4^{th} Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007)
J. Dingliana, F. Ganovelli (Editors)

Medial Surface-Based Real Time Simulation of Elastic Objects

M. Pfaff and C. A. Wüthrich

CoGVis/MMC
Faculty of Media
Bauhaus-University Weimar
D-99421 Weimar, Germany

Abstract

This paper presents a method for the real time simulation of the dynamical behaviour of elastic objects which bases on the medial surface. From a surface based 3D model, the medial surface is generated automatically without any intervention by the user. The medial surface is used as a skeleton for movement. It is connected to the surface of the model through a mass-spring system based on surface bone points that allows the simulation of surface movement. A new technique for constraining movement within the mass-spring system is used to stabilize the physical shape of the object. Different parameter settings allow the simulation of a variety of different elastic materials. Joints can be applied and simulated by parting the skeleton into different joint sections.

Categories and Subject Descriptors (according to ACM CCS): I.6.8 [Simulation and Modeling]: Animation

1. Introduction

Three-dimensional animations have become popular in recent years. Most advertisements, animated movies and computer games rely on three-dimensional animation nowadays. Special effects are often done on computers and integrated into the scene. Often puppet-like movie characters, which consist of elastic material such as latex, foam plastics or fabric, appear in these movies. To animate such puppets realistically, their surface materials have to be rendered with good detail. But also the physical movement on the surface has to be simulated in a realistic way. Due to their complexity the surface rendering and the physical simulation are usually calculated offline for animations. Only recently computer games have started to compute the physics and the surface rendering in real time. To make a physical simulation work in real-time, usually *finite element methods* (FEMs) or a *mass-spring system* are applied. Applications that make use of these technologies were made for medical surgery simulations [CCO02, GCMS00], for cloth simulation [MTCK*04, BW98] and for complete object animations [CGC*02, CKB03]. The soft material layer in these simulation techniques is usually supported by a stiff skeleton which acts as method for a shape stabilization and as a movement constraint for the surface. Often it is the user that has to generate these structures by hand. A basic skeleton consisting of connected edges, which let the skeleton look like a stick-figure, is often used for animations in software frameworks like Maya or 3D Studio MAX. Such one-dimensional skeletons have the advantage of being easily defined, but are on the other hand not able to characterize physical behavior realistically. Locally very detailed simulations built on this basis tend to look artificial in a sense of being too uniform, because the skin seems to have the same thickness at every point of the surface. Two-dimensional skeletons, consisting of triangles, provide a better solution to this problem because they approximate better the different distances between bones and the surface. However, creating such a skeleton is a difficult task for an untrained user.

These skeletal structures should allow the animation of joints to generate a credible simulation of extremities like arms or legs. While direct surface interaction methods [WAB*06, CCO02], simulate a pushing or pulling force on the elastic surface, objects with joints are usually animated by moving the skeletal representation of a joint section. The surface behavior is animated by moving the bones, not the surface.

In this work we focus on an interaction technique for animating hand puppet-like elastic objects in real time. We propose to automatize the generation of a two dimensional skeletal surface to allow direct interaction, and to use a mass spring system for simulating the dynamics on the object surface. We chose a mass spring model to reduce the number of elements to be computed in real time to something proportional of the number of polygons of the surface. A finite element method for the simulation of the physics would have signified a much higher complexity. The two-dimensional skeleton is created automatically from the original object by using a medial surface-based algorithm. The medial surface is the set of points that describe the geometric center of a 3D object [YBS03, HBK02]. These points are connected by using the original connectivity of the mesh, thus creating a skeletal mesh. By applying a mass-spring system that connects the skeletal mesh with the original mesh, the physical behaviour of the object can be simulated. As a welcome plus, material properties such as stiffness can be modified in a user-friendly way by using a skin thickness parameter. This is very similar to the behaviour of real latex puppets, whose stiffness ultimately depends on the thickness of their skin layers.

Movement is achieved by appointing weights to the vertices of a polygonal object and making the edges behave like springs. The properties of the material can be changed at runtime by applying the thickness parameter to the simulation. The whole object or just parts of it can be moved by the dynamics simulation.

This paper is organized as follows: Part 2 describes which previous work has been done on the topic of simulation of elastic surfaces or medial surface based applications. Part 3 illustrates the general principes of our approach by explaining the individual steps that are needed for animation of an elastic object. This consists of the skeleton generation, the construction of a mass-spring system, the modeling of different material properties and the creation of joints. Section 4 shows the results that were obtained, and Section 5 provides informations on the extensions of this approach.

2. Related Work

This work combines two well-known technologies: the first one is a mass-spring based real-time simulation of elastic surfaces. The animation is generated by user interaction with a skeleton, which in turn is generated by the second technique, the medial surface. This section will overview the related animation techniques.

Terzopoulos [TW88] introduced a reference object having the shape of the original to control the deformation of an object and to ensure shape recovery. However, his approach does not support the idea of joints, that could have been simulated with a skeleton-like reference. Likewise, indirect manipulation techniques such as [DDCB01, KR04, WAB*06, CCO02] do not support the animation of extremities.

Skeleton based animations not only offer the possibility of joint animation, they also provide an intuitive and direct way of deforming an object completely. For skeleton generation, a medial axis based algorithm is applied on the original object. The medial axis, and its two-dimensional counter part, the medial surface, have been used for different purposes. The medial axis in 2D first was used by Blum [Blu67, Blu73] for biological shape analysis.

Yoshizawa [YBS03] applies a free-form deformation on the medial surface of an object. The object surface is reconstructed from the position of the skeletal vertices. This method results in realistic large scale object deformations. The drawbacks of this approach are that the physical properties of the material are not considered and the application does not deliver results in real-time.

Gagvani [Gag01] animates a volumetric human dataset [NLoM] by generating the skeleton with the medial surface. He uses motion capturing to animate different joint regions and reconstructs the volume by placing voxeled *medial spheres* around the skeletal voxels. Combining these two technologies results in convincing animations. However, because of the use of volumetric data and the calculations needed for it, this approach is not computable in real-time yet.

Baran [BP07] animates a 3D character by extracting a medial surface approximation from a polygonal object. A reduced stick figure skeleton is generated from the medial surface points. The object can be animated by moving the stick figure using *linear blend skinning* (LBS). This method provides realistic animation results in real-time, but the physical properties of the animated object are not considered.

Corso [CCO02] calculates the medial surface of an implicit surface, and uses a mass-spring system to simulate real-time collisions between the object and a haptic device. The optical results look realistic, but the usage of implicit surfaces needs a large amount of computing power. Also, the animation of joints is not possible with this approach.

Conti [CKB03] calculates the medial axis from a given polygonal object with a decomposition algorithm. The resulting skeleton edges are stiffened by using parameters for flection, torsion and elongation, and the connection points inbetween behave joint-like. The surface is connected to the skeleton, and by applying force to the surface the skeleton and the joints are moved. The results of this method reach real-time rendering rates, however, due to the one-dimensional skeleton the simulation generates homogenous elasticity attributes. Furthermore, the position of the skeleton connection points, which resemble the joints, is extracted automatically by the medial axis generation. Therefore the subdivision of the object cannot be interactively done by the user, which in some cases may lead to unrealistic positioning of the joints of the skeleton.

Most of the previous work discussed is unable to simulate the surface of a 3D model exactly, some other approaches don't permit direct interaction with the object and some rely on the user's input for skeleton generation, which can lead to bad simulation results.

3. Approach

This chapter describes our method for applying elastic real-time simulation to a given polygonal object. Our approach for elastic object animation is divided into four basic steps: the skeletonization, the generation of the mass-spring system, the modeling of materials and the creation of joints. Fig. 1 illustrates the steps of the method.

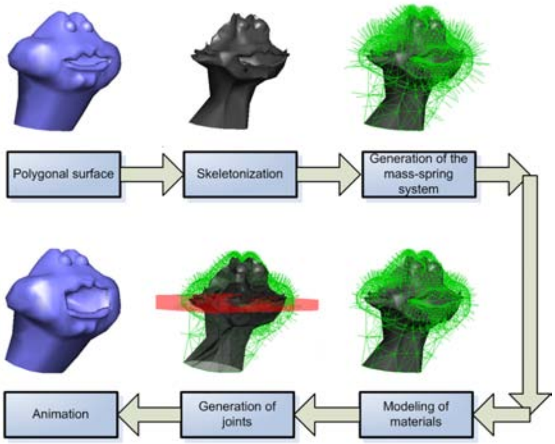

Figure 1: *There are four essential steps from a polygonal model to animation: the skeletonization, the generation of the mass-spring system, the modeling of materials and the creation of joints.*

3.1. Skeletonization

The skeletonization of the polygonal surface is done using a Voronoi-based algorithm for medial surface extraction. We use a medial surface based approach to have non-equidistant distances between the surface and the skeleton of the object for more realistic physical performance. We hereby take advantage of the fact that the medial surface is highly sensitive to surface curvature. As a result of this, the skeleton is generated closer to the surface in regions of high curvature than in those of low curvature. This correlation allows us to generate a mass-spring system with non-homogenous elasticity attributes, that generates less physical movement in areas of high curvature or sharp edges. The method we used is a modified adaption from Yoshizawa's work [YBS03], which calculates a skeletal mesh featuring a one-to-one correspondence with the polygons of the original mesh. The construction of the mass-spring system later on will be faciliated due to this fact.

The process is initiated by computing the Voronoi cells V_i set of from the original set of points P_i of the mesh. The operation results in a partition of 3D space, and each surface point has exactly one corresponding Voronoi cell. In a convex object these cells partition the inside of the volume of the object and meet at *internal Voronoi points* VP_j. If the object is locally concave, some Voronoi points also lie outside of the object, the *outer Voronoi points*. The set of skeletal points SP is computed only from the set of internal Voronoi points, which are extracted by checking a Voronoi point's position against every face normal of the surface that is within the Voronoi cell. If no points satisfy this condition, i.e. if the surface curvature is very high in a very unregular local topology, the surface point is used instead. The one-to-one correspondence between the set of original points and skeletal points is achieved by first calculating the average distance d of the remaining internal Voronoi points to the original point of one Voronoi cell. The skeletal point sp_i of a surface point p_i is then placed on the negative surface normal n_{p_i} in the point p_i at the distance d. Figure 2 shows the skeletal point generation in a two-dimensional diagram. In this example, the skeletal point sp_i is placed on the negative surface normal of p_i , using the arithmetic mean d of the euclidean distances of the two Voronoi points vp_1 and vp_2 to the surface point p_i as a distance factor.

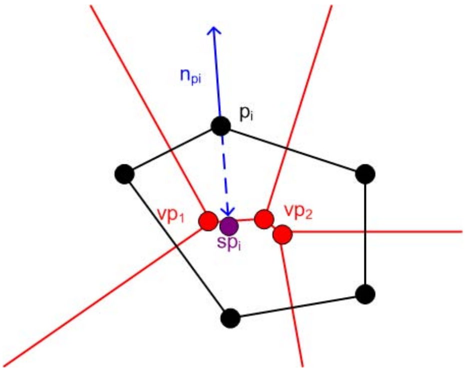

Figure 2: *Skeletal point generation: the skeletal point sp_i is placed on the negative point normal of p_i using the average distance of the Voronoi points vp_1 and vp_2 to p_i.*

By applying this method to an orginal mesh M we get a skeletal mesh S whereas the correlation

$$M = S + dN$$

is obtained. Here N is the field of point normals of the set of points P of the original mesh, and d is the curvature sensitive distance vector between the skeletal mesh S and the original mesh M. This rule is only suspended when the surface point was used as reference, which occurs only in areas of very high curvature. Also, the same connectivity as in the orginal mesh M is given in the skeletal mesh S. By using the point normal for skeletonization we achieve a good physical basis for the skeleton, onto which the mass-spring system is attached to simulate the elastic surface.

3.2. Generation of the mass-spring system

A mass-spring system is a model for the elastic simulation of surfaces. It consists of a fixed topology of springs interconnecting mass-points, which represent weights. By simulating the springs and applying a force to the mass points, the system can be used for animation of dynamical processes. We use a mass-spring system based on the suggestions of Terzopoulos [TPBF87, TF88]. To make this system solvable in real-time, the dynamics are not calculated analytically, but by approximating the behavior of the mass points. This is done by using numerical integration methods, which can be distinguished in *explicit* and *implicit* methods. While explicit methods are fast but unstable, implicit methods are computationally expensive. As the application we work at requires the simulation of complex surface detail, a computational tradeoff between rendering surface materials and physical simulation has to be made. We have chosen to use the explicit method of Verlet for our application, because it presents a fast and stable alternative to implicit methods in context with constraints.

Furthermore, even today implicit methods tend to be to slow when mass-spring systems have a size of more than 10000 springs [BW06], thus limiting the size and level of detail of the simulated object. Further material on the general principles of numerical integration combined with mass-spring systems can be reviewed in publications such as [BW98, BW97].
Mass-spring systems and numerical integration methods have already been used for cloth simulation [BW98, MDDB00] as well as for medical simulations [CCO02, GCMS00, ALC99]. Only recently, this method was used for simulation of grass and fur [BW06] and for the animation of hand puppets [WAB*06].
As our method requires the surface of the object to be elastic, the surface is transformed into a mass-spring system, whereby the edges of the polygons are changed into *surface springs*, and the surface points into mass points. Despite the works in real-time animation that were realized with FEMs [DDCB01, KR04] we favoured mass-spring system over finite element methods because of their computational complexity. A stabilizing basis for our mass-spring system was built by the skeleton, which already features a one-to-one point correspondence to the surface mesh. We exploit this correlation by generating an *inner bone point* bpi_i at the position of the skeletal point sp_i, and by connecting them via a *bone spring*, thus establishing the connection between the skeleton and the surface. *Bone points* in this context resemble mass points, but their positions are not affected by physical simulation. A *bone spring* connects a *bone point* to its mass point on the surface.
This model of a mass-spring system does not deliver simulation results of the thickness and shape preserving forces of a material, and, because of this, is unsuitable for effects like gravity. Internal skeletal structures do not provide enough information for defining a rest position in a mass spring system. Thus, shape stabilizing forces cannot be simulated with this model of a mass-spring system. To resolve this problem, we place another *outer bone point* bpo_i on the outside of the surface, which acts as a counterbalance for the *inner bone point*. This approach is related to the idea of Kahler [KHS01]. The position of this point is calculated by defining the point reflection of the skeletal point sp_i at the point p_i (see fig. 3). The resulting two layers of inner and outer *bone springs* deliver maximum physical influence in orthogonal direction to the surface so that they act as a shape stabilizer for the object. Skeletal influence on the movement of the mass points is achieved by varying the length of the *bone springs*.

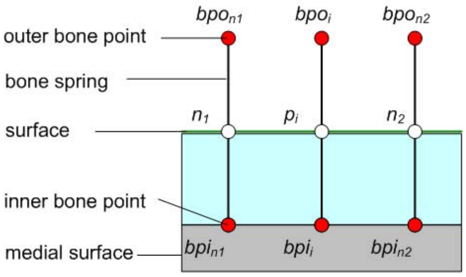

Figure 3: *One inner bone point bpi_i and one outer bone point bpo_i is generated. The inner bone point bpi_i is positioned at the skeletal point sp_i, the outer bone point bpo_i is a point reflection of the skeletal point sp_i on its corresponding surface point p_i.*

We apply a velocity damping coefficient to the system to ensure loss of energy. The surface of the object now can be animated by moving the skeleton with a mouse or a data glove. According to the movement of the skeletal points, the *bone points* are transformed, which results in a dynamic animation of the surface points by simulating the mass-spring system.
However, whenever strong forces are applied to this setup, an overly elongation of the *bone springs* results, which gives the physical simulation an unreasonable, "‘collapsing"’ appearance. This is because the mass-spring system lacks shape stabilizing forces and the inertia of the mass points.

We have implemented two methods that constrain the movement of the mass points, the *circumsphere method* and the *spring reduction method*, to solve this problem. Both methods are applied to the bone springs, as they have most influence on the shape of the surface.
The *spring reduction method*, which is based on [Pro95],

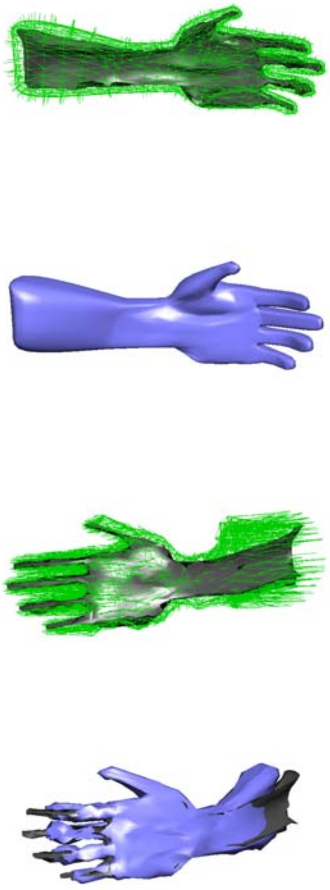

Figure 4: *A hand model in resting position and after application of strong forces. The bone springs get too much elongated with negative effects on shape preservation. As a result the skeleton (shown in gray) coincides with the surface.*

constrains the extension/contraction of a spring to a maximum/minimum level dependent of the rest length of a bone spring. That means that a *bone spring* with the constraint parameters of maximum length $f_{max} = 1.5$ and minimum length $f_{min} = 0.8$ can be stretched to an extend of 150% and compressed to a maximum of 80% of its rest length. The end of a *bone spring* which ends in the mass point is shortened towards the *bone point*, or elongated along the spring's direction vector. Because of the structure of our mass-spring system which features two *bone springs* for one mass point, an operation on one spring can result in the transgression of the other spring's constraints. As a result, this technique has to be applied multiple times to the set of *bone springs* per calculation cycle. This technique results in a natural behavior of the surface, but is computationally more expensive.

We also developed the *circumsphere method*, which constrains the movement of a mass point to the circumsphere around its rest position, to improve performance. The size of the sphere is dependent of the rest length of the bone spring. For example, if a maximum movement factor of $f_{max} = 1.5$ is used for a mass point p_i, a constraint sphere with the radius $r_i = f_{max} * L_{bs_i}$ is generated. L_{bs} defines the rest length of a *bone spring* of p_i. If a mass point's position at a time $t + h$ exceeds the volume that is defined by the constraint sphere, the point is simply placed on the sphere's boundary along the distance vector to its rest position. This step is taken after the physical computations and therefore generates a stable simulation basis. Although this constraint metaphor does not base upon realistic physical calculations, it shows good results in simulation praxis and needs less computations.

Both techniques have a slightly different effect on the object when the same maximal spring factor f_{max}, the (see fig. 5) is applied.

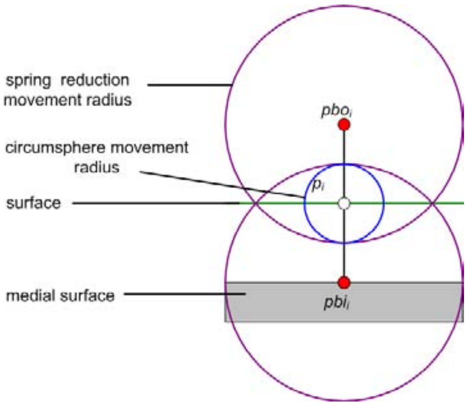

Figure 5: *Comparison of the two constraint methods at a maximum spring factor* $f_{max} = 1.5$ *: The spring reduction method's movement space, which is defined by the intersection of the violet circles, allows more surface tangential movement than the circumsphere method's (blue).*

The *spring reduction method* allows a bigger space for movement of the mass point in tangential direction of the surface than in orthogonal. The *circumsphere method* allows the same amount of movement in all directions. The usage of the *spring reduction* method has the advantage of a better optical impression especially in joint regions (see 3.4), while there is a noticeable performance increase when the *circumsphere method* is used. Because the usability of the two methods is dependent on the surface topology of the model, the user is able to choose between the two of them. Elasticity issues suggest the usage of the *circumsphere method* for models with big surface triangles.

Both techniques apply an additional damping variant to the mass-spring system by constraining the dynamics, which acts also as a shape stabilization to the object. This improves the physical stability properties of a simulated object, but also enables the user to define material properties by changing the minimal and/or maximal *bone spring* length.

3.3. Modeling of materials

By using the methods that were described in the previous paragraphs, we achieve a stable representation of an elastic object, which can be physically simulated by rotating and translating the *bone points*. By modifying the parameters of the constraints and the mass-spring system, different elastic behaviors can be simulated.

Further improvement of the simulation results can be achieved by scaling the distance from the inner and outer *bone points* to their surface point. By changing the *bone distance* factor f_{bd} to a number below 1.0, the position of the *bone points* is shifted away from their skeletal points towards the surface. This technique leaves the skeletal points only as a reference for the position of the *bone points* whose distance to the surface can be adjusted by the user to simulate the "‘thickness"’ of a material. When this variation of the *bone distance* is applied, the rest length of the *bone springs* is modified, which results in different levels of elasticity of the object (see fig. 6). The shorter the *bone springs* are, the firmer the simulated material is. Note that while the *bone distance* factor is applied to all *bone springs* of a model, the relation factor of the lengths of all *bone springs* to each other stays the same, leaving the overall elastic properties within the object the same. That means that locally more elastic parts of an object with a *bone distance* factor of $f_{bd} = 0.8$ stay also more elastic than stiffer parts with a *bone distance* factor of $f_{bd} = 0.4$. A great variety of materials can be simulated using the parameterization of the constraints and the scaling of the *bone distance*.

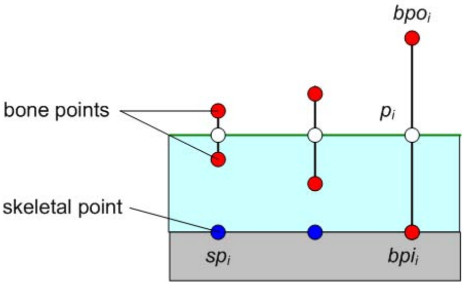

Figure 6: *The variation of the bone distance is used for simulating different attributes of materials. From left to right a the bone distances of 0.25, 0.5 and 1.0 are shown. The skeletal points act as a reference for the position of the bone points*

3.4. Creation of joints

The final step for realistic animation of a model is the creation of joints. To arrange the objects mass points into different joint sections, we use a plane to cut the skeleton of the object. The *bone points* are classified in reference to their corresponding skeletal points, and the generated bone parts are managed in a binary tree. To achieve a credible animation in joint regions, the user has to define a rotation point for each cut that is made, which is used as rotation center for the two generated joint sections. This is usually done by placing the rotation point on the cutting plane. After the joint creation process, the user can freely animate any of the joint sections by choosing and rotating it appropriately. It is also possible to animate more than one joint section at the same time. As the surface springs are shortened or elongated by joint animation, the masspoints in joint-near regions deform accordingly. However, by applying the *spring reduction method*, the masspoints have a bigger radius of movement tangential to the surface, and thus behave more naturally than with the *circumsphere method*. In practice both methods deliver a similar deformation to hand controlled puppets.

4. Results

The method described in this paper was implemented on a Windows work station using OpenGL as graphics API and the MS Visual Studio 2003 IDE. The performance of the application was tested on a Pentium M 1.4 GHz notebook system with a ATI 9700 graphics board (64 MB) and 512 MB of main memory, and on a high-end desktop pc equipped with a Pentium IV D 3.0 GHz CPU, a GeForce 7800 GT graphics board (256 MB) and 2 GB of main memory. All performance results shown here were rendered using the *circumsphere method* as constraint. On the laptop system, realtime interaction ($\approx$ 25 fps) is possible with objects containing up to 12000 springs, on the high-end pc up to 17000 (see fig. 7). At interactive frame rates ($\approx$ 16 fps) objects with 17000 (laptop system) respectively 26000 springs (high-end pc) can be rendered.

The number of faces that can be visualized with this number of springs is dependent on the topology of the surface mesh, ranging from about 4500-5500 faces (laptop system) to 7000-8000 faces (high-end pc). Automatic skeletonization of objects featuring this size does not take a considerable amount of time (1 - 2.5 seconds), and has the big advantage of being independent of potentially incorrect user input.

The visual performance of the object resembles the one of cloth or jelly, dependent on the parameters of the mass-spring system, while the thickness of the material can be adjusted to match the desires of the user. The following factors can be changed at runtime by the user.

The maximum spring length sl_{max} defines the area of movement of the mass points. The bigger sl_{max} is chosen,

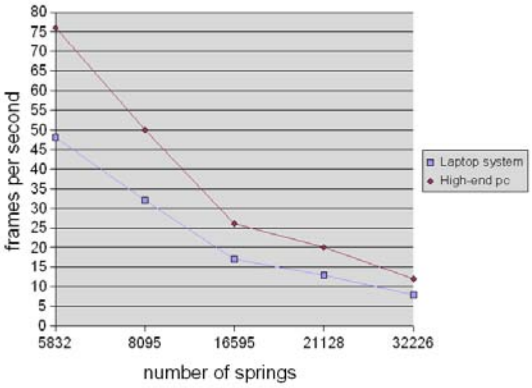

Figure 7: *The performance on a laptop system and a high-end pc.*

sl_{max}	maximum spring length
D	spring constant
C_{vd}	velocity damping coefficient
d_B	*bone distance*

the bigger the area of movement, and thus, the moving range of the object surface increase. D is the spring constant, which defines the oscillation behavior of the surface. A high factor of D reduces the inertia of the mass points. The velocity damping coefficient C_{vd} has influence on the energy absorption rate of the system, which visualizes a more jelly-like behavior with a high value, and more cloth-like with a low value. Finally, d_B specifies the factor of the distance from the bone to the surface, making the surface of the object either look thick or thin. The maximum spring length sl_{max} and the bone distance factor d_B have most influence on the physical behavior of the model, because both together assign the area of movement for the mass points.
The selection of these two variables allows the simulation of a large amount of different materials, which can be easily defined even by an untrained user. Table 1 shows a selection of materials that can be simulated. Positions marked with *var* in the table indicate that these parameters have no influence on the behavior of the surface, because either the maximum spring length sl_{max} or the *bone distance* d_B constrain the system to be non-dynamic. A *bone distance* of 0.0 means that the bone is set on the surface, while a maximum spring length of 1.0 constraints the system to have inelastic springs, de facto allowing no elastic movement.

surface model	C_{vd}	D	d_B	sl_{max}
thick skin	0,7	1,0	0,5	1,3
water	0,99	1,0	1,0	1,4
plastic	var	var	0,0	1,0
skin	0,85	1,0	0,2	1,25
tight cloth	0,85	0,2	0,4	1,35

Table 1: *Examples of parameters for visualization of different materials. "‘var"’ means that these parameters have no influence on the behavior of the surface, because the values of d_B and sl_{max} constrain the system to be non-dynamic*

By rotating the different joint sections the object can be animated. Rotation of multiple sections is also possible and allows the real time simulation of differentiated motion activities such as the synchronous movement of the jaws of a model for speaking. Also, by using a mass-spring system for surface simulation, the material generates a fold on the inner part of the joint (see fig. 8). The nature of the simulation allows the application for areas like the animation of latex or rubber puppets or skin simulation. Unlike the work of Wüthrich *et. al* [WAB*06], which uses an elastic skeletal structure, the application proved to be stable for all kinds of user input, when one of the two methods of constraints are applied to the mass-spring system. This affects also the normally instable numerical integration methods, by preventing situations that could lead to the destruction of the mass-spring system. This was implemented by specifying a particular area of movement for every mass point. Also, in contrast to direct manipulation of the surface, the user interaction method was improved by using a skeletal movement metaphor.

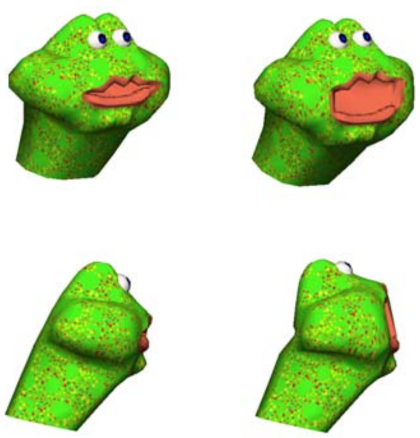

Figure 8: *A frog model being animated in the jaw region. A fold is created on the back of the model when the upper jaw is rotated by a big angle*

Figures 9 and 10 show some snapshots of the animation of skeletal objects such as a cow or an arm.

5. Conclusion

We found a straightforward way to simulate the physical properties of the surface of an object realistically. Unlike many other approaches, a physical simulation of a closed polygonal object can be realized within seconds without previous knowledge and preliminary work. This is achieved by automatizing the skeletonization process and by introducing the metaphor of a cut through the skeleton for joint creation. Many different materials can be simulated by using the *bone distance* factor and the spring constraint metaphors. By applying the latter to the mass-spring system, also shape stabilization can be achieved. This allows the user to change material properties at runtime and simulates different elasticity characteristics for the objects. By using a medial-surface based skeletal representation, the application delivers detailed and non-homogenous simulation results. These attributes make this method suitable for the interactive simulation of hand-operated elastic puppets. The method can be used for the simulation of a puppet head, but also for the puppet's extremities. Also, the method allows very natural simulations when the puppet is manipulated through a data handglove.

Further improvements in surface behavior could be achieved by defining a set of *bone points* in the adjacency of joints as a group of *joint points*, which move at an interpolated rate between the two joint sections. This would permit a gradual deformation of the surface that should result in a more realistic effect.

A yet-to-be-solved issue in our method is the well-known collapsing ellbow problem. Collapsing could be prevented by applying one or more existing approaches to the joint-near *bone points*. Approaches that already take advantage of medial surface systematics such as the works of Bloomenthal [Blo02] could be applied. Also, implementation of wrinkels could further improve the visual results if virtual objects with a more skin-like surface are to be animated.

An intra-object collision detection at the joints based on the *medial balls* which was already introduced by Hubbard [Hub96] could also improve the realism when big rotations are applied to the joint sections.

Different materials on the same object could be rendered by using multiple *bone distances* within one object. The surface points would be grouped in different material classes which assign the surface attributes to a certain region of the object.

A mass-spring system as suggested by d'Aulignac [ALC99] with nonlinear *bone springs* could improve the visual performance and overall impression of the simulated object. Joint generation could be further enhanced by supporting different metaphors for skeleton cutting, like a cuboid or a sphere.

Acknowledgements

The authors would like to thank Sven Banisch and Przemyslaw Musialski for the useful discussions. Furthermore, the authors would like to thank Chrystoph Toll for the precious help given in reviewing this paper. This work would not have been possible without the ideas and consulting of Jing Augusto of the ABS-CBN Foundation in Quezon City, Philippines. This work is part of a cooperation between the Bauhaus-University Weimar and the ABS-CBN Foundation.

References

[ALC99] AULIGNAC D., LAUGIER C., CAVUSOGLU M.: Modeling the dynamics of a human thigh for a realistic echographic simulator with force feed-back, 1999.

[Blo02] BLOOMENTHAL J.: Medial-based vertex deformation. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2002), ACM Press, pp. 147–151.

[Blu67] BLUM H.: A Transformation for Extracting New Descriptors of Shape. In *Models for the Perception of Speech and Visual Form*, Wathen-Dunn W., (Ed.). MIT Press, Cambridge, 1967, pp. 362–380.

[Blu73] BLUM H.: Biological shape and visual science (part I). *Journal of Theoretical Biology 38* (1973), 205–287.

[BP07] BARAN I., POPOVIĆ J.: Automatic rigging and animation of 3d characters. *ACM Trans. Graph. 26*, 3 (2007), 72.

[BW97] BARAFF D., WITKIN A.: Physically based modeling: Principles and practice, 1997.

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. *Computer Graphics 32*, Annual Conference Series (1998), 43–54.

[BW06] BANISCH S., WÜTHRICH C. A.: Making grass and fur move. *Proceedings of WSCG 2006* (30 January–3 February 2006).

[CCO02] CORSO J., CHHUGANI J., OKAMURA A.: Interactive haptic rendering of deformable surfaces based on the medial axis transform. *Eurohaptics* (2002), 92–98.

[CGC*02] CAPELL S., GREEN S., CURLESS B., DUCHAMP T., POPOVI Z.: Interactive skeleton-driven dynamic deformations. *To appear in the Proceedings of SIGGRAPH 2002* (2002).

[CKB03] CONTI F., KHATIB O., BAUR C.: Interactive Rendering Of Deformable Objects Based On A Filling Sphere Modeling Approach. *IEEE International Conference on Robotics and Automation* (2003).

[DDCB01] DEBUNNE G., DESBRUN M., CANI M. P., BARR A. H.: Dynamic real-time deformations using space & time adaptive sampling. In *SIGGRAPH 2001, Computer Graphics Proceedings* (2001), Fiume E., (Ed.), ACM Press / ACM SIGGRAPH, pp. 31–36.

[Gag01] GAGVANI N.: Parameter-controlled skeletonization - a framework for volume graphics. *A Framework For Volume Graphics, Ph. D. Thesis* (2001).

[GCMS00] GANOVELLI F., CIGNONI P., MONTANI C., SCOPIGNO R.: A multiresolution model for soft objects supporting interactive cuts and lacerations. *Computer Graphics Forum 19*, 3 (2000).

[HBK02] HISADA M., BELYAEV A., KUNII T. L.: A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Computer Graphics Forum 21*, 4 (2002), 1–12.

[Hub96] HUBBARD P. M.: Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics 15*, 3 (1996), 179–210.

[KHS01] KÄHLER K., HABER J., SEIDEL H.-P.: Geometry-based muscle modeling for facial animation. In *GRIN'01: No description on Graphics interface 2001* (Toronto, Ont., Canada, Canada, 2001), Canadian Information Processing Society, pp. 37–46.

[KR04] KONDO R. K. T.: Interactive physically-based animation system for dense meshes. In *In Proc. Eurographics 2004 short paper (2004)* (2004), pp. 93–96.

[MDDB00] MEYER M., DEBUNNE G., DESBRUN M., BARR A. H.: Interactive animation of cloth-like objects in virtual reality. *Journal of Vizualisation and Computer Animation* (2000).

[MTCK*04] MAGNENAT-THALMANN N., CORDIER F., KECKEISEN M., KIMMERLE S., KLEIN R., MESETH J.: Simulation of clothes for real-time applications. In *Eurographics 2004, Tutorials 1: Simulation of Clothes for Real-time Applications* (September 2004), INRIA and the Eurographics Association.

[NLoM] NATIONAL LIBRARY OF MEDICINE U. S.: The visible human project. *http://www.nlm.nih.gov/research/visible/visible_human.html*.

[Pro95] PROVOT X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Graphics Interface '95* (1995), Davis W. A., Prusinkiewicz P., (Eds.), Canadian Human-Computer Communications Society, pp. 147–154.

[TF88] TERZOPOULOS D., FLEISCHER K.: Modeling inelastic deformation: viscolelasticity, plasticity, fracture. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1988), ACM Press, pp. 269–278.

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1987), ACM Press, pp. 205–214.

[TW88] TERZOPOULOS D., WITKIN A.: Physically based models with rigid and deformable components. *IEEE Comput. Graph. Appl. 8*, 6 (1988), 41–51.

[WAB*06] WÜTHRICH C. A., AUGUSTO J., BANISCH S., WETZSTEIN G., MUSIALSKI P., TOLL C., HOFMANN T.: Real time simulation of elastic latex hand puppets. *Proceedings of WSCG 2006* (30 January–3 February 2006).

[YBS03] YOSHIZAWA S., BELYAEV A., SEIDEL H.-P.: Free-form skeleton-driven mesh deformations. *Proceedings: 8th ACM Symposium on Solid Modeling and Applications (SM-03)* (2003), 247–253.

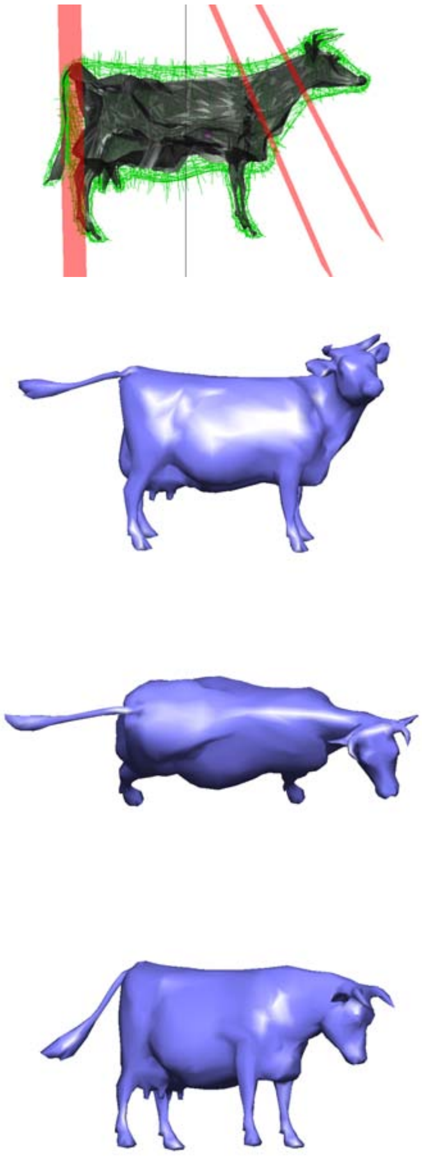

Figure 9: *A cow model being animated to lift and bend its head.*

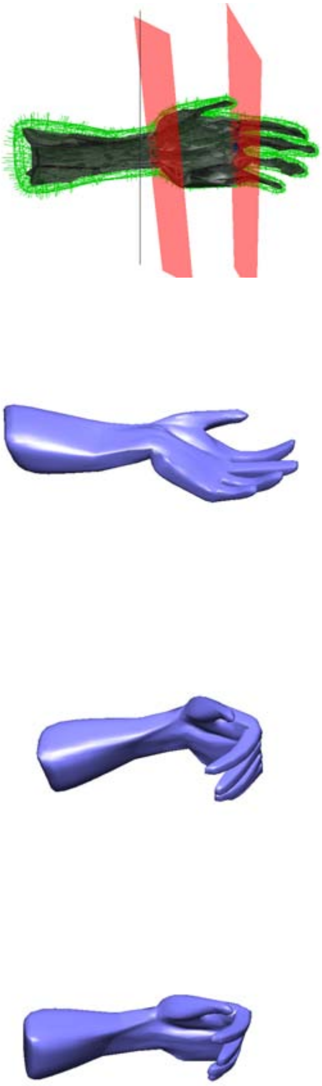

Figure 10: *A hand model animation.*