Desbrun. Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):39–50, March 1997.

[5] John E. Chadwick, David R. Haumann, and Richard E. Parent. Layered construction for deformable animated characters. *Computer Graphics*, 23(3):243–252, July 1989.

[6] Mathieu Desbrun and Marie-Paule Gascuel. Smoothed particles: A new approach for animating highly deformable bodies. In Springer Computer Science, editor, *7th Eurographics Workshop on Animation and Simulation*, pages 61–76, Poitiers, France, September 1996.

[7] Mathieu Desbrun and Marie-Paule Cani-Gascuel. Active implicit surface for animation. In *Graphics Interface'98*, June 1998.

[8] Mathieu Desbrun and Marie-Paule Gascuel. Animating soft substances with implicit surfaces. In *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 287–290. ACM SIGGRAPH, Addison Wesley, August 1995. Los Angeles, CA.

[9] Mathieu Desbrun, Nicolas Tsingos, and Marie-Paule Gascuel. Adaptive sampling of implicit surfaces for interactive modeling and animation. In *Implicit Surfaces'95—the First Eurographics Workshop on Implicit Surfaces*, pages 171–185, Grenoble, France, April 1995.

[10] Eric Ferley, Marie-Paule Cani-Gascuel, and Dominique Attali. Skeletal reconstruction of branching shapes. *Computer Graphics Forum*, 16(5), December 1997. An early version of this paper appeared in *Implicit Surfaces'96*, Eindhoven, The Netherlands, oct 1996.

[11] David R. Forsey. A surface model for skeleton-based character animation. In *Second Eurographics Workshop on Animation and Simulation, Vienna, Austria*, pages 55–73, September 1991.

[12] Marie-Paule Gascuel. An implicit formulation for precise contact modeling between flexible solids. *Computer Graphics*, 27:313–320, August 1993. Proceedings of SIGGRAPH'93 (Anaheim, CA).

[13] Marie-Paule Gascuel, Anne Verroust, and Claude Puech. A modeling system for complex deformable bodies suited to animation and collision processing. *Journal of Visualization and Computer Animation*, 2(3):82–91, August 1991. A shorter version of this paper appeared in *Graphics Interface'91*.

[14] Jean-Paul Gourret, Nadia Magnenat Thalmann, and Daniel Thalmann. Simulation of object and human skin deformations in a grasping task. *Computer Graphics*, 23(3):21–29, July 1989. Proceedings of SIGGRAPH'89 (Boston, MA, July 1989).

[15] Thierry Guiard-Marigny, Nicolas Tsingos, Ali Adjoudani, Christian Benoît, and Marie-Paule Gascuel. 3d models of the lips for realistic speech animation. In *Computer Animation'96*, pages 80–89, Geneva, Switzerland, June 1996.

[16] Jason Harrison and David R. Forsey. A kinematic model for collision response. In *Fifth Eurographics Workshop on Animation and Simulation*, Oslo, Norway, September 1994.

[17] Gavin Miller. The motion dynamics of snakes and worms. *Computer Graphics*, 22(4):169–177, August 1988. Proceedings of SIGGRAPH'88 (Atlanta, Georgia).

[18] Gavin Miller and Andrew Pearce. Globular dynamics: A connected particle system for animating viscous fluids. *Computers and Graphics*, 13(3):305–309, 89. This paper also appeared in SIGGRAPH'89 Course notes number 30.

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

[20] Agata Opalach and Marie-Paule Cani-Gascuel. Local deformations for animation of implicit surfaces. In *SCCG'97*, Bratislava, Slovakia, 1997.

[21] Agata Opalach and Steve Maddock. Implicit surfaces: Appearance, blending and consistency. In *Fourth Eurographics Workshop on Animation and Simulation*, pages 233–245, Barcelona, Spain, September 1993.

[22] Alex Pentland and John Williams. Good vibrations: Modal dynamics for graphics and animation. *Computer Graphics*, 23(3):215–222, July 1989. Proceedings of SIGGRAPH'89 (Boston, MA, July 1989).

[23] John Platt and Alan Barr. Constraint methods for flexible models. *Computer Graphics*, 22(4):279–288, August 1988. Proceedings of SIGGRAPH'88 (Atlanta, Georgia).

[24] Jianhua Shen. Human body modeling and deformation. *PHD Thesis*, EPFL, Lausanne, Switzerland, September 1996.

[25] Jianhua Shen and Daniel Thalmann. Interactive shape design using metaballs and splines. In *Implicit Surfaces'95—the First Eurographics Workshop on Implicit Surfaces*, pages 187–195, Grenoble, France, April 1995.

[26] Demetri Terzopoulos and Kurt Fleischer. Modeling inelastic deformations: Viscoelasticity, plasticity, fracture. *Computer Graphics*, 22(4):269–278, August 1988. Proceedings of SIGGRAPH'88 (Atlanta, Georgia).

[27] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *Computer Graphics*, 21(4):205–214, July 1987. Proceedings of SIGGRAPH'87 (Anaheim, California).

[28] Demetri Terzopoulos, John Platt, and Kurt Fleisher. Heating and melting deformable models (from goop to glop). In *Graphics Interface'89*, pages 219–226, London, Ontario, June 1989.

[29] David Tonnesen. Modeling liquids and solids using thermal particles. In *Graphics Interface'91*, pages 255–262, Calgary, AL, June 1991.

[30] Russel Turner. Leman: A system for constructing and animating layered elastic characters. In *Computer Graphics-Developments in Virtual Environments*, pages 185–203, Academic Press, San Diego, CA, June 1995.

[31] Andrew Witkin and William Welch. Fast animation and control for non-rigid structures. *Computer Graphics*, 24(4):243–252, August 1990. Proceedings of SIGGRAPH'90 (Dallas, Texas, August 1990).