A Kinematic Model for Collision Response

by

Jason Harrison

B.E., The State University of New York at Stony Brook, 1992

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of

Master of Science

in the Faculty of Graduate Studies
Department of Computer Science

We accept this thesis as conforming to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
April, 1994

Computer Science
The University of British Columbia
Department of Computer Science
2075 Westbrook Mall Vancouver, BC, Canada
V6T 1Z1

Date:

Abstract

Over the course of several years, an informal poll by Dave Forsey of commercial animators revealed that the single most useful tool to add to a computer animation system would be a way to control the simple act of pressing one object into another. This ability would drastically increase the range and style of animations that could be created within a production environment.

This thesis 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 is based upon the idea that collision response is a matter of moving specific points on the surface towards specific goal points at a certain rate. This is an operation similar to manipulating chains of rigid bodies using inverse kinematics. This formulation is easily extendable to free-form surface modeling where generally only one point, or possibly more, is manipulated interactively at a time.

We use a multi-resolution approach to deformations to provide control over the response of the surface using a small number of parameters. Each of the parameters determine how a level in the multi-resolution representation of the surface reacts to the interpenetration. The deformations are calculated in time and space proportional to the number of points used to define the surface.

Table of Contents

List of Figures

Acknowledgments

I'd like to thank everyone who helped me with this research. My thesis advisor Dave Forsey started me off on this line of research, and then made sure I didn't get lost for more than a month at a time. Alain Fournier was extremely helpful as my second reader, taking the time out of his busy schedule to dissect my thesis.

To everybody who thought my demos were "Cool!", thanks. The results of this thesis got to be a little stale after a while due to my familiarity with them. The dropped jaws helped a lot to keep me going.

Markus Tessman and Goesta Struve-Dencher provided their experience as professional animators battling with commercial and research software animation tools on a daily basis. I hope this hasn't strayed from what you told me you really wanted.

Uri Asher, Paul Lalonde, Bob Lewis, and Jack Snoeyink helped me understand more of the theory as I tried to explain exactly what I was trying to do. Paul was especially helpful with my first conference paper submission when he really wanted to be somewhere else. And Bob kindly edited an incomplete draft.

Finally, personal thanks to all my friends who kept me away from the lab when I really needed a break, especially on Tuesday nights. Jim and Gayla got me out into the sun on a regular basis. Robyn and Wade put 300 km onto our bikes one weekend. Patrick, Markus, and Dotty managed to be more stressed out than I was and still have fun. And Lauri lead me past hundreds of runners to get us to the finish line in under an hour.

Dedication

I dedicate this thesis to the animators:
Creating the illusion of life, against the odds.

Chapter One

Introduction

I brought my BOWLING BALL — and some DRUGS!!
— Zippy the Pinhead

When confronted with a situation where two or more surfaces interact an animator must usually resort to painstaking manipulation of individual degrees of freedom in the models. 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.

This thesis presents a simple, fast, geometric approach to controlling the nature, extent and timing of the surface deformations of objects arising from the interpenetration of kinematically controlled animated objects. This approach allows an animator to control the result of the simple act of pressing one object into another.

1.1 Kinematic animation

Most animations are specified kinematically either by interpolated key frames or by parametric motion paths. The approach presented here is kinematic in nature and thus fits naturally into this framework of kinematically specified animations. It allows collision response to be used as *secondary motion*: motion that is not the main focus of the animation, but adds character to the primary motion. Examples of secondary motion include jiggles and wiggles that soften the emergent "physical" characteristics of a character.

We would like to avoid the use of dynamics. While their use is increasing they are unsuitable for the largely kinematically specified commercial environment. Also, it is often difficult to determine how to set the parameters of a dynamic simulation to mimic a particular behavior — an emergent property of a system's physical parameters.

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

As a kinematic solution, the method presented is numerically stable, and temporally and spatially compact in its use of resources.

Inverse kinematics is employed to specify the position of the points on the surface not directly in contact with the penetrating object. Their motion is formulated as the result of a hierarchy of vertex and reference offset methods [fors88]. Thus, the response of the surface is controlled by a number of parameters directly proportional to the number of levels in the hierarchy.

1.2 Animation problems addressed by this thesis

This thesis presents a method for controlling the geometric changes in a surface regardless of the origins of the changes. This includes free-form surface modeling — where a sculptor manipulates the surface shape by "pulling" on the surface — and the deformations arising from surface–surface collisions.

For the scope of this thesis, a collision occurs when two (or more) surfaces mutually penetrate. Thus a collision is the result of an interpenetration, rather than the classical definition of an impact between two bodies. If one of the interpenetrating surfaces is a deformable surface, then the methods proposed in this thesis are applicable. The issues surrounding rigid-body motion are not addressed by this thesis. This thesis also is not concerned with the contact of surfaces, only with the result of their interpenetration.

1.3 Thesis roadmap

Previous research on deformable surfaces for modeling and animation will be presented in Chapter 2. The terminology of deformations as they refer to surfaces will be delineated in Chapter 3. The multi-resolution algorithm for deformable surfaces will be presented initially in Chapter 4 and the actual implementation in Chapter 5 with the future extensions. The conclusions follow in Chapter 6. The mathematical notations used and a glossary are at the end of this thesis.

Chapter Two

Previous work

Mmmmmm-MMMMMM!! A plate of STEAMING PIECES of a PIG mixed with the shreds of SEVERAL CHICKENS!! Oh BOY!! I'm about to swallow a TORN-OFF section of a COW'S LEFT LEG soaked in COTTONSEED OIL and SUGAR!! Let's see... Next, I'll have the GROUND-UP flesh of CUTE, BABY LAMBS fried in the MELTED, FATTY TISSUES from a warm-blooded animal someone once PETTED!! YUM!! That was GOOD!! For DESSERT, I'll have a TOFU BURGER with BEAN SPROUTS on a stone-ground, WHOLE WHEAT BUN!!
— Zippy the Pinhead

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.

2.1 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.

2.1.1 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.

2.1.2 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:

$$\iint_{u,v} \left(\|\mathrm{G}\|_\alpha^2 + \|\mathrm{B}\|_\beta^2 \right) \, \mathrm{d}u \, \mathrm{d}v \tag{2.1}$$

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

$$\mathrm{G}(u,v) = \mathbf{x}_u{}^2 + 2\mathbf{x}_u\mathbf{x}_v + \mathbf{x}_v{}^2 \tag{2.2}$$

$$\mathrm{B}(u,v) = \mathbf{x}_{uu} + 2\mathbf{x}_{uv} + \mathbf{x}_{vv}, \tag{2.3}$$

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. The energy matrix norms, $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$, weight the stretching and bending terms, they are introduced by rewriting Equations 2.2 and 2.3 in normal matrix form:

$$\|\mathrm{G}\|_\alpha = [\mathbf{x}_u \ \mathbf{x}_v] \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} [\mathbf{x}_u \ \mathbf{x}_v]^{\mathrm{T}} \tag{2.4}$$

$$\|\mathrm{B}\|_\beta = [\tfrac{\partial}{\partial u} \ \tfrac{\partial}{\partial v}] \begin{bmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{bmatrix} [\tfrac{\partial}{\partial u} \ \tfrac{\partial}{\partial v}]^{\mathrm{T}} \ \mathbf{x}. \tag{2.5}$$

Multiplying out Equations 2.4 and 2.5 and collecting the sums $\alpha_{12} + \alpha_{21}$ and $\beta_{12} + \beta_{21}$, results in the weighted fundamental forms:

$$\alpha_{11}\mathbf{x}_u{}^2 + 2\alpha_{12}\mathbf{x}_u\mathbf{x}_v + \alpha_{22}\mathbf{x}_v{}^2 \tag{2.6}$$

$$\beta_{11}\mathbf{x}_{uu} + 2\beta_{12}\mathbf{x}_{uv} + \beta_{22}\mathbf{x}_{vv}. \tag{2.7}$$

If the matrices α and β are matrices of all 1's, Equations 2.6 and 2.7 result in the the original fundamental forms (Equations 2.2 and 2.3). Squaring the weighted fundamental forms, and substituting back into the original functional (Equation 2.1) yields

$$\iint_{u,v} \left(\begin{array}{c} \alpha_{11}^2 {\mathbf{x}_u}^4 + 4\alpha_{11}\alpha_{12}{\mathbf{x}_u}^3\mathbf{x}_v + 2\alpha_{11}\alpha_{12}{\mathbf{x}_u}^2{\mathbf{x}_v}^2 + 4\alpha_{12}^2{\mathbf{x}_u}^2{\mathbf{x}_v}^2 \\ +4\alpha_{12}\alpha_{22}\mathbf{x}_u{\mathbf{x}_v}^3 + \alpha_{22}^2{\mathbf{x}_v}^4 + \beta_{11}^2{\mathbf{x}_{uu}}^2 + 4\beta_{12}\beta_{12}\mathbf{x}_{uu}\mathbf{x}_{uv} \\ +2\beta_{11}\beta_{22}\mathbf{x}_{uu}\mathbf{x}_{vv} + 4\beta_{12}^2{\mathbf{x}_{uv}}^2 + 4\beta_{12}\beta_{22}\mathbf{x}_{uv}\mathbf{x}_{vv} + \beta_{22}^2{\mathbf{x}_{vv}}^2 \end{array} \right) \mathrm{d}u\,\mathrm{d}v. \tag{2.8}$$

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]:

$$\iint_{u,v} \left(\|\mathrm{G}\|_\alpha + \beta_{11}{\mathbf{x}_{uu}}^2 + 2\beta_{12}{\mathbf{x}_{uv}}^2 + \beta_{22}{\mathbf{x}_{vv}}^2 \right) \mathrm{d}u\,\mathrm{d}v. \tag{2.9}$$

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 it is used in animated sequences to control deformations orders of magnitude greater than the underlying theory allows, the use of this simplification reduces the cost of minimizing the objective functional. For a linear surface representation, such as a tensor product B-spline, Equation 2.9 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 alii*, [terz87] used the thin-plate functional to measure the deformation energy of a surface from its natural shape:

$$\mathcal{E}(\mathbf{r}) = \iint_{u,v} \left(\|\mathrm{G} - \mathrm{G}^0\|_\alpha^2 + \|\mathrm{B} - \mathrm{B}^0\|_\beta^2 \right) \mathrm{d}u\,\mathrm{d}v, \tag{2.10}$$

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

$$\mathcal{E}(\mathbf{r}) = \iint_{u,v} \sum_{i,j} \left(\eta_{ij}(\mathrm{G}_{ij} - \mathrm{G}^0_{ij})^2 + \xi_{ij}(\mathrm{B}_{ij} - \mathrm{B}^0_{ij})^2 \right) \mathrm{d}u \, \mathrm{d}v, \tag{2.11}$$

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 2.11 by keeping only terms of the first order, giving the vector expression :

$$\mathrm{e}(\mathbf{r}) = \sum_{i,j} \frac{\partial}{\partial i} \left(\alpha_{ij} \frac{\partial \mathbf{r}}{\partial i} \right) + \frac{\partial^2}{\partial i \partial j} \left(\beta_{ij} \frac{\partial^2 \mathbf{r}}{\partial i \partial j} \right), \tag{2.12}$$

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

$$\alpha_{ij}(\mathbf{x}, \mathbf{r}) = \eta_{ij}(\mathbf{x})(\mathrm{G}_{ij} - \mathrm{G}^0_{ij}), \tag{2.13}$$

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 2.12 yields unwieldy expressions when the calculus of variations is applied to it, their alternative is to use by analogy to Equation 2.13:

$$\beta_{ij}(\mathbf{x}, \mathbf{r}) = \xi_{ij}(\mathbf{x})(\mathrm{B}_{ij} - \mathrm{B}^0_{ij}), \tag{2.14}$$

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 2.12 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 2.9). 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 alii*, [cari92] used the thin-plate model of [terz87] (Equation 2.11) 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:

$$\iint_{u,v} \left(\left(\frac{\mathrm{d}\kappa_0}{\mathrm{d}u} \right)^2 + \left(\frac{\mathrm{d}\kappa_0}{\mathrm{d}v} \right)^2 \right) \mathrm{d}u\,\mathrm{d}v, \tag{2.15}$$

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:

$$\kappa_0 = \kappa_0(\mathbf{x}, \mathbf{s}) = \frac{\mathrm{B}(u,v)}{\mathrm{G}(u,v)} = \frac{\mathbf{x}_{uu} + 2\mathbf{x}_{uv} + \mathbf{x}_{vv}}{{\mathbf{x}_u}^2 + 2\mathbf{x}_u\mathbf{x}_v + {\mathbf{x}_v}^2}. \tag{2.16}$$

Thus Equation 2.15 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.

2.1.3 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].

2.1.4 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 alii*, [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 alii*, [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 alii*, [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.

2.1.5 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.

2.2 Deformation for animators

The physics and energy based models, while appropriate for modeling deformable objects, are inappropriate in many ways for the purpose of 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). 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.

2.3 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.

Chapter Three

A few fundamentals of surface deformation

I'm in direct contact with many advanced fun CONCEPTS.
— Zippy the Pinhead

The theory of elasticity and the study of elastic materials has a long history in physical modeling [fung81]. Models for the physical characteristics of a material, its elasticity, viscoelasticity, plasticity, and compressibility have been formulated from principles of physics and variational calculus.

We are interested in a kinematic model of deformable surfaces that enclose volumes and objects. The dynamic models provide insight into the design of a kinematic model of deformable materials, but are not the basis of the design presented here.

3.1 What is a kinematic deformation?

We characterize the deformation behavior of a surface without using the mass distributions, volume conservation, or internal forces inherent in the dynamic models of elasticity. The definition of a deformation in our framework is

the change of shape of a surface caused by the
displacement of surface points relative to each other over time.

This definition is compatible with the standard definition of deformation, *the alteration of form or shape* [merr76] [koen90], and will be used to define the properties of a deformable surface.

3.2 What is collision response?

We have previously defined a collision as the mutual penetration of two or more surfaces. We define a collision response, the *effect* of a collision, and for a deformable surface as the *time dependent deformation of the surface so as to resolve the penetration.* This thesis is primarily concerned with the collision response of a surface, but other surface deformation applications will be mentioned. For example, free-form surface editing deforms a surface without regard to time.

3.3 Properties of a deformable surface

We will categorize the properties of elasticity and plasticity of a deformable surface as its *nature*. The compliance or stiffness of a surface combined with its ability to be stretched determines the extent of the surface involved in a deformation. Time dependent properties of a surface material include viscosity.

Animators require control over the objects involved in their animations. This thesis categorizes the controls for a deformable surface as those that specify the *nature*, *extent*, and *timing* of the deformations arising from the interpenetrations of kinematically animated objects.

3.3.1 Nature

The *nature* of a deformation is the combination of its reversibility, and uniformity. A perfectly elastic surface will recover its shape. A surface with a plastic nature will not. An isotropic and homogeneous nature will produce a surface that is uniform and unoriented. An anisotropic surface will be oriented, and a heterogeneous surface will be non-uniform.

Elasticity

An elastic material recovers it size and shape after the effect causing the deformation is removed. A perfectly elastic surface deforms as necessary to resolve the interpenetration, and recovers when possible. This requires that the deformation of the surface from the original shape be recorded for later recovery as an *elastic offset.*

Plasticity

Most solid materials exhibit some ability to be permanently molded. The force required to plastically deform a material is known as its *yield limit.* For a kinematic model, we use a displacement limit — instead of a force limit — to define the yield limit of a deformable surface. Beyond this yield limit, we either evolve the rest shape of the surface as in [terz88b], or record a portion of the elastic displacement as permanent, a *plastic offset.*

Anisotropy and heterogeneity

The alignment of the grain of a material determines whether it will exhibit isotropic or anisotropic behaviors. Anisotropic surfaces deform differently based upon the orientation of the excitation, isotropic surfaces do not. This is different from heterogeneity or homogeneity, where the material properties are either non-uniformly or uniformly varied across the surface. Figure 3.1 illustrates an anisotropic response generated by varying the properties across the surface in strips.

3.3.2 Extent

The extent of a deformation defines how much of the surface is deformed by an interpenetration. Compliant surfaces respond only locally: the area immediately surrounding the penetration.

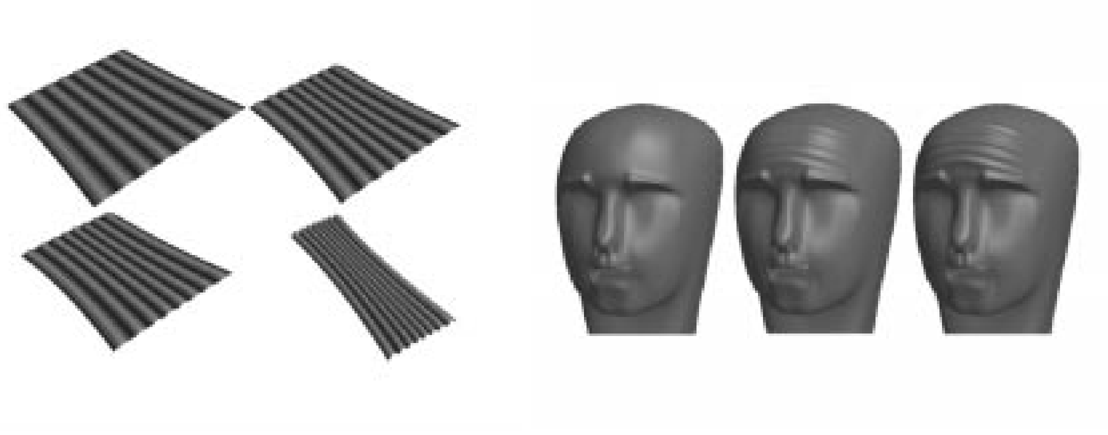


Figure 3.1: *Wrinkling of a skin model is created by using a procedural offset method coupled with variations in the surface properties. Images by Dave Forsey.*

Stiff surfaces respond on a more global extent to a penetration. Underlying of the surface affect the extent of a deformation in various ways. Similarly, the stretchability and compressibility of a surface determines how much of the surface will be affected.

Local vs. global compliance

A compliant surface, one that is *soft*, deforms locally. As the stiffness of the surface is increased, the deformations becomes more global, until the whole surface is deformed by an interpenetration. The simulation of ideal rigidity, the limit of stiffness, is not the concern of this thesis (the interested reader is referred to the excellent papers on simulating rigid body motion: [moor88], [bara90], [herz90], [snyd93]). Figure 3.2 shows four different surface responses.

Underlying features of the surface

Surfaces are often overlaid over or sandwiched between other objects. For example, a common model of the human face is an epidermal layer over a dermal layer attached to the muscle layer that is attached to the skull and jaw. Each of the layers has its own variable thickness and properties. As the muscle layer changes shape, it affects the dermal and epidermal layers.

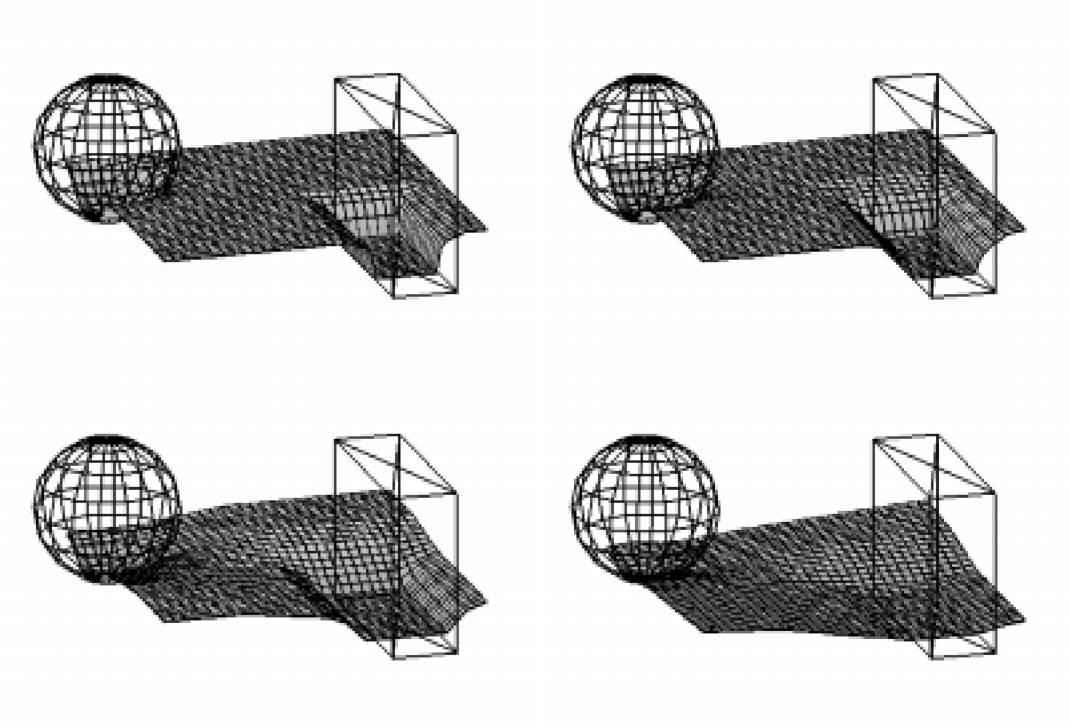


Figure 3.2: *Four different surface responses generated by varying the compliance of the surface.*

Similarly, a deformation of the epidermal layer is constrained by the thickness of the dermal and muscle layers, and ultimately by the structure of the underlying skull and jaw.

The ability to create objects by layering surfaces allows more complicated behaviors to be created. For example, rather than create an object that is deformable everywhere, a small deformable patch could be placed over the surface where a surface deformation is needed. The combination of a complicated object and a few deformable patches can create the *illusion* that the object is deformable everywhere.

Resistance to stretching and compression

The resistance of a surface to stretching affects the response of surface. A stretchy surface responds to deformations locally, while a surface that is not easily stretched deforms globally as it attempts to maintain a constant surface area. Similarly, a deformable surface surrounding

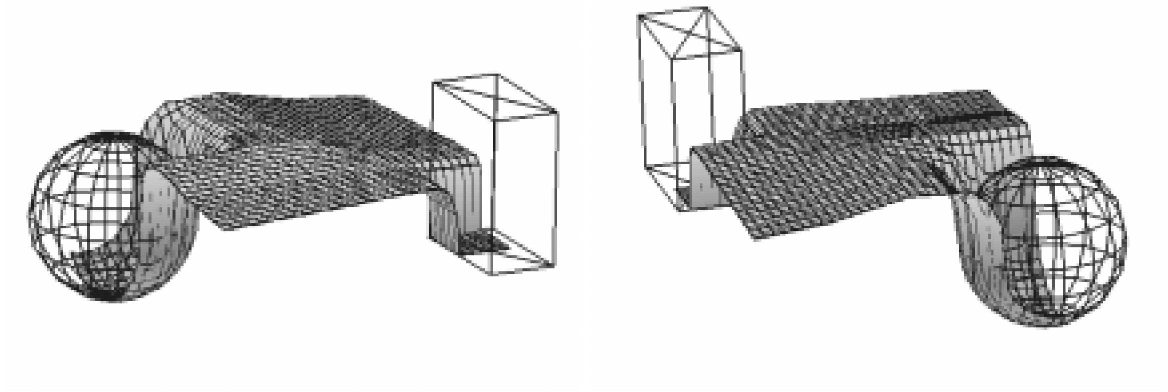


Figure 3.3: *Two views of the same system involving two objects and a surface response that includes bulging.*

an incompressible volume will bulge outwards as it is penetrated. Figure 3.3 shows two views of the local bulging of a surface.

3.3.3 Timing

One of the most important aspects of animation is timing. Motions and reactions must occur at precise relative times, otherwise the illusion of animation is destroyed. Viscosity is more than a damping of velocities, it also creates follow-through, one of the timing elements important to animation [lass87]. Additionally, the animator needs to be able to specify when a particular action starts and stops, and what happens in between.

Viscosity

Viscosity is a measure of the damping, or limiting, of the velocities that a kinematic elastic material deforms at. By damping or clamping the velocities of the points of a deforming surface, the amount of time a collision response takes is extended. Similarly, by damping the recovery of an elastic surface, the “internal forces” restoring shape are resisted. To simulate the inertial mass of the surface, the rate of change — or acceleration — of the deformation of a surface is damped.

Kinematic motion specification

Because the motions of the penetrating objects and the deformable surfaces are specified kinematically, it is easy for an animator to control the timing of a deformation. This does not mean that we totally exclude the use of dynamics, just that our framework is kinematic.

3.4 Non-orthogonality of the surface properties

The surface properties described above have been categorized as controlling the *nature*, *extent*, or *timing* of a deformation. These categories are not completely orthogonal, but the interelationships are easily understood. Though there are multiple ways of creating the same collision response, different properties will generally control different aspects of deformation. Also, if the system runs at interactive rates, an animator can play with the modeled surfaces and quickly build an understanding of their behavior.

Chapter Four

Calculating the collision response

> *OMNIVERSAL AWARENESS?? Oh, YEH!! First you need 4 GALLONS of JELL-O and a BIG WRENCH!! I think you drop th'WRENCH in the JELL-O as if it was a FLAVOR, or an INGREDIENT...or...I...um. WHERE'S the WASHING MACHINES?*
> *— Zippy the Pinhead*

The full algorithm presented in the next chapter calculates the deformations of a surface, using as input a multi-resolution surface, the displacement vectors for the surface, and the interlevel transfer functions. In this chapter we discuss these inputs in detail and then provide a glance at the algorithm used to calculate the deformation of a multi-resolution surface.

4.1 A multi-resolution model of a deformable surface

Other than the properties of compliance and stretching, the surface properties discussed above require only bookkeeping to simulate: the displacement of the points are computed using some function of the previous displacements and the current displacement, and then recorded for future use. This requires only local or bounded information.

Simulation of global compliance however requires global information. As we saw in Chapter 2, global surface energy functionals are very complicated to compute and have not yet been computed in real time. Thus we desire a way to calculate a global or local deformation using only "local" information.

One way of doing this is to use a multi-resolution representation of the surface. Fine resolutions correspond to local features, and coarse resolutions correspond to global features. Similarly, deformations, or displacements, of the points of the fine resolutions result in local deformations.

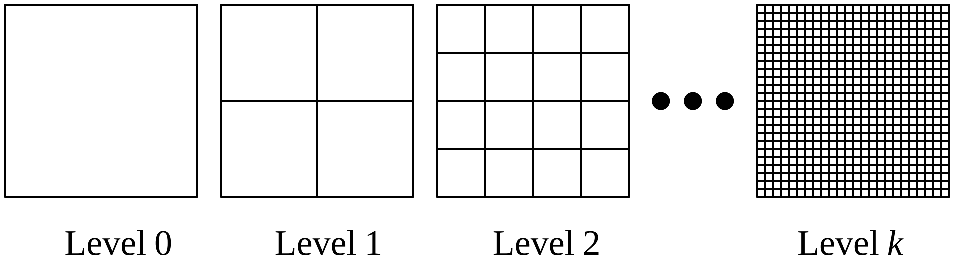


Figure 4.1: *The coarsest level of a multi-resolution model is level 0, and the finest is level k.*

Global deformations are the result of deforming the coarse resolutions. By combining the coarse and fine resolution deformations with an appropriate set of weights a variety of deformation can be generated.

A multi-resolution surface has $k + 1$ levels, the coarsest level is numbered 0, and the finest k, as illustrated in Figure 4.1. Figure 4.2 shows how local details are defined as offsets from the coarser levels. To deform a surface while simulating the behaviors of plasticity, elasticity, and viscosity, the offsets will be changed over time.

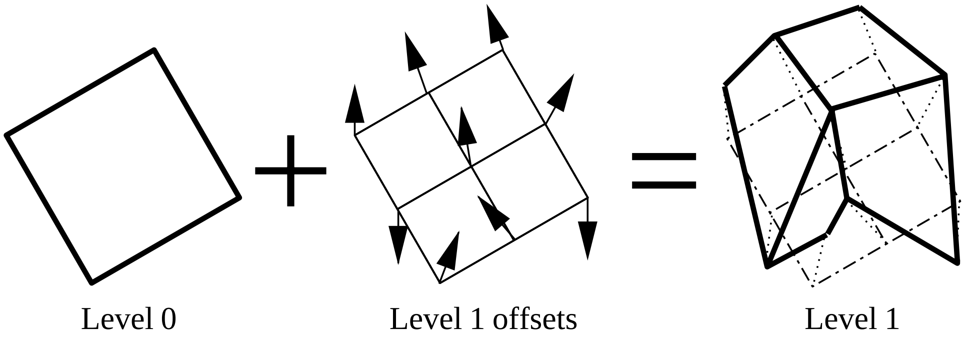


Figure 4.2: *A multi-resolution surface defines its finer levels as offsets from the coarser levels.*

4.2 Computing a collision response

Our algorithm to compute a collision response is as follows:

- The necessary displacements for the finest level to simulate perfect compliance are transferred to the coarser levels using a *restriction* operator.
- Starting at the coarsest level, each of the levels weights its displacements, computes the actual displacements it will accept for this time step, and *prolongates* the results to the next finer level.
- The results are combined with the contribution of the finer level, and similarly prolongated until the finest level is reached.

This process — the restriction of displacements, followed by the prolongation of partial solutions — is the basic idea behind the algorithm.

4.2.1 Penetration detection

When a deformable surface and an object interpenetrate, the first step is to detect the interpenetration and calculate the minimal displacement vectors for the surface. This thesis is not concerned with the actual process of penetration detection or the resulting calculation of the displacement vectors (see the previous papers referenced on rigid body simulation for collision detection algorithms). Thus, the displacement vectors could represent the displacements for a surface controlled by an interactive modeling system.

The displacement vectors represent the local deformation of the surface necessary to provide a completely compliant collision response. The accuracy of the calculation of the displacement vectors controls how accurately the surface will conform to the penetrating object. A higher accuracy surface fit to the object may be attained by locally refining or subdividing the surface.

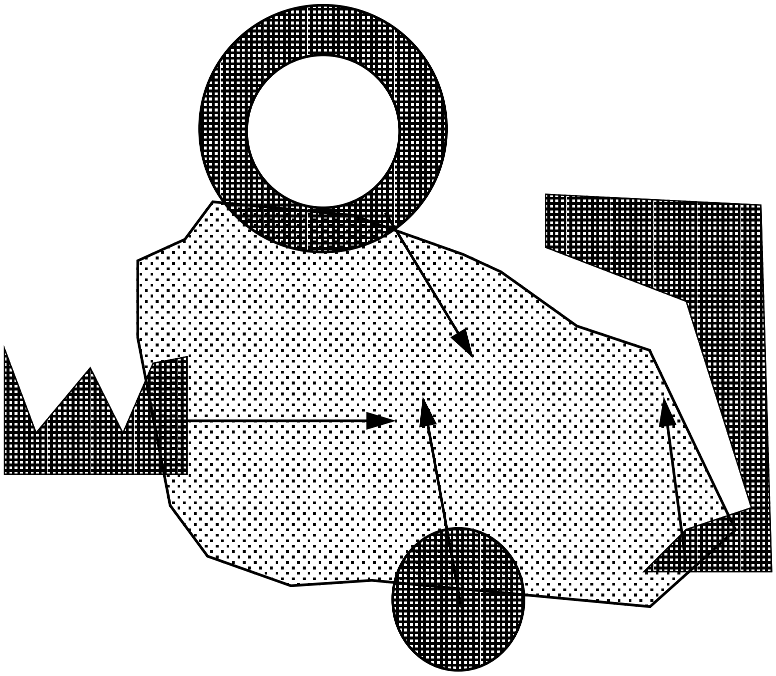


Figure 4.3: *Several special cases for the displacement vector caster created by the convexity of the interpenetrating objects (dark objects) and their position relative to the surface (light object).*

4.2.2 Displacement vectors

For each point of the deformable surface within the penetrating object, a displacement vector is defined. Casting a ray from the current position of the surface point to the furthest exterior of the object within the surface is one method of calculating the displacement vector for a surface sample point. This process is not part of the actual algorithm presented by this thesis. Instead, the displacement vectors are part of the input for the algorithm.

Concave objects can be either decomposed into sets of convex objects, or "ray-traced": starting at a sample surface point, advance along the ray until exiting the bounding box of the object and then back up along the ray to the object. This process is repeated for each convex object and the longest (deepest penetration) vector is taken as the displacement vector. Figure 4.3 illustrates several special cases of the surface–object interpenetration requiring special care.

We have experimented with the direction the displacement vectors are cast. The two basic

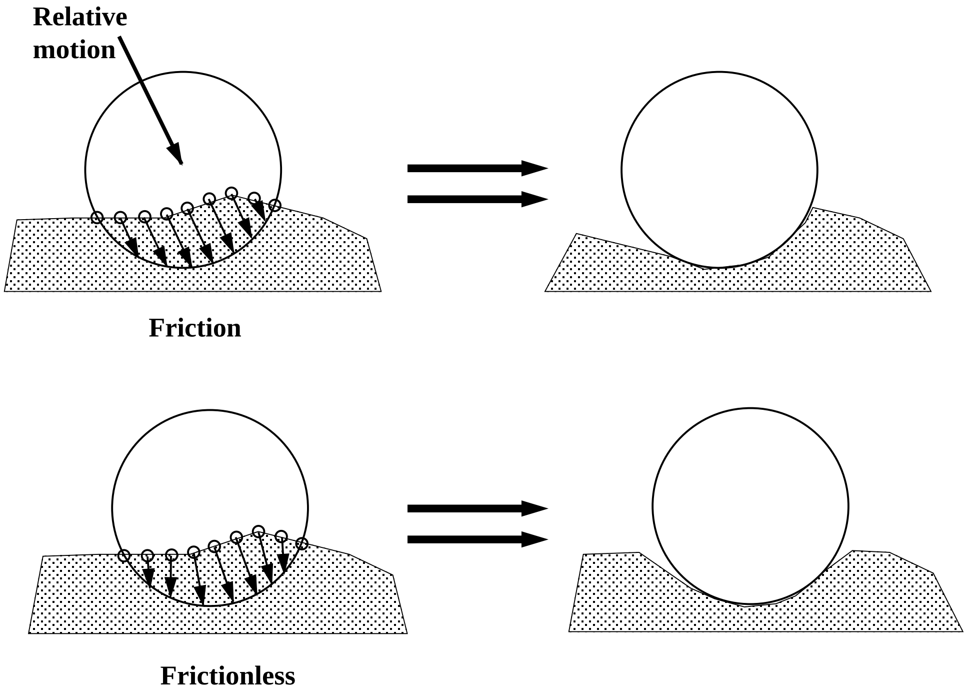


Figure 4.4: *The direction the displacement vectors are cast greatly alters the surface response.*

directions are into the surface — parallel to the surface normals — and along the vector of relative motion of the object and the surface — parallel to the direction the object is "pushing" the surface.

The actual 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. Figure 4.4 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.

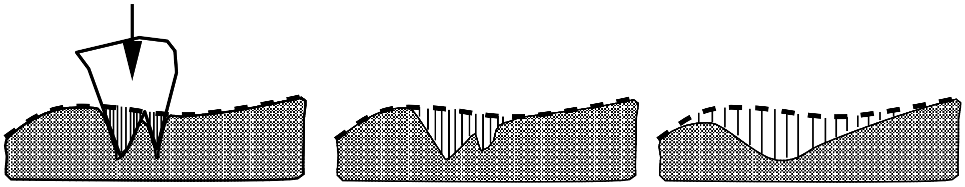


Figure 4.5: *From left to right: the approximations to the displacement vectors create increasingly global and smoother deformations.*

4.2.3 Restriction of displacement vectors

To use the displacement vectors, calculated for the finest level of the surface, to deform the coarser levels, they must be transferred to the coarser levels using a *restriction* operator.

The restriction operator, I_j^{j-1}, takes the displacement vectors for level j of the surface and calculates the displacement vectors for the next coarser level $j-1$.

The filtering operation performed by the restriction operator is non-linear. The restricted displacements should approximate the original vectors, but should also maintain one basic property of the original vectors: compliant surface deformations resolving penetrations *into* the surface.

The restriction operator is given a sparse distribution of displacement vectors defined for the fine level, and computes the displacement vectors for the next coarser level. As the resolution of the levels decreases, the restriction operator propagates the deformation away from the region of contact and across the surface as illustrated in Figure 4.5.

4.2.4 Prolongation of the intermediate results

The other interlevel transfer function is *prolongation*. Prolongation, I_{j-1}^{j}, takes a surface level $j-1$ and refines it to compute the next finer level j.[1]

[1] To remember the difference between I_j^{j-1}(restriction), and I_{j-1}^{j}(prolongation), read the scripts in the order 'sub' then 'super'. That is the direction of the interlevel transfer.

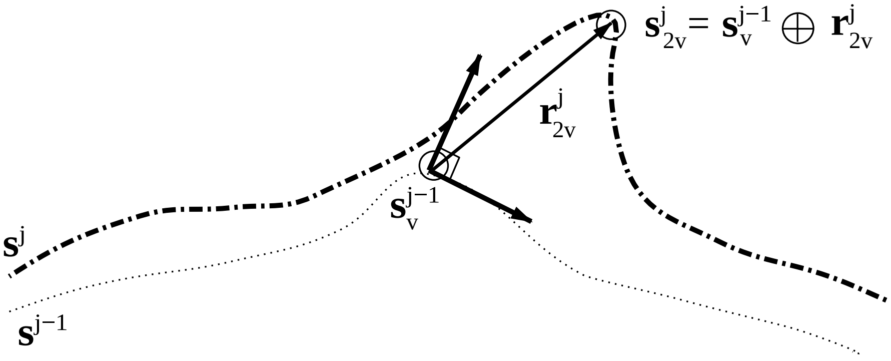


Figure 4.6: *The reference-offset operator creates a coordinate frame on the parent surface used to orientate the offsets of the child surface.*

Prolongation has fewer constraints than restriction, the implementations for this thesis have used linear and cubic interpolation for polygonal surface representations. For a spline formulation, midpoint refinement would be used.

4.2.5 Intralevel offsets

While prolongation computes a finer level from coarse level, it is the intralevel offests that control the different shape of the levels[2]. Offsets represent the deformation of point relative to its prolongated position.

Changes in the offsets of a coarse level cause global changes in the the surface shape, while changes in the offsets of a finer level cause local changes in the surface shape. Additionally we would like to define the offsets so that the local details follow the global changes in shape. One way of computing the offsets to produce this behavior is to use the reference-offset method defined by Forsey and Bartels for Hierarchical B-spline surfaces [fors88].

The reference-offset method uses a coordinate frame attached to the parent surface, $\mathbf{s}^{j-1}$, to define the orientation of the fine level offset vectors, $\mathbf{r}^j$, as illustrated in Figure 4.6. The coordinate frame is attached to the parent surface using a parametrization to locate the parent

[2] No new information is introduced into a surface by simply prolongating it.

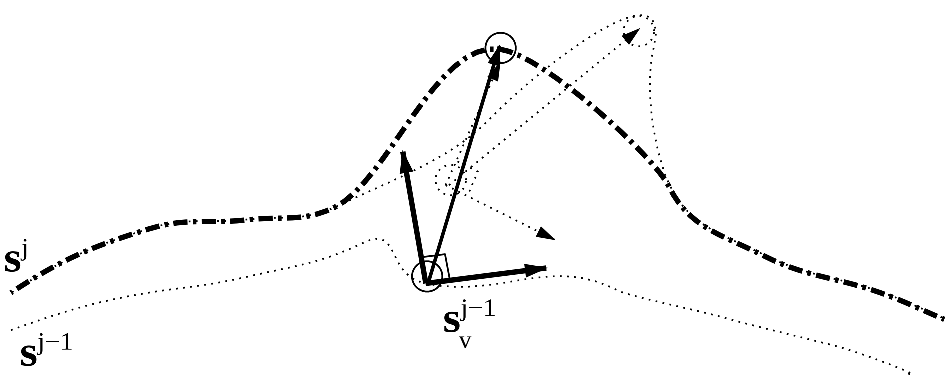


Figure 4.7: *Local details follow the changes in the shape of the coarser levels because of the reference-offset operator.*

vertex $\mathbf{s}_v^{j-1}$ on the surface. Two of the coordinate axes are the parametric directions of the surface at $\mathbf{s}_v^{j-1}$, and the third is aligned with the surface normal. As the parent level changes shape, the orientation of the coordinate frame changes, causing the details to follow the global changes in the surface shape. Figure 4.7 demonstrates how the reference-offset operator causes fine level details to follow the changes in the coarser levels.

4.2.6 Absorption of each of the levels

The restricted displacement vectors are used to calculate the new offsets for each level. This process starts at the coarsest level and moves upwards to the finest level.

The new offsets are calculated by weighting the restricted displacements with a per-level *absorption percentage* that controls the amount of deformation absorbed by a level. Additionally the total change in the position of a point (caused by the deformations of the coarser levels) is clamped to the length of displacement vector for the point. The weights control the extent of the deformation, and the clamping insures the surface will be deformed at any point only as much as necessary.

4.2.7 Contribution of the finest level

After the contributions for all the levels have been calculated, the finest level's offsets then absorb the remaining displacements, ensuring only a single pass through the levels is required to produce a collision response.

4.2.8 Basic algorithm

The basic algorithm to calculate a surface deformation for a single time step, illustrated in Figure 4.8, is:

1. Restrict displacement vectors to all levels.
2. For the coarsest level: weight the displacement vectors by the absorption coefficient and add to the offsets.
3. Apply the offsets to level 0 to deform it.
4. From the level 1 to level $k - 1$
   (a) Prolongate the solution of the coarser level.
   (b) Weight the displacements by the absorption coefficients.
   (c) Calculate the change in the position of the level caused by the deformation of the coarser level(s).
   (d) Modify the offsets by the smaller of the weighted displacements and unabsorbed total displacements.
   (e) Apply the offsets to the level.
5. Prolongate the solution of level $k - 1$.
6. Calculate the change in the position of level k caused by the deformation of the coarser levels.
7. Modify the offsets of level k to absorb the remaining displacements.

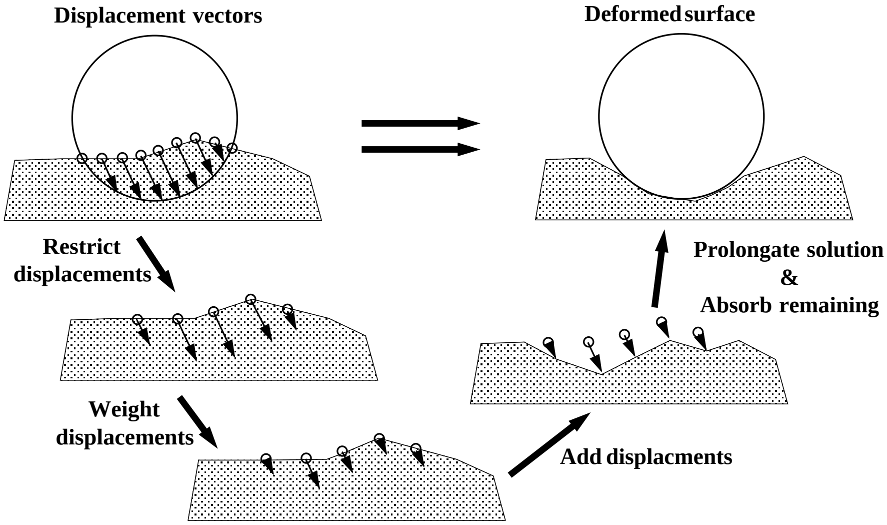


Figure 4.8: *The basic deformation algorithm: displacement vectors are restricted, added to the surface deforming it. The coarse solutions are then prolongated and the finest level absorbs the remaining deformation.*

4.3 Control over the surface

The animator's control over the behavior of the surface is through a set of parameters that determine the nature, extent, and timing of the deformations. The choice of multi-resolution representation, interlevel transfer functions, reference-offset operator, and initial surface definitions also influence the surface behavior. But these choices may not be under the direct control of the animator. Here we discuss the parameters that the animator specifies. We will discuss the specifics of the implementation in the next chapter.

4.3.1 Nature

The nature of a surface is how a deformation affects the shape of a surface after the cause of the penetration is removed. If the surface doesn't change, then it is plastic. If it relaxes completely, regaining it's original shape before the computation of the new displacement vectors then it is perfectly elastic. For free-form modeling, a perfectly plastic surface would be used.

The plasticity of the surface is specified through the per level yield limits. If the length of an offset becomes longer than its level's yield limit, the extra amount is recorded as permanent. By varying the yield limit and shortening the elastic offsets between the calculation of the displacement vectors the surface can be elastic (large yield limit), elastic but moldable (small yield limits), or plastic (zero yield limits).

To create the anisotropic and heterogeneous behaviors, it is only necessary to vary the per level values across each (or any) level. This is called a *behavior map*. In the same way a texture map specifies the appearance of a rendered surface, a behavior map specifies the behavior of a deformable surface. The images in Figure 3.1 were created by specifying a behavior map for the surfaces, and then applying a procedural offset method that displaced points on the surface proportional the value stored in the behavior map for the point. For the figure, a lighter shade indicates a high value, and a dark shade indicates a low value.

4.3.2 Extent

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

By combining the different solutions, the deformation response 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 vectors of each level contributed toward the final solution. Figure 4.9 demonstrates how different absorption coefficients alter the response of the surface.

4.3.3 Timing

Finally, the timing of the deformation is specified by per level damping coefficients that limit the rate of change of the offsets as the surface deforms and relaxes. These, combined with the

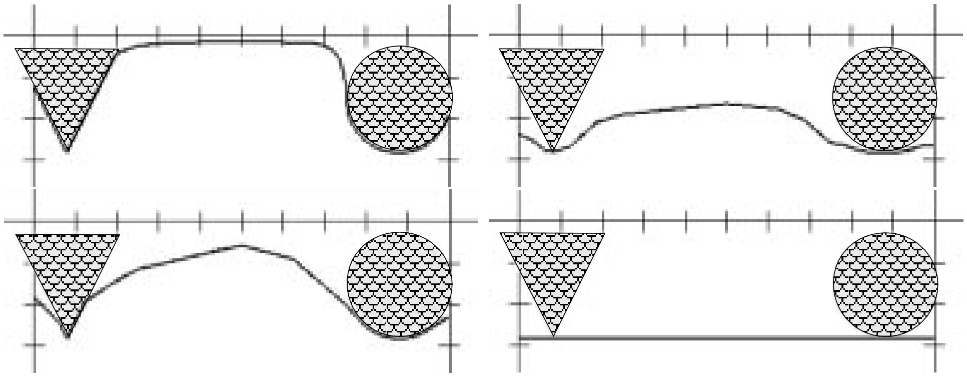
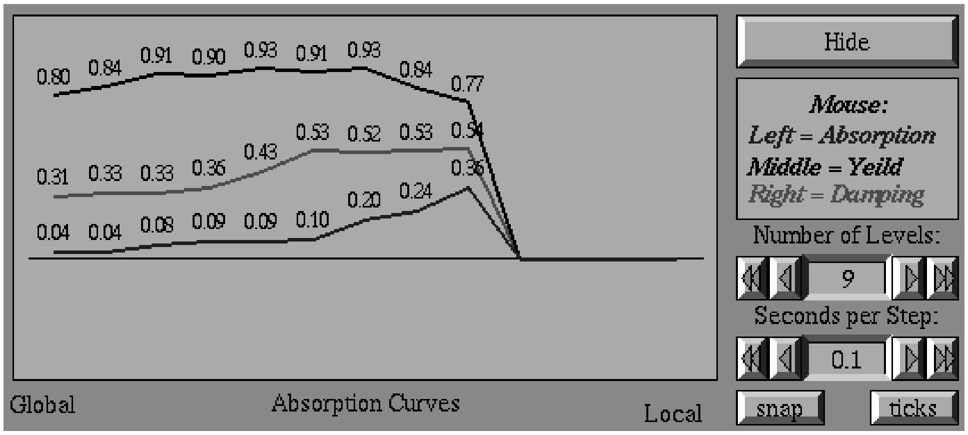


Figure 4.9: *Cross-sections of four different surface responses of a piecewise-linear surface in response to displacement by two objects (finest level is 513 points across). The different responses were created by changing the absorption coefficients.*

relaxation of the surface between time steps, creates the viscoelastic behaviors.

4.4 Review

This chapter has presented how the multi-resolution representation of a surface is used by the deformation algorithm to calculate the deformation of a surface penetrated by kinematically controlled objects. The controls available to the animator to specify the behavior of the surface have also been introduced, Figure 4.10 is a screen shot of a prototype control interface that allows the animator to specify the amount of deformation absorbed, the amount of damping, and an elasticity limit for each of the levels of a surface. In the next chapter the full algorithm, with implementation details of the surface representation, and interlevel operators will be presented.



Mouse:
Left = Absorption
Middle = Yeild
Right = Damping

Number of Levels: 9

Seconds per Step: 0.1

Figure 4.10: *Screen shot of a prototype interface for specifying the behavior of a deformable surface. The three curves specify for the levels of the surface the elastic limit (top), the viscous damping (middle), and the percentage of the deformation absorbed (bottom).*

Chapter Five

Implementation and results

I'm into SOFTWARE!
— Zippy the Pinhead

This chapter presents the full algorithm for calculating the deformations of a multi-resolutions surface as implemented for a prototype system. The surface representation, interlevel transfer operators, and the extensions to the basic algorithm are also discussed.

5.1 Surface representation

A gridded surface representation was used because it was easy to experiment with and implement. However, any surface representation that is hierarchical and refineable could have been used. The finest level, k, of the surface consists of an $N \times N$ grid of vertices. For every level j, its resolution is $(2^j + 1) \times (2^j + 1)$. The coarsest representation, level 0, is a 2×2 grid of vertices.

For any arbitrary vertex, v, on level j, there is a displacement vector, $(\mathbf{d}_v^j)$, an elastic offset vector $(\mathbf{o}_v^j)$, a plastic offset vector $(\mathbf{p}_v^j)$, and the reference vector $(\mathbf{r}_v^j)$. The displacement vector is the total desired displacement the vertex could be displaced by one application of the algorithm.

The elastic offset and plastic offset vectors record the displacement of a vertex from its rest position and are combined with the reference vector to calculate the current position of the vertex. This surface shape is stored as a set of vectors $(\mathbf{s}_v^j)$, as is the prolongated new solution $(\mathbf{n}_v^j)$.

Sometimes it is necessary to refer to the neighborhood of a vertex in a gridded level. When doing so we shall refer to vertex (u, v), its edge adjacent neighbors horizontally $(u \pm 1, v)$ and vertically $(u, v \pm 1)$, and its diagonally adjacent neighbors $(u \pm 1, v \pm 1)$. Within the hierarchy of levels, the parent of a vertex $(2u, 2v)$ in level j is the vertex (u, v) in level $j - 1$.

Each level also has associated with it a set of coefficients that weight the displacements passed from one level to the next (α^j), damp changes in the offset vectors (γ^j and μ^j), and set the yield limit (τ^j), beyond which the elastic offsets become "permanent."

5.2 Interlevel transfer operators

5.2.1 Restriction

The displacement vectors at each level are restricted to form the displacement vectors 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 influences the behavior of the surface, and it is important that it forms an accurate approximation. This is especially true as objects slide across the surface, we want to diminish the effect of the sampling artifacts created as the displacements are restricted to the coarser levels. We have experimented with two simple restriction operators: maximum-displacement and average-displacement, and propose the testing of another operator: full-weighting.

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

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

This operator forms fairly inaccurate approximations, but it does propagate the deformation across the surface quickly as illustrated in Figure 5.1.

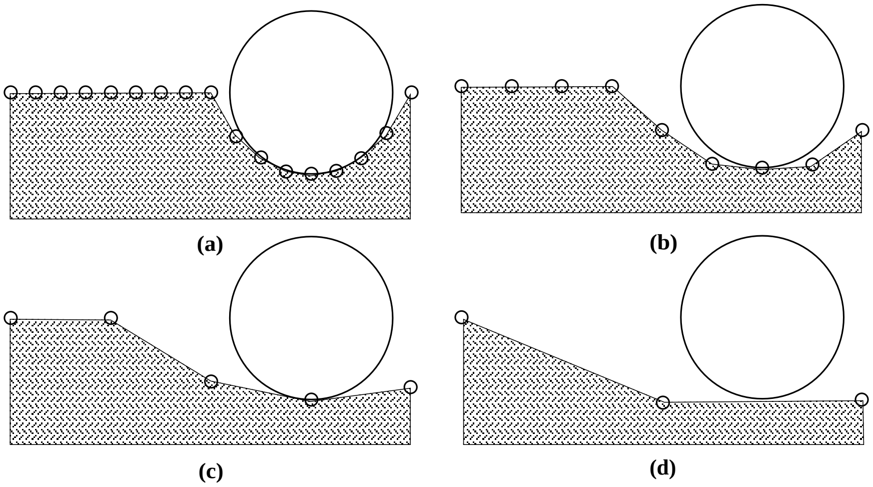


Figure 5.1: *The maximum-displacement operator propagates the deformation across the surface quickly.*

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:

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

This tends to smooth out variations in the field of displacement vectors. For parent vertices whose center child vertex is zero-length, it behaves like the maximum-displacement operator, propagating the edge of contact outwards as illustrated in Figure 5.2.

As future work, other restriction operators could be created by combining the maximum-displacement operator and a full-weighting of the vectors in a larger neighborhood.

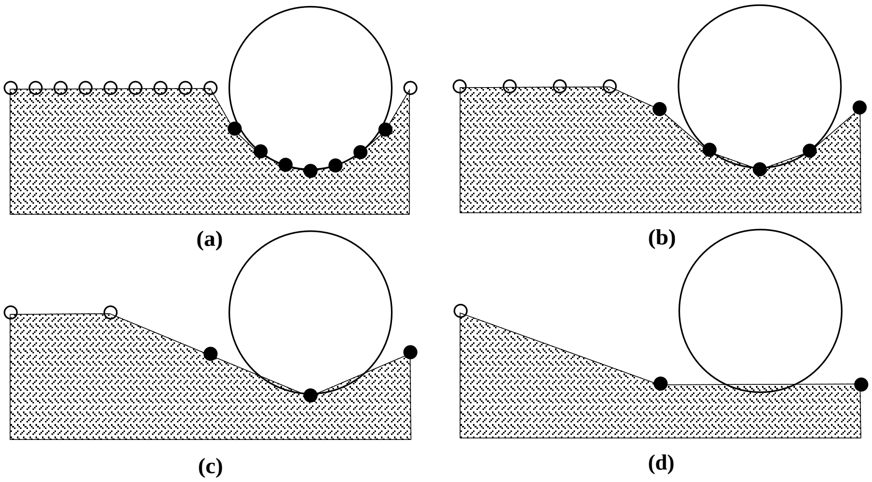


Figure 5.2: *The average-displacement operator creates tighter approximations to the original displacement vectors for the intermediate levels, and propagates the displacements across the coarser levels.*

5.2.2 Prolongation

The partial solutions computed on the coarse levels are prolongated to the finer levels by using linear interpolation:

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

Or by using cubic interpolation:

1. Common vertices: $\mathbf{n}^{j}_{2u,2v} = \mathbf{s}^{j-1}_{u,v}$.

2. New columns: $\mathbf{n}^{j}_{2u+1,2v} = -\frac{1}{16}\mathbf{s}^{j-1}_{u-1,v} + \frac{9}{16}\mathbf{s}^{j-1}_{u,v} + \frac{9}{16}\mathbf{s}^{j-1}_{u+1,v} - \frac{1}{16}\mathbf{s}^{j-1}_{u+2,v}$.

3. Remaining vertices: $\mathbf{n}^{j}_{u,v} = -\frac{1}{16}\mathbf{s}^{j}_{u,v+3} + \frac{9}{16}\mathbf{s}^{j}_{u,v+1} + \frac{9}{16}\mathbf{s}^{j}_{u,v-1} - \frac{1}{16}\mathbf{s}^{j}_{u,v-3}$.

With the appropriate one-sided interpolation performed at the grid boundaries.

5.3 The basic algorithm: Plastic deformations

Given a surface, the system detects penetrations of the surface by other objects and calculates the displacement vectors. Local refinement may be performed to achieve an error bound on the calculation of the displacement vectors while limiting the total number of vertices needed, but was not implemented for the prototype system. The surface, interlevel transfer operators, and displacement vectors are the input to the deformation algorithm. The steps of this algorithm are as follows:

```
1. For $j = k$ down to 1
   Restrict displacement vectors $\mathbf{d}^{j-1} \leftarrow I_j^{j-1} \mathbf{d}^j$.
2. For each vertex $v$ of level 0
   (a) Set new solution $\mathbf{n}_v^0 \leftarrow \mathbf{r}_v^0$.
   (b) Modify offset vectors by displacements $\mathbf{o}_v^0 \leftarrow \mathbf{o}_v^0 \oplus \alpha^0 \mathbf{d}_v^0$.
   (c) Add new solution and offsets $\mathbf{s}_v^0 \leftarrow \mathbf{n}_v^0 \oplus \mathbf{o}_v^0$.
3. For $j = 1$ to $k$
   (a) Prolongate $\mathbf{n}^j \leftarrow I_{j-1}^j \mathbf{s}^{j-1}$.
   (b) For each vertex $v$ of level $j$
       i. $\mathbf{n}_v^j \leftarrow \mathbf{n}_v^j \oplus (\mathbf{r}_v^j + \mathbf{o}_v^j)$.
       ii. $\mathbf{change} \leftarrow \mathbf{n}_v^j - \mathbf{s}_v^j$.
       iii. if $(\mathbf{change} \cdot \widehat{\mathbf{d}_v^j}) < ||\mathbf{d}_v^j||$
            • then $\mathbf{remaining} \leftarrow \left( ||\alpha^j \mathbf{d}_v^j|| - \widehat{\mathbf{d}_v^j} \cdot \mathbf{change} \right) \widehat{\mathbf{d}_v^j}$.
            • if $||\mathbf{remaining}|| < ||\alpha^j \mathbf{d}_v^j||$
              – then $\delta \leftarrow \mathbf{remaining}$.
              – else $\delta \leftarrow \alpha^j \mathbf{d}_v^j$.
       iv. $\mathbf{o}_v^j \leftarrow \mathbf{o}_v^j \oplus \delta$.
       v. Update old solution $\mathbf{s}_v^j \leftarrow \mathbf{n}_v^j \oplus (\mathbf{r}_v^j + \mathbf{o}_v^j)$.
```

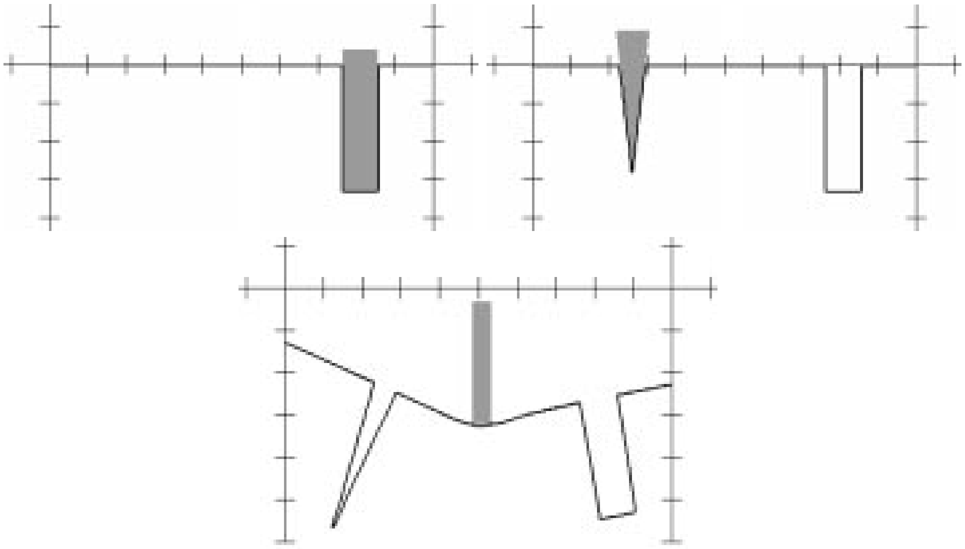


Figure 5.3: *Cross-section of a surface undergoing plastic deformation in three steps. The first two steps are local deformations, and the last is a coarse level response.*

This algorithm does not relax the surface, or decrease the length of the offsets. It simply calculates the deformation of the surface given a single set of displacement vectors.

5.4 Results from the basic algorithm

Figure 5.3 demonstrates a surface being molded by a plastic deformation process. For the first two steps, the surface was made to be very compliant and local features were introduced. For the final step, the surface was made to be somewhat stiffer, deforming only at coarser levels. This final step demonstrates the utility of the reference offset operator and the multi-level representation to create a deformation response for a moldable surface that would be very difficult to produce with other methods such as energy-minimization or a finite-element model. The control over the response of the surface allows it to be molded at multiple scales.

5.5 Extensions to the basic algorithm

The basic algorithm deforms a surface when given a set of displacement vectors for the surface. To add time-dependent behaviors such as elasticity and viscosity, the surface must be relaxed after it is rendered, and before the next set of displacement vectors is calculated:

1. If a penetration exists:
   (a) Calculate displacement vectors.
   (b) Apply deformation algorithm.
2. Output surface.
3. Relax surface.

The simplest relaxation method is to zero the plastic offsets. This creates a perfectly springy and non-viscous surface, that "pops" back into shape as the penetrating object moves away, a behavior inappropriate for animation. Instead the offsets are shortened by a percentage per relaxation step. This percentage, a viscous relaxation coefficient, is specified per-level (μ^j). Before the displacement vectors are calculated a shortened copy of the offsets, is made: $\mathbf{q}_v^j \leftarrow \mu^j \mathbf{o}_v^j$. The reference vectors and these shortened offsets are combined to form a relaxed surface state:

1. Compute relaxed offsets $\mathbf{q}_v^0 \leftarrow \mu^0 \mathbf{o}_v^0$.
2. Set relaxed solution $\mathcal{S}_v^0 \leftarrow \mathbf{r}_v^0 + \mathbf{q}_v^0$.
3. For $j = 1$ to k
   (a) Prolongate $\mathcal{S}^{j-1}$'s to $\mathcal{S}^j$.
   (b) For each vertex v of level j
      i. $\mathbf{q}_v^j \leftarrow \mu^j \mathbf{o}_v^j$.
      ii. $\mathcal{S}_v^j \leftarrow \mathcal{S}_v^j \oplus (\mathbf{r}_v^j + \mathbf{q}_v^j)$.

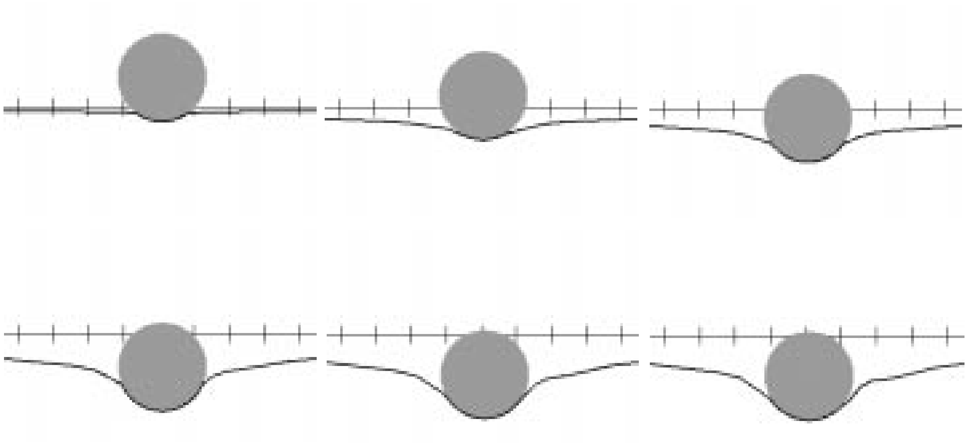


Figure 5.4: *The deformation of a viscously damped surface.*

The relaxed surface, $\mathcal{S}^k$, is used to calculate the displacement vectors for the next time step. For any vertex, if the new $\mathbf{d}_v^j$ is non-zero then the original offset, $\mathbf{o}_v^j$, is used, otherwise $\mathbf{o}_v^j$ is set to the relaxed length $\mathbf{q}_v^j$. The relaxed surface is then used to calculate the new displacement vectors to be used for the next time step.

5.5.1 Damped deformation

In addition, the change of the offsets within the deformation algorithm is damped by a per-level damping coefficient γ^j, replacing step 3(b)iv with

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

Figures 5.4 and 5.5 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. When objects 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^j and μ^j. For example, if a blob was thrown at great speed against a wall, the anima-

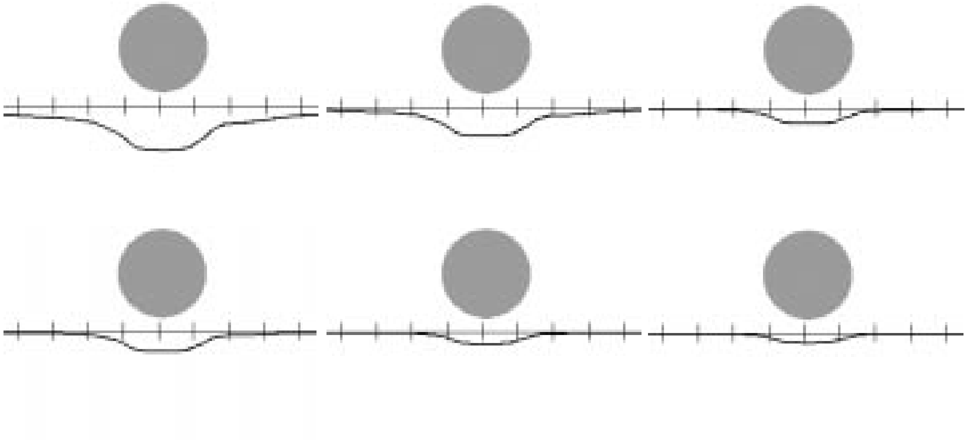


Figure 5.5: *The relaxation of a previously deformed viscous surface.*

tion might call for it either to deform very quickly and permanently, or to hit the wall stiffly, then slowly deform over several seconds. The first blob would appear to be made of soft sticky clay, and the second from slime[1].

5.5.2 Plastic and elastic deformations

The relaxation step changes the deformation algorithm from computing only plastic deformations to computing only elastic deformations. To incorporate plasticity, the plastic offsets ($\mathbf{p}_v^j$) are used to represent the plastic deformation of the surface. As the elastic offsets exceed the yield limits of the surface, the plastic offsets are modified. This changes step 3(b)i into

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

and step 3(b)v into

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

with similar changes to the relaxation steps. When the length of the offset vector grows beyond a yield limit (τ^j) the plastic offset vector is modified to reflect the permanent deformation,

[1] A homogeneous system of particles suspended in a fluid exhibiting non-Newtonian viscous flow: it gets stiffer the more it is stressed, *e.g.* catsup.

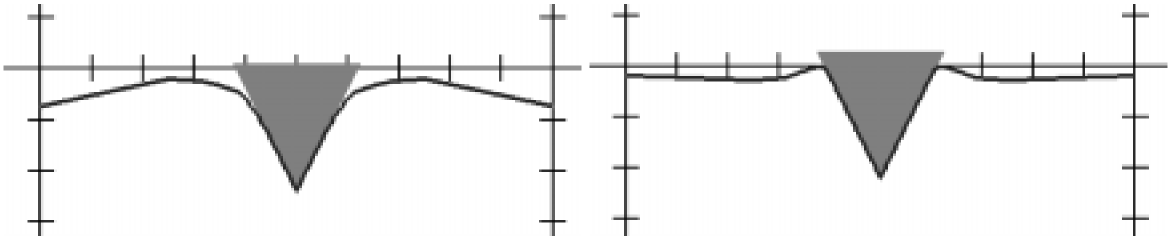


Figure 5.6: *Global and local "volume conservation" is simulated by making some of the coarse (global) or finer (local) absorption coefficients negative.*

requiring the addition of the final step 3(b)vi to the algorithm:

- if $\|\mathbf{o}_v^j\| > \tau^j$ then
  - Amount beyond limit $\mathbf{t} \leftarrow \mathbf{o}_v^j - \tau^j \widehat{\mathbf{o}_v^j}$.
  - Modify plastic offset $\mathbf{p}_v^j \leftarrow \mathbf{p}_v^j + \mathbf{t}$.
  - Clamp elastic offset $\mathbf{o}_v^j \leftarrow \tau^j \widehat{\mathbf{o}_v^j}$.

Bulging around the region of interpenetration is 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 5.6. 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:

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

5.6 Unexplored extensions

The algorithm and its extensions were designed to calculate the deformations of a surface penetrated by rigid objects. 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 strange results, *e.g.*, when a stiff portion of the surface suddenly deforms compliantly. Instead we do exactly what is done in a dynamic simulation: we shorten the step size. 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 work also 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.

When a surface surrounds an articulated skeleton, the surface must continue to enclose the interior objects while deforming in response to exterior objects. The first check is for collisions between the surface and the interior structures, followed by checks for self-surface collisions (at the joints), and finally for collisions with exterior objects.

We can generate reaction forces from a deformation by using a spring model for the elastic offsets. An ideal Hookean spring, $\mathbf{F}(\mathbf{x}) = -k\mathbf{x}$, will work for highly compressible objects. A better approximation of a more incompressible object is a biphasic model for fascia 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 forces (pressure). Or, integrated to calculate the torsional and translational forces upon the body the surface is attached to.

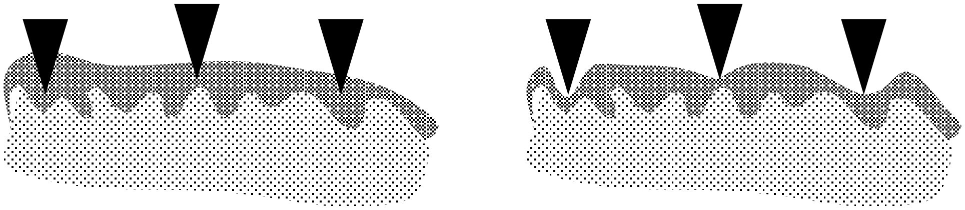


Figure 5.7: *The variation in a compliant layer's thickness (left) results in a surface with a heterogeneous local/global response (right).*

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 5.7 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 5.8 demonstrates how a coarse level offset could react 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 found in [plat88]. Surface area can be conserved by applying a separate multigrid technique such as [pala93] with geometric constraints added to minimize surface shear.

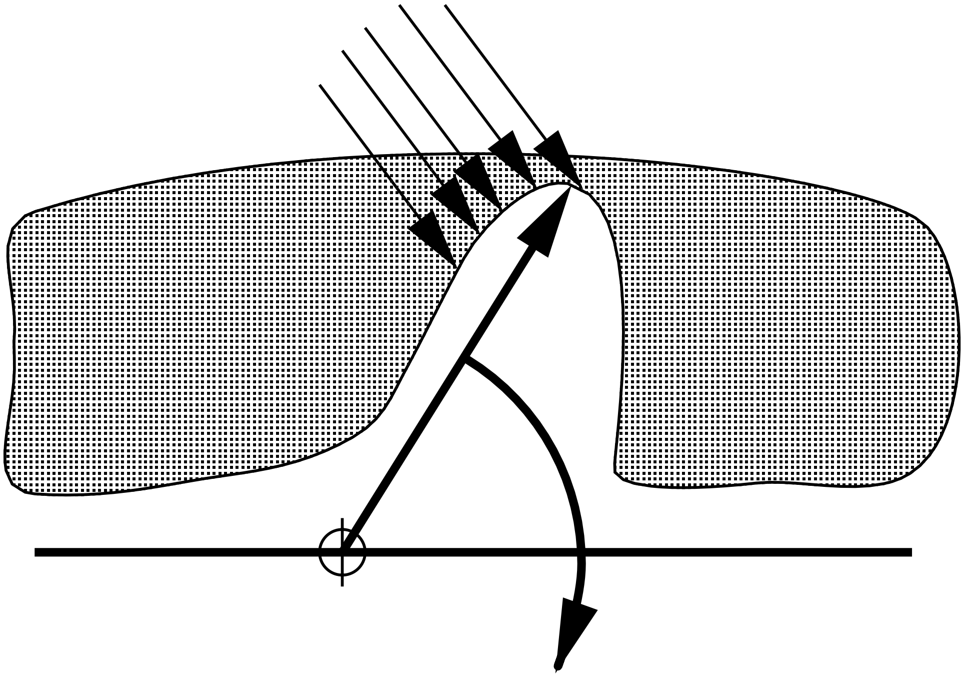


Figure 5.8: *A coarse level offset could react stiffly to restricted displacement vectors. The response would be exhibited as a rotation about an axis in the offset's coordinate frame.*

Chapter Six

Conclusions

This must be what it's like to be a COLLEGE GRADUATE!!
— Zippy the Pinhead

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 surface representation to provide the control and speed necessary for real-time interaction.

The algorithm is numerically stable and requires only the calculation of vector displacements for each time step. Only one application of the algorithm is necessary to resolve a surface-object interpenetration and calculate the response of the surface.

We have been able to simulate the deformation of plastic and elastic surfaces with and without viscous effects. Anisotropic and heterogeneous responses are a simple extension of the per-level controls used in the prototype system. Local and global volume conservation has also been simulated as an extension of the per level absorptance controls.

We have primarily focused on uniform gridded surfaces. Extending this work to non-uniform, or non-gridded hierarchies requires only the proper refinement and interlevel transfer functions. Without these functions, the surfaces can be used as deformable "patches" that control a free-form deformation lattice on an otherwise rigid object. This extends the approach from free-form surfaces to solid objects, better defining the concept of "volume conservation" for a surface.

Layering the surfaces creates complicated non-linear effects that are controlled by varying the thickness, internal features, and behavior of each of the layers.

It is easy to attach a surface to an articulated figure. The reference vectors of the coarsest level are attached to the figure to create a deformable patch, that responds to penetration from external objects. If the patch spans a joint, the interior skeleton can push out the surface as the joint is rotated (*e.g.*, the outside of the elbow).

There is a great deal of future work that can be done to extend the basic techniques presented here. Since the prototype system only allows surface-object interpenetrations, exploring surface-surface and self-surface penetrations is necessary before the approach can be applied to interacting articulated figure animation.

Notations

Vector quantities are written in **bold**.
Scalar quantities are written in *italics*.

Operators

$\mathbf{Ave}(\mathbf{d}^j, u, v)$ Restriction operator, returns the average of the non-zero length vectors in the neighborhood around $\mathbf{d}^j_{u,v}$.

$\mathbf{Max}_{||d||}$ Restriction operator, returns the longest vector from a set of vectors.

I^j_{j-1} The canonical prolongation operator, prolongates a set of values defined for level $j-1$ to level j. Used to form a finer resolution solution from a coarser one: $I^j_{j-1} : \mathbf{s}^{j-1} \rightarrow \mathbf{s}^j$.

I^{j-1}_j The canonical restriction operator, restricts a set of values defined for level j to level $j-1$. Used to form the displacement vectors for the coarser levels: $I^{j-1}_j : \mathbf{d}^j \rightarrow \mathbf{d}^{j-1}$.

$\oplus$ The reference-offset operator.

$\mathbf{a} \cdot \mathbf{b}$ The inner product (dot) vector multiplication: $\mathbf{a} \cdot \mathbf{b} = \sum_i a_i b_i$.

Levels and Vertices

k The finest resolution (level) of a surface, the displacement vectors, $\mathbf{d}^k$, are computed for this level and then restricted to the coarser levels. Typically used as a superscript.

j A resolution (level) of the surface, $0 \leq j \leq k$. Typically used as a superscript.

v As a subscript, any arbitary vertex of a level of a surface.

u, v As a subscript, a vertex of a level of a gridded surface.

Vectors

$\mathbf{d}^j$ The displacement vectors for level j, for $j < k$, then the *restricted displacement vectors*, $\mathbf{d}^{j-1} \leftarrow I_j^{j-1}\mathbf{d}^j$ for level j. If $j = k$, the displacement vectors input to the algorithm.

$\mathbf{n}$ The newly computed solution from the coarser levels, $\mathbf{n}^j = I_{j-1}^j \mathbf{s}^{j-1}$.

$\mathbf{o}$ The elastic offset vectors.

$\mathbf{r}$ The rest offset vectors that define the original shape of the surface.

$\mathbf{p}$ The plastic offset vectors that record the permanent deformation of the surface.

$\mathbf{s}$ The old (last step) solution before application of the algorithm, the new solution afterwards.

$\mathbf{q}$ The relaxed (shorted) elastic offset vectors.

$\mathcal{S}$ The relaxed surface computed by using the $\mathbf{q}_v^j$'s.

$\hat{\mathbf{a}}$ The normalized unit-vector of $\mathbf{a}$.

$||\mathbf{a}||$ The euclidian length of $\mathbf{a}$, $||\mathbf{a}|| = \sqrt{\mathbf{a} \cdot \mathbf{a}}$.

$\delta, \mathbf{t}$ Temporary vectors.

Surface Behavior Controls

α^j The per-level absorptance coefficient, the maximum proportion of a displacement vector, $\mathbf{d}_v^j$, for level j that will be absorbed by an offset ($\mathbf{p}_v^j$ or $\mathbf{o}_v^j$). The absorptance coefficients control the nature and extent of the deformation.

μ^j The per-level relaxation coefficient, the maximum proportion of an elastic offset vector, $\mathbf{o}_v^j$, that will be subtracted from $\mathbf{o}_v^j$ per time step, $\mathbf{q}_v^j \leftarrow \mu^j \mathbf{o}_v^j$. The relaxation coefficients control the nature and extent of the relaxation of the surface following the removal of the penetrating object.

γ^j The per-level damping coefficient, the maximum proportion of a change in a offset vector, $\mathbf{o}_v^j$, that will be added to $\mathbf{o}_v^j$ per time step. The damping coefficients control the viscous response of the surface to interpenetrations.

τ^j The per-level yield limit, the maximum length of an elastic offset, $\mathbf{o}_v^j$, before the corresponding plastic offset, $\mathbf{p}_v^j$, is modified. The yield limits control the plasticity and elasticity of the surface.

Glossary

Absorption: the process of absorbing or of being assimilated.

Absorption Coefficient: the maximum percentage of a *displacement* to be absorbed or assimilated by a *level* of *surface*, α^j.

Anisotropic: exhibiting properties with different values when measured along axes in different directions, e.g. anisotropic light reflection properties of brushed aluminum.

Collision: the mutual penetration of two or more surfaces.

Collision response: the effect of a *collision*, e.g. the resulting forces, changes in velocities, and *deformations* produced by the collision of two or more objects.

Deform: to alter the shape of by *stress* or external influence.

Deformation: alteration of form or shape.

Displacement: the difference between the initial position of something (as a body or surface vertex) and any later position.

Displacement vector: a vector assigned to a specific point on a surface that specifies the desired *displacement* of the point during the current time step, $\mathbf{d}_v^j$.

Dynamics: the aspects of motion requiring the consideration of forces.

Elastic: capable of recovering size and shape after *deformation*, the elastic offsets, $\mathbf{o}_v^j$, store elastic deformations.

Elasticity: the capability of a *strained* or *deformed* body to recover its size and shape after a *deformation*.

Fair: to smooth a *surface*, more uniformly distributing the changes in curvature of the surface.

Finite-Element Model: FEM, a small network of springs and masses used to model an elastic system.

Force: an agency or influence that if applied to a free body results chiefly in an acceleration of the body and sometimes in *deformation* and other effects.

Functional: a mapping from a set of well-defined states to a single real-number; also: a metric or measure of the state of a system.

Heterogeneous: consisting of dissimilar or diverse ingredients or characteristics.

Homogeneous: of uniform structure or composition throughout.

Hysteresis: a retardation of the effect when the forces acting upon a body are changed (as if from *viscosity* or internal friction).

Interpenetration: to mutually to pass into or through.

Interpolate: to estimate values of (a function) between two known values.

Irreversible: not reversible, not "self-reversible", e.g. molded materials are the result of irreversible *deformations*.

Isotropic: exhibiting properties with the same values when measured along axes in all directions. Non-grained.

Kinematics: a branch of *dynamics* that deals with aspects of motion apart from considerations of force.

Level: a specific resolution of a *multi-resolution surface*, j. Level 0 is the coarsest resolution, level k the finest resolution.

Multi-resolution: analysis and synthesis using multiple spatial resolutions or *levels*.

Offset: a stored *displacement*, e.g. the displacement, $\mathbf{o}_v^j$, between the *reference position* of a vertex and it's actual position.

Plastic: capable of being *deformed* continuously and permanently in any direction without rupture. The plastic offset vectors, $\mathbf{p}_v^j$, reflect the plastic deformation of a surface.

Plasticity: the quality or state of being *plastic*; esp: capacity for being molded or *deformed*.

Prolongate: to compute a higher-resolution representation of a *multi-resolution* system by interpolating the known values, the operator $I_{j-1}^j : \mathbf{s}^{j-1} \rightarrow \mathbf{s}^j$.

Reference-Offset Operator: an operator that computes the effect of an *offset*, $\mathbf{o}_v^j$, to a surface vertex, $\mathbf{s}_v^j$, with respect to a coordinate frame attached to the vertex, $\mathbf{s}_v^j$, rather than with respect to a world-coordinate frame.

Reference Position: the default or known position, the *prolongated* position of a vertex combined with the reference vector, $\mathbf{r}_v^j$, for that vertex.

Refinement: to add additional degrees of freedom to a system by interpolating the known degrees of freedom.

Relaxation: the release of stored energy or *strain* when the caused of the *stress* (a force) is removed, e.g. the relaxed *offset* vectors $\mathbf{q}_v^j$.

Restrict: to compute a lower-resolution representation of a *multi-resolution* system by filtering a higher-resolution, the operator $I_j^{j-1} : \mathbf{d}^j \rightarrow \mathbf{d}^{j-1}$.

Strain: the *deformation* of an object under the action of applied forces operators.

Stress: a force exerted when one object presses on, pulls on, pushes against, or tends to compress or twist another object (or itself); esp: the intensity of this mutual force commonly expressed as in units of pressure.

Surface: the exterior boundary of an object, $\mathbf{s}^k$.

Viscoelasticity: having appreciable and conjoint *viscous* and *elastic* properties.

Viscosity: the property of a fluid or semifluid that resists the forces tending to cause flow. Resistance to external forces are controlled by the damping coefficients, γ^j, and to internal forces by the relaxation coefficients, μ^j.

Yield limit: the *displacement* limit, τ^j, beyond which *elastic* deformations ($\mathbf{o}_v^j$) effect *plastic* (irreversible) *deformations* ($\mathbf{p}_v^j$).

Bibliography

[bara90] David Baraff. "Curved Surfaces and Coherence for Non-penetrating Rigid Body Simulation". *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, No. 4, pp. 19–28, August 1990.

[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.

[fung81] Y. C. Fung. *Biomechanics: Mechanical properities of living tissues.* Springer-Verlag, New York, 1981.

[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.

[herz90] Brian Von Herzen, Alan H. Barr, and Harold R. Zatz. "Geometric Collisions for Time-Dependent Parametric Surfaces". *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, No. 4, pp. 39–48, August 1990.

[koen90] Jan J. Koenderink. *Solid shape.* MIT Press, Cambridge, Massachussets, 1990.

[lass87] John Lasseter. "Principles of traditional animation applied to 3D computer animation". *Computer Graphics (SIGGRAPH '87 Proceedings)*, Vol. 21, No. 4, pp. 35–44, July 1987.

[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.

[merr76] Merriam-Webster. *Webster's seventh new collegiate dictionary.* G. & C. Merriam Co, Springfield, Massachussets, 1976.

[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.

[moor88] Matthew Moore and Jane Wilhelms. "Collision Detection and Response for Computer Animation". *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, No. 4, pp. 289–298, 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.

[snyd93] John M. Snyder, Adam R. Woodbury, Kurt Fleischer, Bena Curring, and Alan H. Barr. "Interval methods for multi-point collisions between time-dependent curved surfaces". *Computer Graphics (SIGGRAPH '93 Proceedings)*, pp. 321–334, August 1993.

[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.