A Kinematic Model for Collision Response

Jason Harrison David Forsey

Department of Computer Science, U.B.C
{drforsey,harrison}@cs.ubc.ca

Abstract

One aspect of traditional 3D animation using clay or plasticine is the ease with which the object can be deformed. Animators take for granted the ability to interactively press complex objects together. In 3D computer animation, this ability is severly restricted and any improvement would drastically increase the range and style of animations that can be created within a production environment.

This paper presents a simple, fast, geometric approach to controlling the nature, extent and timing of the surface deformations arising from the interpenetration of kinematically controlled animated objects. Rather than using dynamic simulations, which are difficult to configure, code, and control, the algorithm presented here formulates collision response kinematically by moving points on a multiresolution surface towards goals points at a certain rate.

This new multi-resolution approach to deformations provides control over the response of the surface using a small number of parameters that determine how each level in the multi-resolution representation of the surface reacts to the interpenetration. The deformations are calculated in linear time and space proportional to the number of points used to define the surface.

Keywords: Animation, collision, deformation, inverse kinematics, modeling, surface.

1 Introduction

When confronted with a situation where two or more polygonal surfaces interact an animator must usually report to painstaking manipulation of individual vertices in the models, or indirect methods such as free-form deformations [sede86]. In existing commercial modeling and animation systems there is very little support for operations involving interactions between surfaces. A simple to use, interactive tool providing this functionality would dramatically increase the range and style of animation produced in a commercial production environment.

Commercial animation companies usually avoid the use of dynamics. Such simulations are expensive to run, prone to numerical instabilities, often difficult to configure because the behavior of the surface is an emergent property of its physical parameters making it difficult to determine how to set the parameters to mimic a particular behavior.

Our approach to calculating the surface deformation induced by an interpenetration is geometric, supporting the traditional methods of kinematic motion specification. Control is provided so that the nature, extent and timing of the deformation is controllable, the surface response can vary from highly complaint, perfectly elastic, local reactions, to rigidly inelastic global reactions.

Here we define a collision as the mutual penetration of two or more surfaces. Collision response is the *effect* of a collision, and for an animated surface the time dependent deformation of the surface that resolves the penetration.

As a kinematic solution, the method presented is numerically stable, and temporally and spatially compact in its use of resources. The displacement of contact points is arbitrary, by formulating their motion as the result of a hierarchy of vertex and reference offset methods [fors88] the response of the surface is controlled by a number of parameters directly proportional to the number of levels in the hierarchy.

2 Related Work

Reproducing aspects of the real-world is a major focus of research in computer graphics. The behavior of deformable materials is just one example of a class of models that attempt to increase the realism of computer generated animations [terz87] [chad89] [mill88] [witk90].

Whatever their value as simulations, one factor that has prevented the widespread use of these models as animation tools is the lack of precise control over the geometric and temporal behavior of the surfaces. With these models it is difficult to specify forces, constraints, damping factors, mass distribution, spring constants, elasticity, yield limits, and interconnections required to change the system in a particular, precise way.

This situation is perfectly reasonable if the goal is to determine how the simulated system behaves under varying conditions (this is typically the reason for building a simulation in the first place). However, this behavior may not be suitable in situations where animators demand complete control over the principle characters and are uninterested in physical validity unless it directly affects a viewer's appreciation of a scene

3 Models of deformable surfaces

The creation of a deformable surface model for animation begins, like all things in computer science, with an underlying model. Proposed models for deformable surfaces have generally used either a derivative of the thin-plate model, or a finite-element model. Although a few authors have proposed the application of elasticity theory (see references below).

Surface models for free-form modeling have also been animated. Recent models have generally either introduced new surface representations, the application of geometric constraints, or the use of the thin-plate model to fair (smooth) the surface.

3.1 The basis-functions approach

The computational resources available to an interactive algorithm are bounded by the today's technology. This realization, coupled with the difficulty of manipulating "traditional" free-form surfaces (*e.g.*, Bezier, B-spline, Catmull-Rom), has driven the creation of new basis-functions and data-structures to represent free-form surfaces [fors88] [fink94], and interactive manipulation methods for controlling the shape of the resulting tensor product surfaces [fowl92] [celn92].

Free-form surfaces have been animated by specifying time dependent changes in the values of the underlying surface controls. For example, Weil's geometric model for cloth [weil86] has been animated by specifying the motion paths of the control points. By moving the control points and then applying the relaxation step between animation frames, the "quality" of the resulting animation is due primarily to the motion of the control points. However, most surface models do not incorporate a relaxation phase, and may, depending upon the surface representation, produce extraneous creases, wrinkles, and wiggles as the surface is deformed unless explicitly smoothed.

3.2 Energy-based functionals

To fair a surface, removing unwanted wrinkles, the *thin-plate model* has been used. It attempts to measure the stretch and bending energies of an ideal thin-plate. energy of a surface. It, or its linearization, is used by a variety of researchers outside the area of computer animation (*e.g.*, for visual surface reconstruction from incomplete or noisy data [terz88a]). The functional is:

$$\iint_{u,v} \left(\|\mathrm{G}\|_\alpha^2 + \|\mathrm{B}\|_\beta^2 \right) \, \mathrm{d}u \, \mathrm{d}v \tag{1}$$

where G and B represent the first and second fundamental surface forms [fari88]:

$$\begin{aligned} \mathrm{G}(u,v) &= {\mathbf{x}_u}^2 + 2\mathbf{x}_u\mathbf{x}_v + {\mathbf{x}_v}^2 & (2) \\ \mathrm{B}(u,v) &= \mathbf{x}_{uu} + 2\mathbf{x}_{uv} + \mathbf{x}_{vv}, & (3) \end{aligned}$$

where the partial derivatives $\frac{\partial \mathbf{x}}{\partial u}$ and $\frac{\partial^2 \mathbf{x}}{\partial u^2}$ are written $\mathbf{x}_u$ and $\mathbf{x}_{uu}$ respectively, and $\mathbf{x}$ is a contiguous set of points in 3-space parametrized by u and v.

This functional is highly nonlinear in the vector and matrix norms, and leads to a difficult nonlinear optimization problem. It is therefore common [welc92] [cari92] [celn91] [terz87] to simplify the functional by linearizing the matrix norms and B to produce the *thin plate under tension* model [schw66]:

$$\iint_{u,v} \left(\|\mathrm{G}\|_\alpha + \beta_{11}{\mathbf{x}_{uu}}^2 + 2\beta_{12}{\mathbf{x}_{uv}}^2 + \beta_{22}{\mathbf{x}_{vv}}^2 \right) \, \mathrm{d}u \, \mathrm{d}v. \tag{4}$$

This approximation is only accurate near the actual minimum — physically it is only valid if the deformation is less than the thickness of the plate — but it is well behaved away from the minimum. Although used in animated sequences to control deformations orders of magnitude greater than the underlying theory allows, the simplification reduces the cost of minimizing the objective functional. For a linear surface representation, such as a tensor product B-spline, Equation 4 is quadratic in the underlying surface degrees of freedom, and the optimization problem can be cast as a constrained least-squares minimization. Minimizing the system can be done at interactive rates since the resulting matrices are banded and sparse.

Terzopoulos, *et al*, [terz87] used the thin-plate functional to measure the deformation energy of a surface from its natural shape:

$$\mathcal{E}(\mathbf{r}) = \iint_{u,v} \left(\|\mathrm{G} - \mathrm{G}^0\|_\alpha^2 + \|\mathrm{B} - \mathrm{B}^0\|_\beta^2 \right) \, \mathrm{d}u \, \mathrm{d}v, \tag{5}$$

where, $\mathbf{r} = \mathbf{x}(t)$. But they quickly approximated their functional with

$$\mathcal{E}(\mathbf{r}) = \iint_{u,v} \sum_{i,j} \left(\eta_{ij}(\mathrm{G}_{ij} - \mathrm{G}_{ij}^0)^2 + \xi_{ij}(\mathrm{B}_{ij} - \mathrm{B}_{ij}^0)^2 \right) \, \mathrm{d}u \, \mathrm{d}v, \tag{6}$$

where each of the parametric directions u and v are substituted for i and j. $\eta_{ij}(\mathbf{x})$ and $\xi_{ij}(\mathbf{x})$ are weighting functions. They approximated the first variational derivative $\delta\mathcal{E}(\mathbf{r})/\delta\mathbf{r}$ of Equation 6 by keeping only terms of the first order, giving the vector expression :

$$\mathrm{e}(\mathbf{r}) = \sum_{i,j} \frac{\partial}{\partial i}\left(\alpha_{ij}\frac{\partial \mathbf{r}}{\partial i} \right) + \frac{\partial^2}{\partial i \partial j}\left(\beta_{ij}\frac{\partial^2 \mathbf{r}}{\partial i \partial j} \right), \tag{7}$$

The constitutive functions $\alpha_{ij}(\mathbf{x}, \mathbf{r})$ and $\beta_{ij}(\mathbf{x}, \mathbf{r})$ describe the elastic properties of the surface:

$$\alpha_{ij}(\mathbf{x}, \mathbf{r}) = \eta_{ij}(\mathbf{x})(\mathrm{G}_{ij} - \mathrm{G}^0_{ij}), \tag{8}$$

so when α_{ij} is positive the surface wants to shrink in extent, and when α_{ij} is negative, it wants to grow. Since the second term in Equation 7 yields unwieldy expressions when the calculus of variations is applied to it, their alternative is to use by analogy to Equation 8:

$$\beta_{ij}(\mathbf{x}, \mathbf{r}) = \xi_{ij}(\mathbf{x})(\mathrm{B}_{ij} - \mathrm{B}^0_{ij}), \tag{9}$$

so when β_{ij} is positive the surface wants to be flatter, and when β_{ij} is negative the surface wants to be more curved. To simulate the elastic forces of the deformed surface, they discretized Equation 7 using finite differences to form equations approximating the stress and strain energies. By applying external forces to the resulting equations and numerically integrating through time, the behavior of a surface is simulated.

Celniker and Gossard [celn91] applied a weighted sum of continuous shape functions from finite-element theory to approximate the thin-plate under tension functional (Equation 4). This approximation was used to set the remaining degrees of freedom of triangular surface patches constrained by both geometric constraints (input by the sculptor) and a $G^{[1]}$ continuity constraint along the patch boundaries. Although their approximation only guarantees $C^{[1]}$ continuity they state that the surface will tend to be $C^{[3]}$ continuous. The resulting surface could then be modified interactively by changing the constraints, or the values in the matrix norms α and β.

Carignan, *et al*, [cari92] used the thin-plate model of [terz87] (Equation 6) with an added damping term from [plat88] to simulate the shape and motion of deformable clothes for synthetic actors. Interestingly, they formulated the animator's control over the physical model of the fabric by specifying only three values: its density, its percentage of elongation under gravity, and its resistance to bending. But the actual *behavior* of the clothes was determined by the simulation, not by the animator.

Moreton and Séquin [more92] minimized the *variation* in surface curvature using the functional:

$$\iint_{u,v} \left(\left(\frac{\mathrm{d}\kappa_0}{\mathrm{d}u} \right)^2 + \left(\frac{\mathrm{d}\kappa_0}{\mathrm{d}v} \right)^2 \right) \mathrm{d}u\,\mathrm{d}v, \tag{10}$$

where κ_0 is the normal curvature of the surface in the direction $\mathbf{s}$ (defined by $\mathrm{d}u/\mathrm{d}v$). The normal curvature is given by the second fundamental form divided by the first fundamental form:

$$\kappa_0 = \kappa_0(\mathbf{x}, \mathbf{s}) = \frac{\mathrm{B}(u,v)}{\mathrm{G}(u,v)} = \frac{\mathbf{x}_{uu} + 2\mathbf{x}_{uv} + \mathbf{x}_{vv}}{{\mathbf{x}_u}^2 + 2\mathbf{x}_u\mathbf{x}_v + {\mathbf{x}_v}^2}. \tag{11}$$

Thus Equation 10 integrates to zero for cyclides: spheres, cones, cylinders, tori, and planes. Their functional was used to fair triangular and quadrilateral patches fitted to interpolated geometric constraints consisting of positions and, optionally, surface normals and surface curvatures. As an initial guess for their surfaces, they used the connectivity of the geometric constraints to fit minimum variation curves, that were then used as the boundaries for the patches that composed the final surface. To their credit, rather than linearize their functional, they used nonlinear optimization to calculate the value of the gradient of the functional during the fairing process. This process can not be considered interactive given the computational power of today's workstations.

3.3 Combining rigid-body and deformable-body dynamics

The formulations presented above work well in practice for models that are moderately to highly deformable, but they become numerically ill-conditioned as the rigidity of the models is increased. By modeling objects using a combination of rigid-body and deformable-body dynamics, it is possible to make the surfaces stiffer than with a pure energy-potential functional.

Terzopoulos and Witkin [terz88c] introduced this technique and used linear elasticity theory to govern the dynamics of the deformable component. While the rigid reference-body handled rigid-body transformations, the deformable component reflected the deformation of the object from its rest state.

Immediately Terzopoulos and Fleischer [terz88b] extended the technique by evolving the reference component in response to the forces the surface is subjected to. This makes it possible to simulate viscoelasticity, plasticity, and fracture. Additionally, it allows stiffer objects and broadens the range of possible behaviors of the objects, but remains computationally expensive as the number of state variables easily reaches magnitudes in the tens of thousands. The application of multigrid methods to these systems has been attempted, but is difficult to correctly code due to the "irregularities" that evolve within a system undergoing irreversible deformations [terz88b].

3.4 Finite-element models

Finite-element models of deformable objects have been used to model surfaces, human skin, and even the motion dynamics of snakes and worms. As a discrete representation of continuous media, the three-dimensional lattices of springs and masses used in an FEM are computationally intensive. They provide the animator with objects that behave like jello: they wiggle, droop, and move about without the express direction of the animator.

Miller [mill88] quickly covers the small number of details involved in animating the motion dynamics of snakes and worms using an (admittedly) greatly simplified model of elastically deformable strands.

Chadwick, *et al*, [chad89] combine kinematic motion control with an elastic finite element model used to control a free-form deformation lattice [sede86]. They discuss several of the problems inherent in designing a system for animators, including motion specification, secondary motion, and the critical damping of the springs of to remove oscillations.

Gourret, *et al*, [gour89] used a finite element model to simulate the deformation of objects and human skin in a grasping task driven by kinematic motion control. To model the deformations around the joints of the human hand during flexion they used a modified set of phalangeal bones carefully shaped to avoid compressing the flesh model because they didn't have the computational power to simulate at interactive rates incompressible bones or non-linear material models for the flesh.

Platt and Barr [plat88] proposed the application of reaction constraints to control the motion of finite element models (path following), and augmented Lagrangian constraints to implement physical properties (incompressibility and moldability). These are the formal mathematical models that are usually implemented in an *ad hoc* fashion for FEM systems (see [lee93] for an *ad hoc* volume conservation method).

Thingvold and Cohen [thin90] devised a spring and hinge B-spline FEM model that was refineable on the fly. By controlling the refinement, "stiff" or "soft" surfaces could be created without fear of numerical instabilities, but this requires special handling of the external forces to determine which mass points an external force is applied to.

Recently, both Turner and Thalmann [turn93], and Lee, *et al*, [lee93] have proposed modeling only the skin of a character with a finite element model. This reduces the computation time, but requires that the interior of the character be composed of rigid objects. Since the finite element model remains springy, the simulation of the character's skin must still be carefully handled within an animation.

3.5 Isopotential surfaces: precise contact surfaces

Isosurface models have also been used to simulate deformable objects by using the collective isosurface of many points "attached" to an FEM [wyvi86].

Gascuel [gasc93] used the inside/outside functions used to define implicit solids for both collision detection and response. By modifying the field functions of the interpenetrating surfaces and using their relative stiffness, the precise contact surface was calculated. Finally, a non-linear elastic compression model was used to calculate the reaction forces applied to the rigid skeletons defining the isosurfaces.

4 Deformation for animators

Physically based models reduce the number of parameters required to specify surface behavior, making them practical for specifying secondary motions. However, the difficulty in determining the forces, constraints, and physical parameters to produce a particular motion sequence makes them less useful for controlling the principal components of a scene. This is further complicated because it is not obvious in any non-trivial system which parameters need to be modified to generate a desired effect.

This aspect gives rise to approaches that search the n-dimensional parameter space to determine the appropriate values.

Witkin and Kass's *Spacetime Constraints* [witk88], and Cohen's interactive *Spacetime Windows* [cohe92] are such approaches. However they have not yet been applied to the huge systems that describe surfaces, only to very small systems of rigid links connected by joints.

Some systems attempt to present the animator with a set of parameters that specify the physical model of the system (*e.g.*, the cloth model of [cari92]). But the difficulty lies in controlling the behavior of the model within an animation. One advantage of computer-based animation is the ability to reduce the number of parameters required to animate figures (see [brud89] for a model of human walking) and is what we attempt to do here for collision response.

5 Simulation vs. animation

Although these models and techniques are extremely powerful (and expensive), they are truly more appropriate to be used as *simulation* models rather than as *animation* models. The primary difference between simulation and animation is that between accuracy and meaning. Simulations should strive to be accurate, and higher meaning interpreted within a simulation is an illusion. Animations should also strive to be well designed and presented. Achieving this goal of *well presented meaning* often requires the use of "physically implausible" models, and *ad hoc* methods.

6 Algorithm and Implementation

The surface model used is a multi-resolution representation with $k + 1$ levels, with the coarsest level numbered 0, and the finest k, as illustrated in Figure 1. Fine resolutions correspond to

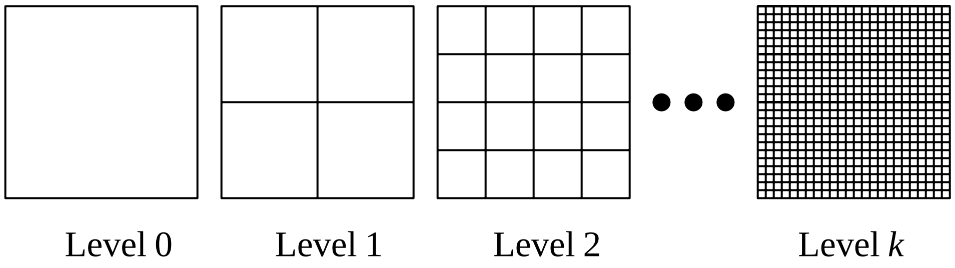

Figure 1: *The coarsest level of a multi-resolution model is level 0, and the finest is level k.*

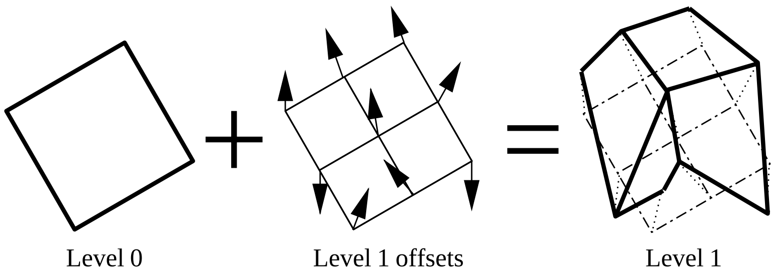

Figure 2: *A multi-resolution surface defines its finer levels as offsets from the coarser levels.*

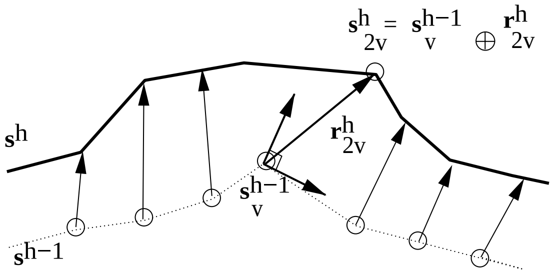

Figure 3: *The offset reference operator.*

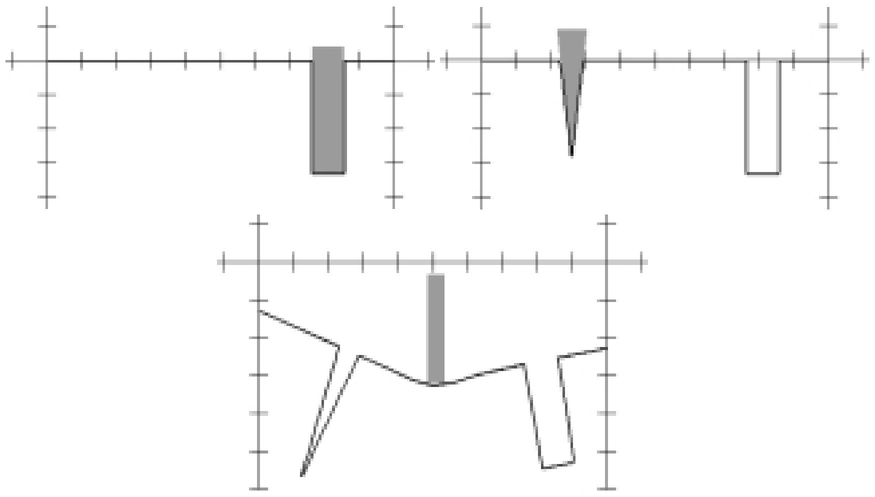

Figure 4: *Local vs. global deformations. The reference offset operator causes the initial deformations to follow the change in shape of the underlying surface.*

local features, and coarse resolutions correspond to global features. Similarly, deformations or displacements of the points of the fine resolutions result in local deformations and by combining the coarse and fine resolution deformations a wide variety of deformations can be generated.

At its finest representation, level k, the surface consists of an $N \times N$ patch of vertices. For every level j of resolution $(2n-1)\times(2n-1)$, its coarser resolution parent, level $j-1$ is of resolution $n \times n$. The coarsest representation, level 0, is a 2×2 patch of vertices.

The levels are inter-related by a *restriction operator* that coarsens the mesh, and a *prolongation operator* that refines it. These operators are essentially filters and are based upon those used in multigrid analysis [brig87], but those based upon spline basis functions or wavelets are also possible. Figure 2 shows how local details are defined as offsets from the coarser levels. To deform a surface while simulating the behaviors of inelasticity, elasticity, and viscosity, the offsets are changed over time as explained below.

To combine the prolongated basis of level 0 with the reference vectors of level 1 an offset referencing method is used in combination with the prolongation operator. This operator, $\oplus$, uses a local coordinate frame determined by the geometry of the prolongated coarse surface to calculate the world position of the fine level surface (Figure 3). Thus for each vertex, v, on level j, there is a displacement vector, $(\mathbf{d}_v^j)$, an offset vector $(\mathbf{o}_v^j)$, and the reference vector $(\mathbf{r}_v^j)$ mentioned previously. The offset vector records the displacement of a vertex from its rest position and is combined with the reference vector to calculate the final position of the vertex.

This offset reference method [fors88] allows finer-level details to follow the changes geometry of the coarser levels, as illustrated in Figure 4. This response is difficult to produce with other methods.

In addition to the offset information, each level also has associated with it a set of coefficients that are used determine how much of the total displacement of the surface will be absorbed by that level (α^j), how much to damp changes in the offset vectors $(\gamma^j$ and $\mu^j)$, and set the yield limit (τ^j), beyond which the elastic offsets become "permanent." The exact details of how these coefficients are used is described below.

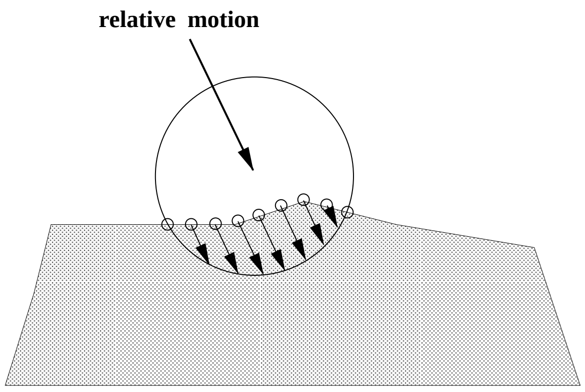

Figure 5: *A moving object interpenetrates a surface creating displacements vectors for points of the finest resolution of the surface.*

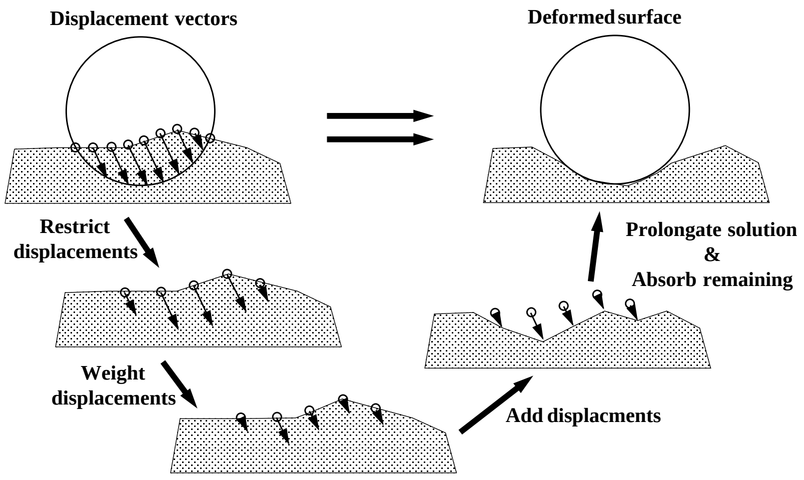

Figure 6: *Restriction of displacements followed by prolongation of results.*

7 Deformation as a Inelastic Process

When the surface is interpenetrated by an object, displacement vectors are created across the surface parallel to the vector of relative motion of the surface and object, as illustrated in Figure 5. These displacement vectors, if applied to the points of the finest level would resolve the existing interpenetration (to the resolution of the mesh), producing inelastic conformity of the surface to the interpenetrating object.

The geometric problem to solve is that when given a set of displacement vectors for the finest level, calculate a set of displacement vectors for all the levels that resolves the interpenetration.

The basic steps of our algorithm (illustrated in Figure 6) are as follows:

1. Using the finest level, k, sample the penetrating objects and calculate the displacement vectors, $\mathbf{d}^k$.
2. Normalize the per level absorptances, α^j, such that $\sum_{j=0}^{k} \alpha^j = 1$.
3. For $j = k - 1$ down to 0
   Restrict displacement vectors of level $j + 1$ to level j.
4. For each vertex v of level 0
   (a) Set new solution $\mathbf{n}_v^0 = \alpha^0 \mathbf{d}_v^0$.
   (b) Add old reference vectors $\mathbf{n}_v^0 = \mathbf{n}_v^0 \oplus \mathbf{r}_v^0$.
5. For $j = 1$ to k
   (a) Prolongate $\mathbf{s}^{j-1}$ to $\mathbf{n}^j$.
   (b) For each vertex v of level j
      i. $\mathbf{n}_v^j = \mathbf{n}_v^j \oplus \mathbf{r}_v^j$.
      ii. $\mathbf{change} = \mathbf{n}_v^j - \mathbf{s}_v^j$.
      iii. if $\mathbf{change} \cdot \widehat{\mathbf{d}_v^j} < \mathbf{d}_v^j$
         - then $\mathbf{remaining} = \left(||\alpha^j \mathbf{d}_v^j|| - \widehat{\mathbf{d}_v^j} \cdot \mathbf{change} \right) \widehat{\mathbf{d}_v^j}$.
         - if $||\mathbf{remaining}|| < ||\alpha^j \mathbf{d}_v^j||$
            – then $\delta = \mathbf{remaining}$.
            – else $\delta = \alpha^j \mathbf{d}_v^j$.
      iv. $\mathbf{r}_v^j = \mathbf{r}_v^j \oplus \delta$.
      v. Update old solution $\mathbf{s}_v^j = \mathbf{n}_v^j + \delta_v^j$

We present the details of the above process and extensions to it in the following sections.

7.1 Calculation of initial displacement

The displacement vectors of the finest level are calculated by casting rays along the vector of relative motion between the surface and object. This simple method requires a sampling resolution greater than that of the objects, and only handles convex objects. Concave objects can either be decomposed into sets of convex objects, or "ray-traced": starting at a surface point, step along the ray of relative motion until it exits the bounding box of the object and then determine the first intersection from this position back to the object.

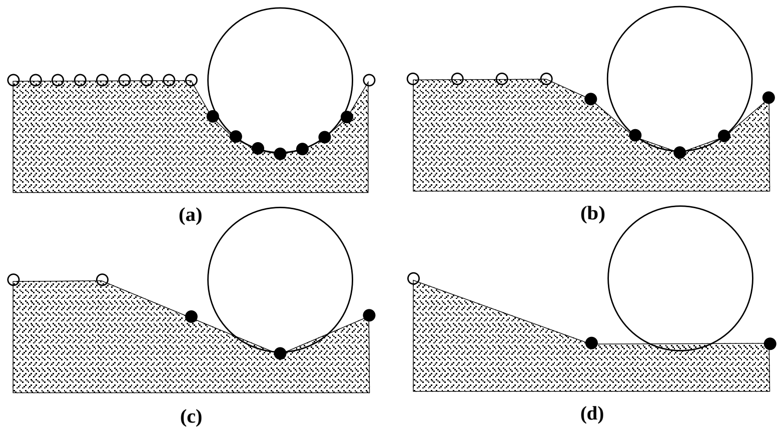

Figure 7: *Restriction of displacement vectors forms an approximation to the original object–surface interpenetration.*

7.2 Displacement Restriction

The displacement field at each level is restricted to form the displacement field of the next coarser level. As the resolution of the levels decreases, this propagates the deformation away from the region of contact and across the surface. The choice of restriction operator determines the behavior of the surface and it is important that it forms an accurate approximation. We have experimented with two simple restriction operators: maximum-displacement and average-displacement (Figure 7).

The maximum-displacement operator takes the longest vector from the children of a vertex and assigns this vector to the vertex:

$$\mathbf{d}_{i,j}^{h-1} = \mathrm{Max}_{\|\mathbf{d}\|} \begin{bmatrix} \mathbf{d}_{2i-1,2j-1}^{h} & \mathbf{d}_{2i-1,2j}^{h} & \mathbf{d}_{2i-1,2j+1}^{h} \\ \mathbf{d}_{2i,2j-1}^{h} & \mathbf{d}_{2i,2j}^{h} & \mathbf{d}_{2i,2j+1}^{h} \\ \mathbf{d}_{2i+1,2j-1}^{h} & \mathbf{d}_{2i+1,2j}^{h} & \mathbf{d}_{2i+1,2j+1}^{h} \end{bmatrix}$$

This operator produces fairly inaccurate approximations, but illustrates how the deformation propagates across the surface.

With the average-displacement operator, the parent vertices whose center child vertex has a non-zero length displacement vector, is assigned the maximum of this child vector and the average of the *non-zero vectors* among the other children:

$$\mathbf{d}_{i,j}^{h-1} = \mathrm{Max}_{\|\mathbf{d}\|} \left(\mathbf{d}_{2i,2j}^{h}, \mathrm{Ave}(d^h, 2i, 2j) \right)$$

This tends to smooth out variations in the displacement vector field. For parent vertices whose center child vertex is zero-length, it behaves like the maximum-displacement vector, propagating the edge of contact outwards.

7.3 Absorption coefficients

The restricted displacements are approximations to the original displacements calculated for level k, and reflect the propagation of the deformation across the surface as the resolution decreases.

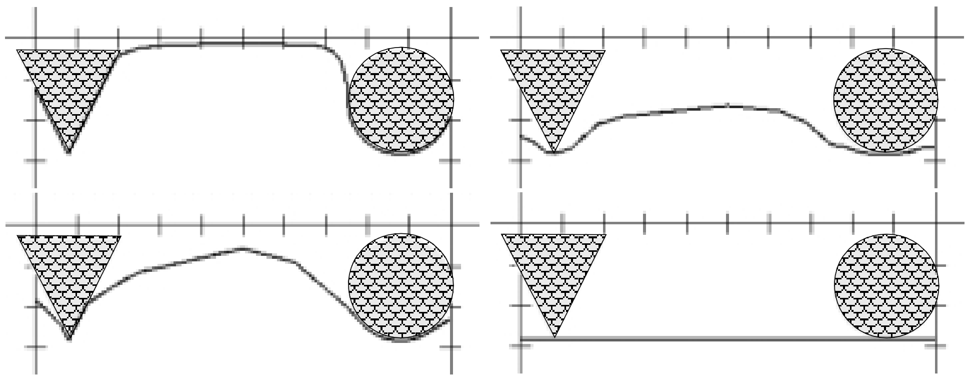

Figure 8: *Cross-sections of a piecewise-linear surface in response to displacement by two objects (finest level is 513 points across).*

By combining the different solutions, it is possible to generate a deformation response that varies, under the direct control of the animator, from completely global (stiff) to local (compliant). The absorption coefficients for each level, determine the fraction of the displacement field of each level that is contributed toward the final solution.

The normalization of the absorption coefficients such that $\sum_{j=0}^{k} \alpha^j = 1$ is not necessary. Instead, if $\sum_{j=0}^{k} \alpha^j < 1$, then the remaining deformation can be absorbed by the finest level, or left for the next iteration, depending upon the choice of the animator.

Figure 8 demonstrates how different absorption coefficients alter the response of the surface.

7.3.1 Combinating Levels

Starting with the weighted coarsest-level response, the solutions are collected upwards toward the finest level. For each vertex in a level, the displacement caused by its parent levels from the last solution (the previous state of the surface) is compared to the restricted displacement. The contribution of a vertex, a change in its reference vector, is the smaller of either displacement remaining to be absorbed or the weighted displacement.

Once this process is completed for a level, this intermediate solution is prolongated using the linear interpolation prolongation operator to the next level:

$$\begin{aligned}
\mathbf{n}^h_{2i,2j} &= \mathbf{s}^{h-1}_{i,j} \\
\mathbf{n}^h_{2i,2j+1} &= \tfrac{1}{2}(\mathbf{s}^{h-1}_{i,j} + \mathbf{s}^{h-1}_{i,j+1}) \\
\mathbf{n}^h_{2i+1,2j} &= \tfrac{1}{2}(\mathbf{s}^{h-1}_{i,j} + \mathbf{s}^{h-1}_{i+1,j}) \\
\mathbf{n}^h_{2i+1,2j+1} &= \tfrac{1}{4}(\mathbf{s}^{h-1}_{i,j} + \mathbf{s}^{h-1}_{i+1,j} + \mathbf{s}^{h-1}_{i,j+1} + \mathbf{s}^{h-1}_{i+1,j+1})
\end{aligned}$$

8 Deformation as an Elastic Process

This algorithm is easily extended to model to elastic time-dependent behaviors. Instead of modifying the reference positions of the vertices, offsets ($\mathbf{o}^h$) are introduced that are used to hold the displacement of the surface. As these offsets grow and shorten, the surface deforms elastically over time.

This changes step 5(b)i into

$$\mathbf{n}_v^j = \mathbf{n}_v^j \oplus \left(\mathbf{r}_v^j + \mathbf{o}_v^j\right),$$

and step 5(b)iv to

$$\mathbf{o}_v^j = \mathbf{o}_v^j \oplus \delta_v^j.$$

A perfectly springy and non-viscous surface will have offsets that return to zero-length upon the removal of the intruding objects. As the objects move across the surface it will "pop" suddenly into position, a behavior inappropriate for animation.

To remove these pops, viscosity is specified via a per-level relaxation coefficient (μ^h). Before the displacement field is calculated a copy of the offsets, $\mathbf{p}^k$, is made such that $\mathbf{p}_v^k = \mu^k \mathbf{o}_v^k$. The reference vectors and these shortened offsets are combined to form a relaxed surface state:

```
1. Set relaxed solution $\mathbf{S}_v^0 = \mathbf{r}_v^0 + \mathbf{p}_v^0$.
2. For $j = 1$ to $k$
   (a) Prolongate $\mathbf{S}^{j-1}$'s to $\mathbf{S}^j$.
   (b) for each vertex $v$ of level $j$
       i. $\mathbf{S}_v^j = \mathbf{S}_v^j \oplus \left(\mathbf{r}_v^j + \mathbf{p}_v^j\right)$.
```

This relaxed surface, $\mathbf{S}^k$, is used to calculate the displacement vectors. For any vertex, if $\mathbf{d}_v^k$ is non-zero then the original offset is used, otherwise $\mathbf{o}_v^k$ is set to the relaxed length $\mathbf{p}_v^k$.

In addition, the growth of the offsets is damped by the use of a per-level damping coefficient γ, replacing step 5(b)iv with

$$\mathbf{o}_v^j = \mathbf{o}_v^j \oplus \gamma^j \delta.$$

Figure 9 and Figure 10 illustrate the behavior of a viscous surface, where the coarse levels are lightly damped, and the finer levels are heavily damped.

To an animator, the ability to control exactly when an event occurs is extremely important. For objects that have ceased their relative motion, it is possible to bring a system of interpenetrating objects to a rest state before a specific time by calculating the appropriate damping values γ^h and μ^h.

Finally, if the length of the offset vector grows beyond a yield length (τ^h) the reference vector is modified to reflect this permanent deformation, necessitating the addition of a final step to the algorithm,

```
• if $||\mathbf{o}_v^k|| > \tau^k$ then
    – $\mathbf{t} = \mathbf{o}_v^k - \tau^k \widehat{\mathbf{o}_v^k}$.
    – $\mathbf{r}_v^k += \mathbf{t}$.
    – $\mathbf{o}_v^k = \tau^k \widehat{\mathbf{o}_v^k}$.
```

Bulging around the region of interpenetration simulated by setting some of the absorption coefficients to negative values. The coarser the level is, the more global the bulging will be, and the finer the more local as illustrated in Figure 11. If α^j is negative, then the restricted displacement vectors are weighted by α^j and then added to the current solution, replacing steps 3(b)ii through 3(b)iv with the single assignment:

$$\mathbf{o}_v^j \leftarrow \mathbf{o}_v^j \oplus \gamma^j \alpha^j \mathbf{d}_v^j.$$

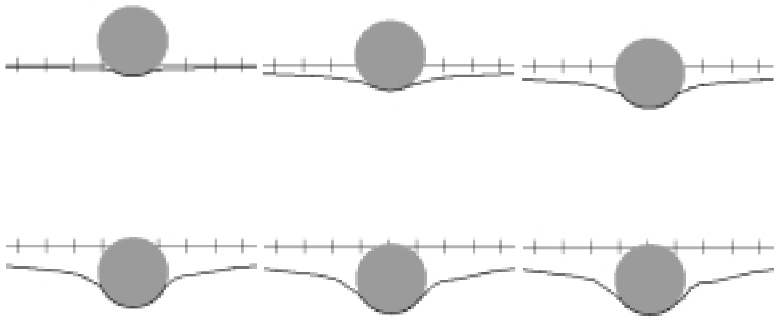

Figure 9: *The deformation of a viscously damped surface.*

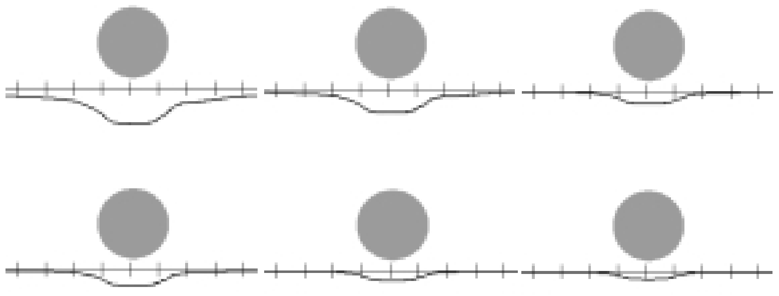

Figure 10: *The relaxation of a previously deformed viscous surface.*

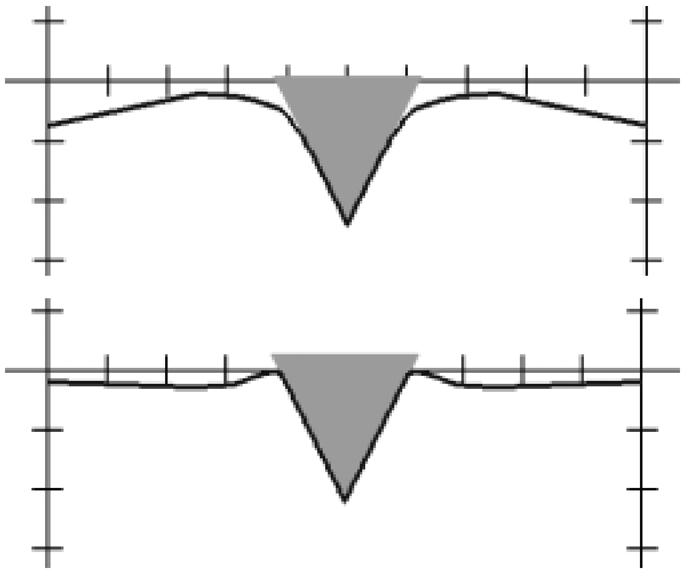

Figure 11: Global and local bulging of the surface by negating absorption coefficients.

9 Conclusions and Future Work

We have presented a simple, fast, geometric approach for controlling the nature, extent, and timing of the surface deformations arising from the interpenetration of kinematically controlled animated objects. This approach takes advantage of a multi-resolution representation to provide the control and speed necessary for real-time interaction.

Future extensions to this work include calculating the response of two deformable surfaces interpenetrating each other, and resolving self-surface interpenetrations so that these surfaces can be used as the skin for articulated figures.

The reference offset method used to represent inelastic deformations is a powerful primitive that will be further explored as a modeling tool in its own right. Currently, the displacement vector calculation ignores surface details such as protrusions, that could be specified as stiffer or more elastic than the surrounding surface.

When two deformable surface interpenetrate the resulting deformations should take into account the relative stiffness of the surfaces. The approach used by [gasc93] is to weight the displacement vectors for both surfaces by the relative stiffness of the surfaces.

If the surfaces are both homogeneous and isotropic then this approach will work. Otherwise, relying upon the finest level to absorb the remaining displacements will produce non-intuitive results if a stiff portion of the surface suddenly deforms compliantly. In these situations the step size would be shortened. The displacement vectors are weighted by an additional "step size" coefficient and the deformation algorithm is iteratively applied until the deformation absorbed by the finest level has been minimized. These methods would also work for self-surface collisions.

To create heterogeneous and anisotropic surfaces the per-level controls: α^j, μ^j, γ^j, and τ^j, must vary across a level. A direct method such as that presented in [hanr90] could be used to "paint" the appropriate values onto the shaped surface.

Reaction forces from a deformation can be generated by using a spring model for the elastic offsets. An ideal Hookean spring, $\mathbf{F}(\mathbf{x}) = -k\mathbf{x}$, is okay. But an approximation of a biphasic model for facial tissue as presented in [lee93]:

$$k = \begin{cases} k_\alpha & \text{when } x \leq x_l, \\ k_\beta & \text{when } x > x_l, \end{cases}$$

where the small-deformation stiffness k_α is smaller than the large-deformation stiffness k_β is better when we're modeling surfaces surrounding volumes that are only moderately compressible.

The reaction forces can either be prolongated from the coarser levels upwards to calculate the per-surface area reaction force (pressure). Or integrated to calculate the torsional and translational forces upon the body the surface is attached to.

Many things we would like to model with a deformable surface have a complicated internal structure. Two ways of replicating these structures are to layer internal surfaces upon one another, and to create internal features. Layering allows each layer to have its own properties and behaviors, like stiffness, compressibility, and actuation (muscles). Figure 12 shows how the variation in a compliant layer's thickness varies the response of the surface.

The simplest way to layer surfaces is to replicate a resolution of a level. This copy can then be procedurally or kinematically animated to change the shape of the surface. This is equivalent to giving the animator control over the offsets, or muscles, of the layer. Another method is to make the coarsest level correspond to the coarse "natural" resolution of the object the surface defines. Deforming this level deforms the surface.

Internal features can be created by using the reference offsets. This creates surface features that should respond to penetrations in novel ways. Figure 13 demonstrates how a bump in the surface could react either elastically or stiffly. Creating these responses requires new restriction operators that recognize such features and calculate the necessary displacement vectors.

The relaxation step is a good place to insert volume and surface area conservation functionals. While relaxing the surface, volume conservation can be simulated by shorting the elastic offsets to recover the volume lost by the deformation. This is similar to what is done in [lee93]. Available functionals can be taken from [plat88]. Surface area can be conserved by applying a separate multigrid technique such as [pala93].

We have also experimented with the direction the displacement vectors are cast. The two basic directions are into the surface — parallel to the surface normals, — and along the vector of relative motion of the object and the surface.

The direction is determined by the amount of friction between the surface and the object to be simulated. If the contact is perfectly frictionless, then there are no "forces" deforming the surface perpendicular to its normal: the surface slides around the object. If the contact represents perfect friction, the surface will be displaced along the vector of relative motion: the surface is pushed by the object. An adhesive contact will cause the contact points of the surface to remain in contact. Figure 14 shows the result of applying displacements in different directions.

Between the perfect friction and frictionless conditions a "coefficient of friction" for the surface and the object is used to weight the surface normals and the relative motion vector when calculating the direction to cast the displacement vectors along.

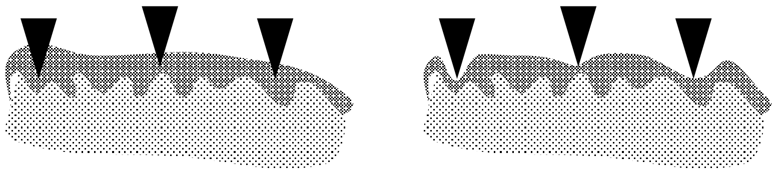

Figure 12: The variation in a compliant layer's thickness results in a heterogeneous surface.

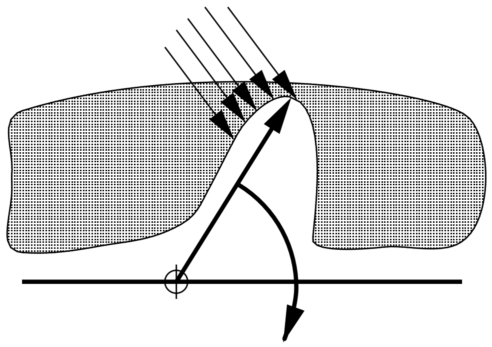

Figure 13: The underlying features of a surface create different coarse scale responses.

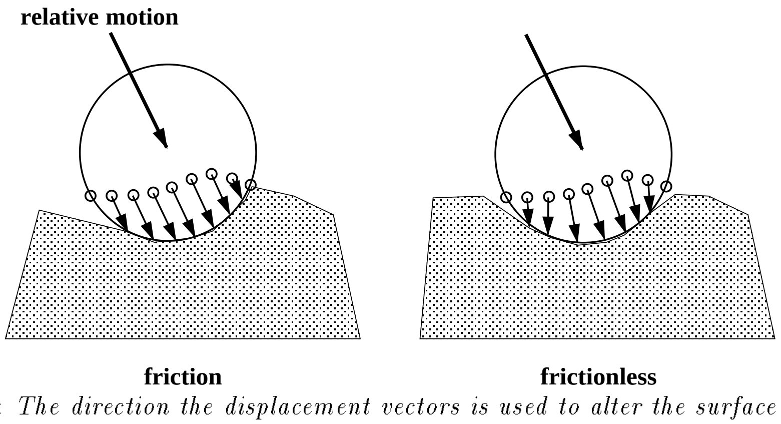

Figure 14: *The direction the displacement vectors is used to alter the surface response.*

References

[brig87] William L. Briggs. *A multigrid tutorial.* Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1987.

[brud89] Armin Bruderlin and Thomas W. Calvert. “Goal-directed, dynamic animation of human walking”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, No. 3, pp. 233–242, July 1989.

[cari92] Michel Carignan, Ying Yang, Nadia Magnenat-Thalmann, and Daniel Thalmann. “Dressing animated synthetic actors with complex deformable clothes”. *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, No. 2, pp. 99–104, July 1992.

[celn91] George Celniker and Dave Gossard. “Deformable curve and surface finite-elements for free-form shape design”. *Computer Graphics (SIGGRAPH '91 Proceedings)*, Vol. 25, No. 4, pp. 257–266, July 1991.

[celn92] George Celniker and Will Welch. “Linear constraints for deformable B-spline surfaces”. *Computer Graphics Special Issue (1992 Symposium on Interactive 3D Graphics)*, Vol. 26, pp. 165–170, March 1992.

[chad89] John E. Chadwick, David R. Haumann, and Richard E. Parent. “Layered construction for deformable animated characters”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, No. 3, pp. 243–252, July 1989.

[cohe92] Michael F. Cohen. “Interactive spacetime control for animation”. *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, No. 2, pp. 293–302, July 1992.

[fari88] Gerald E. Farin. *Curves and surfaces for computer aided geometric design: a pritical guide.* Academic Press, Inc., San Diego, 1988.

[fink94] Adam Finkelstein and David H. Salesin. “Multiresolution curves”. Technical Report 94-01-06, Department of Computer Science and Engineering, University of Washington, Seattle, Washington, 1994.

[fors88] David R. Forsey and Richard H. Bartels. “Hierarchical B-spline refinement”. *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 205–212, August 1988.

[fowl92] Barry Fowler. “Geometric manipulation of tensor product surfaces”. *Computer Graphics Special Issue (1992 Symposium on Interactive 3D Graphics)*, Vol. 26, pp. 101–108, March 1992.

[gasc93] Marie-Paule Gascuel. “An implicit formulation for precise contact modeling between flexible solids”. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pp. 313–320, August 1993.

[gour89] Jean-Paul Gourret, Nadia Magnenat Thalmann, and Daniel Thalmann. “Simulation of object and human skin deformations in a grasping task”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, No. 3, pp. 21–30, July 1989.

[hanr90] Pat Hanrahan and Paul Haeberli. “Direct WYSIWYG Painting and Texturing on 3D Shapes”. *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, No. 4, pp. 215–223, August 1990.

[lee93] Yuencheng Lee, Demetri Terzopoulos, and Keith Waters. "Constructing physics-based facial models of individuals". *Proceedings of Graphics Interface '93*, pp. 1–8, May 1993.

[mill88] Gavin S. P. Miller. "The motion dynamics of snakes and worms". *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 169–178, August 1988.

[more92] Henry P. Moreton and Carlo H. Séquin. "Functional optimization for fair surface design". *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, No. 2, pp. 167–176, July 1992.

[pala93] Larry Palazzi. "Deformable models using displacement constraints". M.Sc. Thesis, Department of Computer Science, University of British Columbia, October 1993.

[plat88] John C. Platt and Alan H. Barr. "Constraint methods for flexible models". *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 279–288, August 1988.

[schw66] D. G. Schweikert. "An interpolation curve using a spline in tension". *Journal of Math and Physics*, Vol. 45, pp. 312–317, 1966.

[sede86] T.W. Sederberg and S.R. Parry. "Free-form deformation of solid geometric models". *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, No. 4, pp. 151–160, August 1986.

[terz87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. "Elastically deformable models". *Computer Graphics (SIGGRAPH '87 Proceedings)*, Vol. 21, No. 4, pp. 205–214, July 1987.

[terz88a] Demetri Terzopoulos. "The computation of visible-surface representations". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, pp. 1–18, July 1988.

[terz88b] Demetri Terzopoulos and Kurt Fleischer. "Modeling inelastic deformation: Viscoelasticity, plasticity, fracture". *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 269–278, August 1988.

[terz88c] Demetri Terzopoulos and Andrew Witkin. "Physically-based models with rigid and deformable components". *Proceedings of Graphics Interface '88*, pp. 146–154, June 1988.

[thin90] Jeffrey A. Thingvold and Elaine Cohen. "Physical modeling with B-spline surfaces for interactive design and animation". *Computer Graphics (1990 Symposium on Interactive 3D Graphics)*, Vol. 24, No. 2, pp. 129–137, March 1990.

[turn93] Russell Turner and Daniel Thalmann. "The elastic surface layer model for animated character construction". *Proceedings of Computer Graphics International '93*. Springer-Verlag, 1993.

[weil86] J. Weil. "The synthesis of cloth objects". *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, No. 4, pp. 49–54, August 1986.

[welc92] William Welch and Andrew Witkin. "Variational surface modeling". *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, No. 2, pp. 157–166, July 1992.

[witk88] Andrew Witkin and Michael Kass. "Spacetime Constraints". *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 159–168, August 1988.

[witk90] Andrew Witkin, Michael Gleicher, and William Welch. "Interactive dynamics". *Computer Graphics (1990 Symposium on Interactive 3D Graphics)*, Vol. 24, No. 2, pp. 11–21, March 1990.

[wyvi86] Geoff Wyvill, Craig McPheeters, and Brian Wyvill. "Data structure for soft objects". *The Visual Computer*, Vol. 2, No. 4, pp. 227–234, August 1986.