A Multilevel Approach to Surface Response in Dynamically Deformable Models

Larry Palazzi* David R. Forsey
Imager Research Lab
Department of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, B.C., V6T 1Z4 Canada
{palazzi|drforsey}@cs.ubc.ca

Abstract

Discretized representations of deformable objects, based upon simple dynamic point-mass systems, rely upon the propagation of forces between neighbouring elements to produce a global change in the shape of the surface. Attempting to make such a surface rigid produces stiff equations that are costly to evaluate with any numerical stability. This paper introduces a new multilevel approach for controlling the response of a deformable object to external forces. The user specifies the amount of flexibility or stiffness of the surface by controlling how the applied forces propagate through the levels of a multi-resolution representation of the object. A wide range of surface behaviour is possible, and rigid motion is attained without resort to special numerical methods. This technique is applied to the displacement constraints method of Gascuel and Gascuel [5] to provide explicit graduated control of the response of a deformable object to imposed forces.

1 Introduction

The quest for visual realism has been a major driving force in computer graphics research for many years. This realism applies to problems in global and local illumination, as well as problems in specifying geometry of objects and their interactions with other objects.

Modeling is hard, and modeling the dynamics of objects that deform and interact with other objects over time is even harder - doubly so if physical validity and/or real time interaction is required. In situations such as commercial animation where control is more important than physical validity, certain simplifying assumptions can be made to decrease the amount of computation required while retaining specific key features of the model's behaviour. Even so, models based upon even simplified models of real-world physics are often difficult to configure and the complex dynamics of the interacting components makes it difficult to elicit specific behaviours from the system.

For the purposes of this paper, a *local* response of a surface to an imposed force is defined as one where forces act upon individual elements of the surface definition (such as point masses) and any *global* response or overall change in surface shape arises through the propagation of the reaction of the affected elements with adjoining components of the surface.

This paper explores an new approach to determining the local and global response of a surface. This technique, when applied to a local model of surface deformation, endows it with the capacity for global response without the overhead incurred by the use of a more complex model of dynamics or a system requiring stiff equations. The goal of our work is to develop a system specifically targeted for animators where speed and control are the essential elements of any dynamic model of deformation, and not physical validity.

For the local model of surface deformation we will use a technique introduced by Gascuel and Gascuel [5] called "displacement constraints". This approach uncouples the constraints connecting components of the surface (in this case line segments) from the effects of any external forces. First, the dynamic behaviour of each component is determined based upon the physical properties of the component and the laws of rigid body dynamics. This is followed by an iterative step that attempts to satisfy the geometric constraints between components. This approach satisfies our definition of local response in that all the calculations operate on the individual components and any global effect must arise by propagation of the effects between elements.

Section 2 presents previous research on deformable and flexible modeling. The displacement constraints algo-

*Supported in part by an NSERC Research Grant

rithm itself is discussed in section 3. In section 4, we present our technique where the degree of local vs. global response of a surface is controlled by using the original displacement constraints algorithm on each level of a multi-resolution surface representation and by controlling how applied forces are propagated between these levels. Conclusions are presented in section 5 and future work in section 6.

2 Related Work

This section presents a brief overview of the body of research dealing with models of deformable surfaces for animation as they pertain to local and global aspects of shape control.

In 1987, Demetri Terzopoulos et al. [15] introduced elastically deformable models to the computer graphics community in which they use a simplified nonlinear elasticity theory and potential energies of deformation to develop deformable models of elastic curves, surfaces and solids. Terzopoulos and Witkin [17] developed a hybrid formulation for deformable models that decomposes an object into a reference component and a displacement component. Terzopoulos and Fleischer [13] extend this work to model inelastic behaviour such as viscoelasticity, plasticity and fracture and in Terzopoulos et al. [16] further extend this to include simulation of thermal phenomena.

In 1991, Terzopoulos and Metaxas [14] introduce deformable superquadrics, used for transforming geometric primitives and deformations into dynamic models. This hybrid model supports both local (based on finite elements) and global (through shape parameters acting upon the underlying superquadric shape) deformations. Metaxas and Terzopoulos [9, 10] present an extension to [14], which they claim is a more physically valid approach. This model supports local and global transformations in a similar manner as in [14]. They extend this work to include parameterized global deformations such as tapers and bends.

In 1988, Platt and Barr [11] use mathematical constraint methods based on physical constraints and optimization theory to model and animate constrained flexible solids. The reaction constraints guide a flexible solid along a path and prevent these solids from penetrating other polygonal objects. Baraff and Witkin[1] present a dynamic model for flexible bodies where deformations take the form of global deformations of an object's rest shape. Objects are deformed by parametric "space warps" applied to all the points on the object.

In 1992, Szeliski and Tonnesen [12] present a surface representation based on "oriented particles", where individual elements have a geometry and react locally to external forces to create and manipulate elastic surfaces. House et al. [8] also use particle systems to model flexible materials. More recently, Gascuel [6] presents a model for deformable objects based on iso-surfaces of potential fields.

In 1988, Barzel and Barr [2] present a modeling system for constraint-based dynamics. This model is similar to the displacement constraints model. Rigid primitive objects are linked together by various geometric constraint mechanisms such as "point-to-point" and "point-to-nail" constraints. Here, the constraint forces are incorporated into the dynamic equations of motion, resulting in a system of coupled differential equations. Another similar approach was presented by van Overveld [18]. As in [2], and also with displacement constraints, rigid objects are connected together by geometric constraints to form a more complex object. This work differs from [2] in that it makes the assumption that the constraint forces and the external forces may be uncoupled. Displacement constraints also makes this assumption, but the manner in which the constraints are satisfied is quite different.

3 Displacement Constraints

Displacement constraints (DC) is a technique introduced by Gascuel and Gascuel [5] to provide deformables surfaces for use in the animation of articulated figures. DC uses a simplified dynamic modeling technique that, although using a very simplified model of physics, attempts to preserve first order linear momentum. Constraints, such as point-to-point, point-to-line and point-to-plane, are geometric, i.e. displacements, rather than actual forces, and are solved using an iterative scheme described in more detail below. By avoiding explicit calculation of constraint forces and by solving the the constraints separately from the equations of motion, DC is presented as being simpler and more efficient than coupled dynamic systems.

Complex objects are defined by connecting primitive rigid objects (typically lines) together. Each primitive object has its own mass and tensor of inertia. Several primitive objects are linked together by geometric constraints (point-to-point, point-to-line, etc.) to form a composite object. During the initial dynamic step of the calculations, each primitive object is treated independently and reacts to external forces according to the physical laws describing the motion of rigid bodies. Constraints are enforced by iteratively adjusting the position and orien-

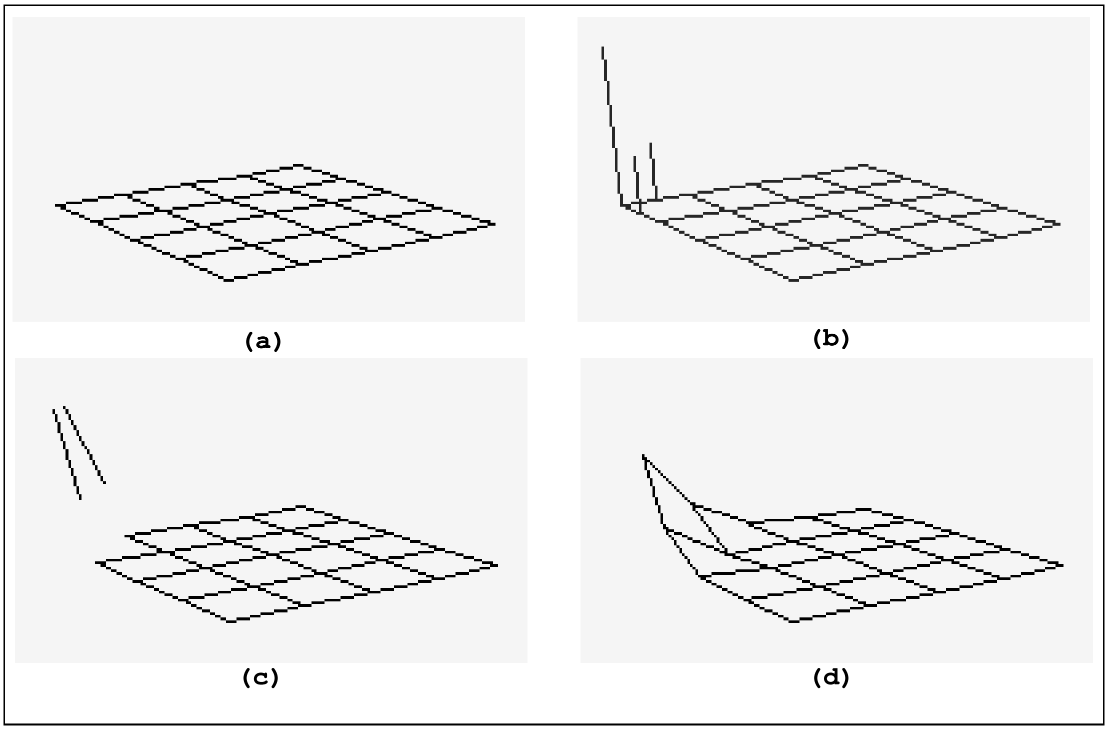

Figure 1: *Steps in one iteration of the Displacement Constraints Algorithm.*

tation of each primitive object (in a manner that attempts to conserve first-order linear momentum) until the error meets a set tolerance or the number of iterations exceeds some maximum.

3.1 The General Algorithm

For an object composed of many primitive objects, the new position and orientation of the composite object are computed using the algorithm in figure 2.

Step 1: Determine all external forces (gravity, contact forces, user-applied forces, etc.) acting on each primitive object.

Step 2: Solve the rigid-body equations of motion for each primitive object [7] as determined by the physical characteristics of the primitive object and the applied forces. Each primitive object is treated independently from all other objects.

Step 3: The constraints are satisfied by iteratively displacing (rotating and translating) each primitive object until the constraints are met (within a set threshold), or until the maximum number of iterations is reached. The magnitude of the translation and rotation applied during one iteration is controlled by the animator.

Step 4: The current linear and angular velocities of each primitive object are calculated with respect to the change in position and orientation obtained in the previous time step.

For each time step:

1. Calculate all external forces.
2. Solve the equations of motion.
3. Solve the constraint system.
4. Update the linear and angular velocities.

Figure 2: *Original DC Algorithm*

Figure 1 depicts one complete timestep of the DC calculation. Figure 1(a) shows the initial configuration of a mesh created by linking line segments with point-to-point constraints. In figure 1(b) an upward force (shown as vertical lines) is applied to a corner of the mesh. Note that a portion of the force is applied to the center of the adjacent segments to imbue linear acceleration to the segment in addition to the angular acceleration induced by the force applied at the endpoint. The distribution of the force between the center and end of a line segment is under the control of the animator. Figure 1(c) shows the result of solving the equations of motion separately

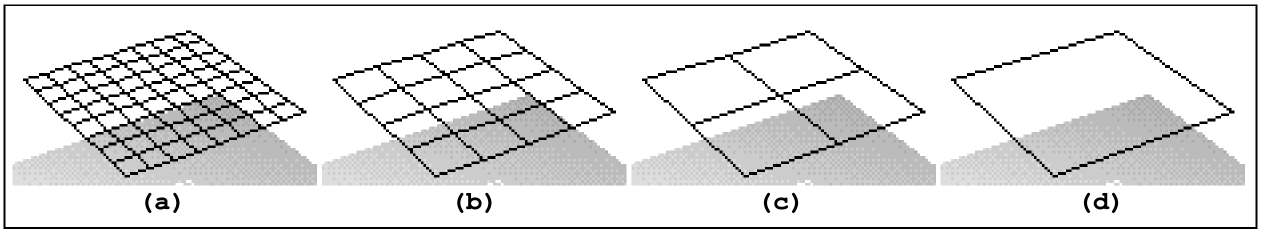

Figure 3: *Multilevel breakdown of a 9×9 mesh*

for each primitive object. Note that at this stage only two of the segments are affected. Figure 1(d) shows the final mesh after several iterations of constraint satisfaction where all the constraints imposed upon the mesh are within tolerance.

3.2 Local Response

Displacement constraints, as with many techniques involving involving a discretized model, relies upon the propagation of a localized surface disturbance to neighbouring elements to produce an overall or global effect upon the surface. As demonstrated in figure 1, an upward force applied to a corner of the mesh does in fact affect the entire surface, but the magnitude of the effect is rapidly reduced away from the site of the disturbance.

This effect is even more pronounced with DC because each sub-component of the object is treated as a completely independent object during the dynamic phase of the calculation. There is no notion of imposing upon the object a global attribute, such as angular velocity, to cause the entire object to spin about its aggregate center of mass. The discrete nature of DC makes it almost impossible to elicit a global behaviour such as a stiff piece of paper falling to the ground, or a leaf being blown by the wind - without some drastic change in the basic formulation.

4 A Multilevel Approach

For an object to respond globally to an external force, there must exist some mechanism that allows an object composed of many elements to be treated as a whole rather than a discretized collection of sub-parts. It is also not sufficient to have an "all-or-none" global response, there must also be specific control over the solution such that the behaviour of the surface in response to an external force can range from "very flexible" (i.e. behaves just as it would using the unmodified DC method) to nearly rigid. By using a multilevel approach inspired by the numerical multigrid method, the DC technique can

For each time step:

1. Determine forces acting on original mesh.
2. Restriction Phase - For all levels starting at finest:
   (a) Apply the restriction operator to transfer positions, orientations, masses and forces through to next coarsest level.
3. For all levels starting at coarsest:
   (a) Apply forces and solve using rigid body dynamics.
   (b) Satisfy the constraints using Gascuel's method to determine final position, orientation and linear and angular velocity at this level.
   (c) Prolongation Phase: use the prolongation operator to pass the solutions (positions and orientations) to the next finer level.

Figure 4: *DC Algorithm with Multilevel Solution*

be directly applied to a multi-resolution model of the surface to control the range of local/global behaviour of a surface. To distinguish our method from the multigrid method, we will refer to ours as a *multilevel* method.

In the basic multigrid method [3], an initial value problem over a rectilinear mesh is solved by decomposing the mesh into several levels of decreasing resolution. This process is called *restriction* and is shown in Figure 3. The equations are solved, starting at the coarsest mesh using an iterative approach such as Gauss-Seidel, and the results distributed to the next finer mesh (called *prolongation*) to act as the initial approximation for another application of the iterative equation solver. This process is repeated for this level and the prolongation step produces the initial approximation at the next finer level and so on until a solution at the original resolution is calculated. In many situations a multigrid formulation will converge faster than the same iterative solver applied to the original mesh.

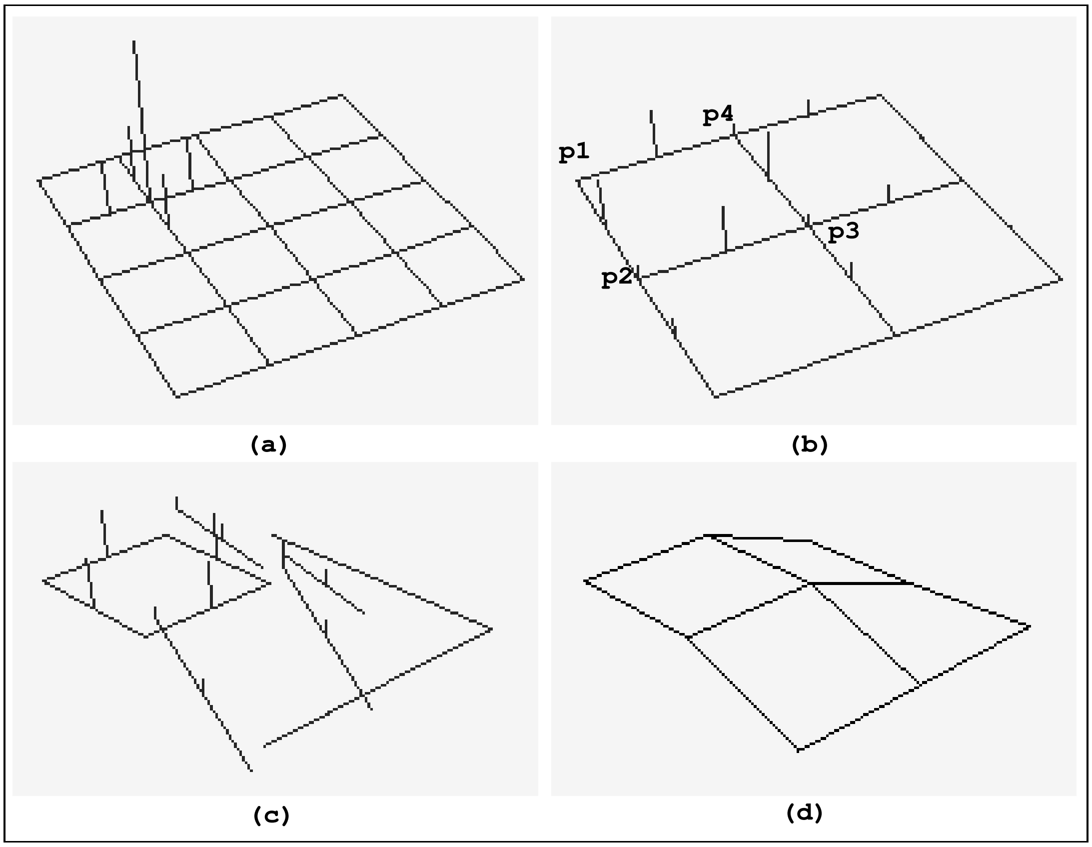

Figure 5: *Restriction of a center mesh point.*

The multilevel method using DC takes a similar approach, not to improve numerical qualities, but to increase the control and expand the range of possible behaviours of the surface. In short, DC applied to the finest level mesh exhibits a purely local response (by definition), and the same algorithm applied to the coarsest mesh (i.e. a rigid body) exhibits a purely global response (by definition). By controlling the application of DC to all levels of a multi-resolution surface representation, the surface will exhibit a response that is neither purely local or purely global. We assume that the spacing between mesh points in a particular level is half the spacing between mesh points in the next coarser level. In other words, we will be working with meshes of size $2^n + 1$ where $n >= 0$.

The general multilevel algorithm is given in figure 4. In the first step, any external forces acting on each line segment in the original mesh are collected. In the restriction phase, a restriction operator (described in greater detail below) transfers positions, orientations, masses and forces from the present level to the next coarsest mesh and so on to all levels in the multi-resolution representation of the surface. In the dynamics phase, starting from the coarsest mesh, any forces that remain are applied to the segments in that mesh and standard equations for rigid body dynamics are applied to each individual segment (the dynamic phase). From this initial configuration the constraints are solved using Gascuel's DC algorithm to determine the final position of all the segments (the constraint phase). A prolongation operator (also described in greater detail below) uses the final configuration to calculate the initial position and orientation of the segments of the next finer level. The results of the constraint phase applied to the finest resolution mesh produces the final configuration of the surface for this time step.

4.1 Restriction

The restriction operator passes the positions, orientations, masses and forces at level $k+1$ in the multi-resolution representation of the surface to the next coarsest resolution at level k.

The positions and orientations for the segments in level k, are obtained directly from the next finer level $k+1$

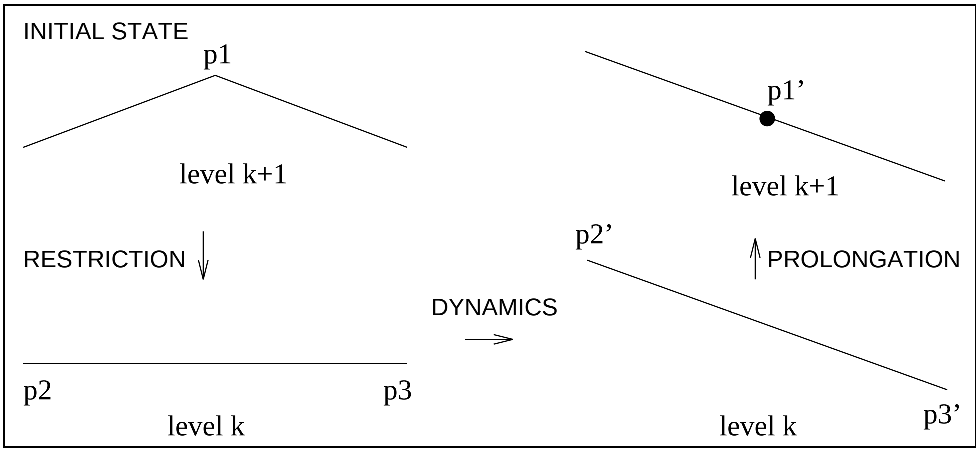

Figure 7: *Prolongation Operator using linear interpolation.*

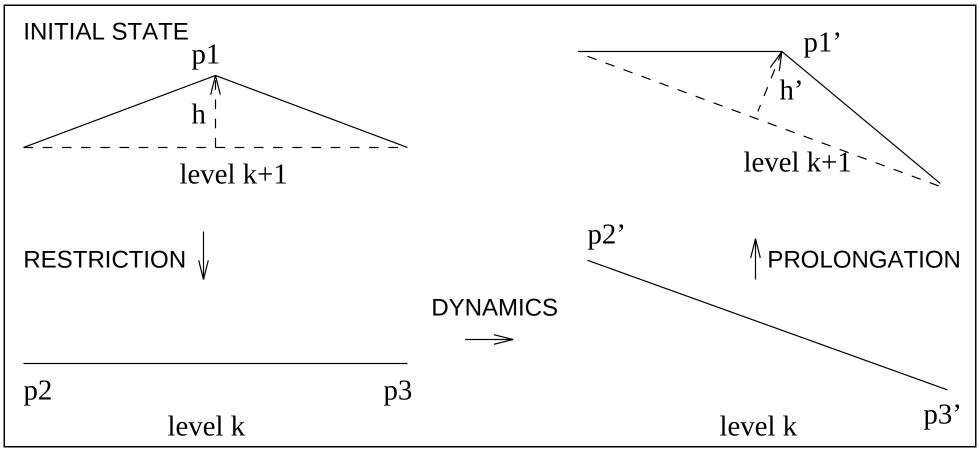

Figure 8: *Prolongation Operator using offset information.*

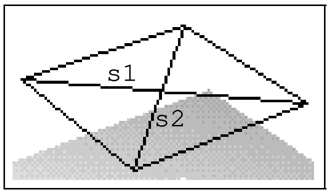

Figure 6: *Adding two cross segments at coarsest level to enforce rigidity*

using the *injection* method [3]. The endpoints of a segment in level k are assigned the values of the corresponding mesh points of level $k+1$ and these two endpoints define the segment's orientation. If r_{k+1} and c_{k+1} are the coordinates (row and column) of a mesh point in level $k+1$ (finer), then the corresponding coordinates, r_k and c_k, in level k (coarser) are defined by:

$$r_k = \lfloor \frac{r_{k+1}}{2} \rfloor, \quad c_k = \lfloor \frac{c_{k+1}}{2} \rfloor$$

In the initial configuration of the mesh, all linear and angular velocities for all segments in all levels are set to zero, but note that the linear and angular velocities are not passed between levels but are retained from the previous timestep.

The coarsest level (level 0) of the mesh merits special attention. In Figure 3(d), the mesh is composed of four connected segments free to move in relationship to each other as long as the endpoint constraints are met. This does not correspond with the notion of a single rigid body. Therefore, we designate this configuration as level 1, and create a rigid level 0 by adding two cross segments $s1$ and $s2$ (Figure 6) to ensure that no bending occurs.

The masses are distributed to each segment in level k

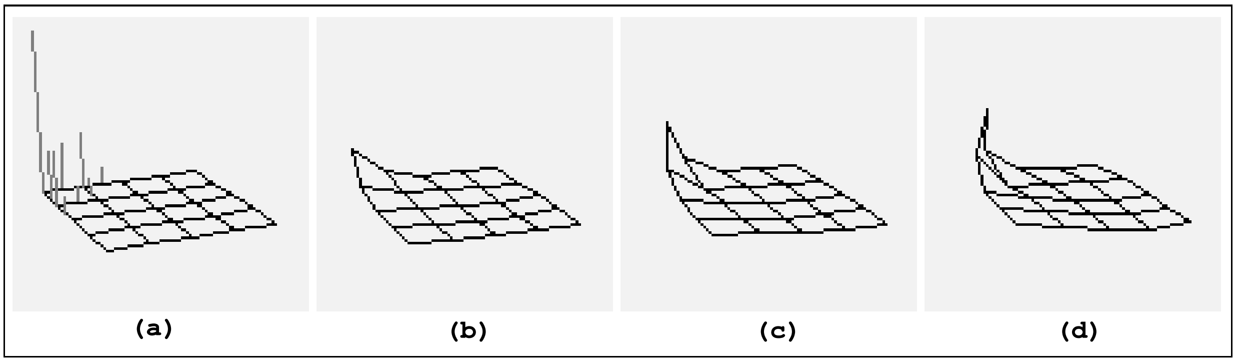

Figure 9: *Local Behaviour - 100% of force acts upon the finest level.*

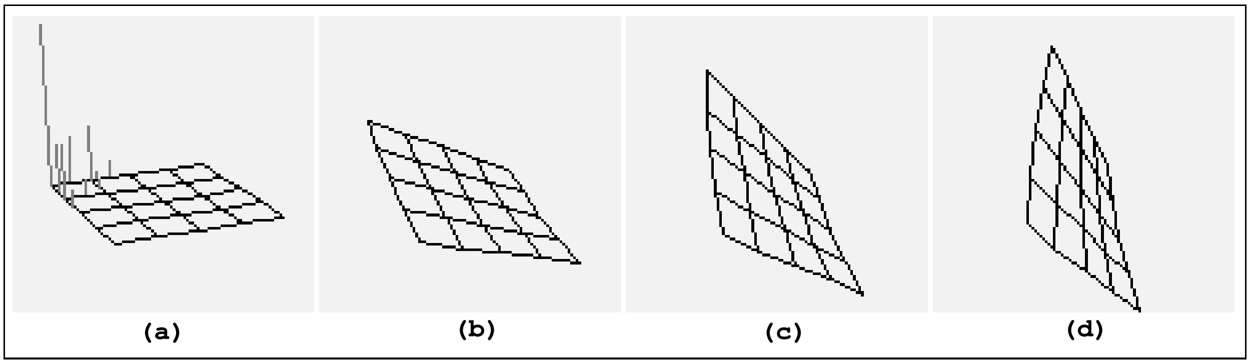

Figure 10: *Global Behaviour - 100% of force acts upon the coarsest level.*

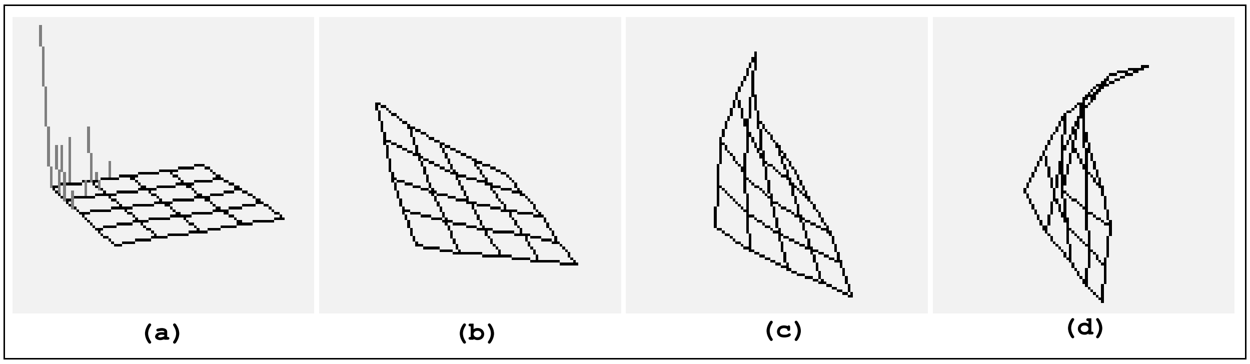

Figure 11: *Force distributed equally among levels.*

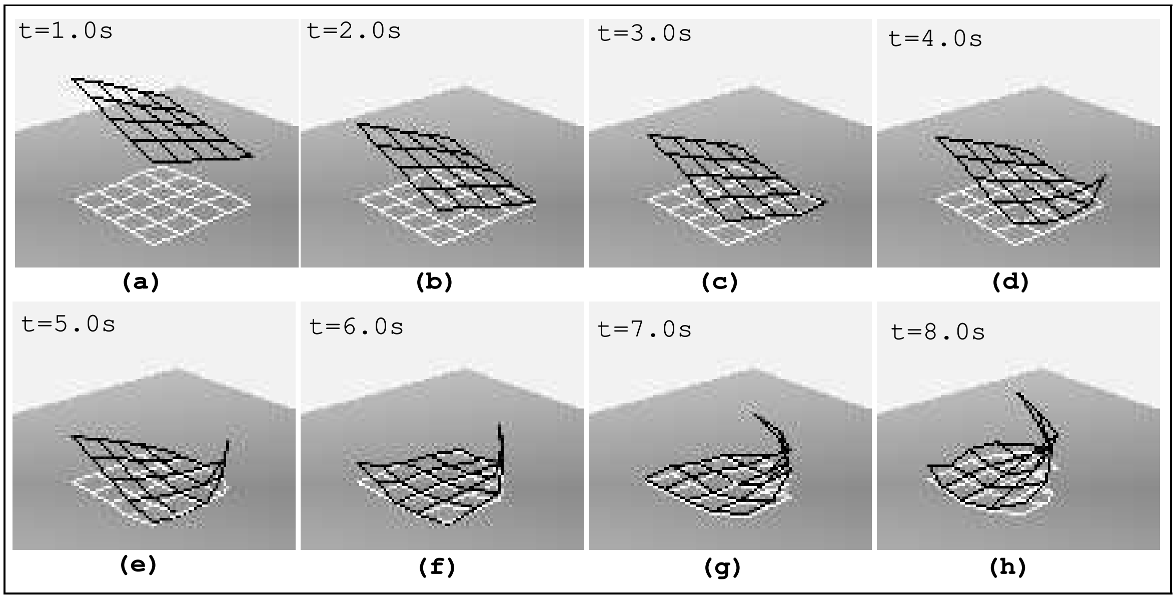

Figure 12: *Flexible mesh bouncing off floor. 100% of the force acts upon the finest level.*

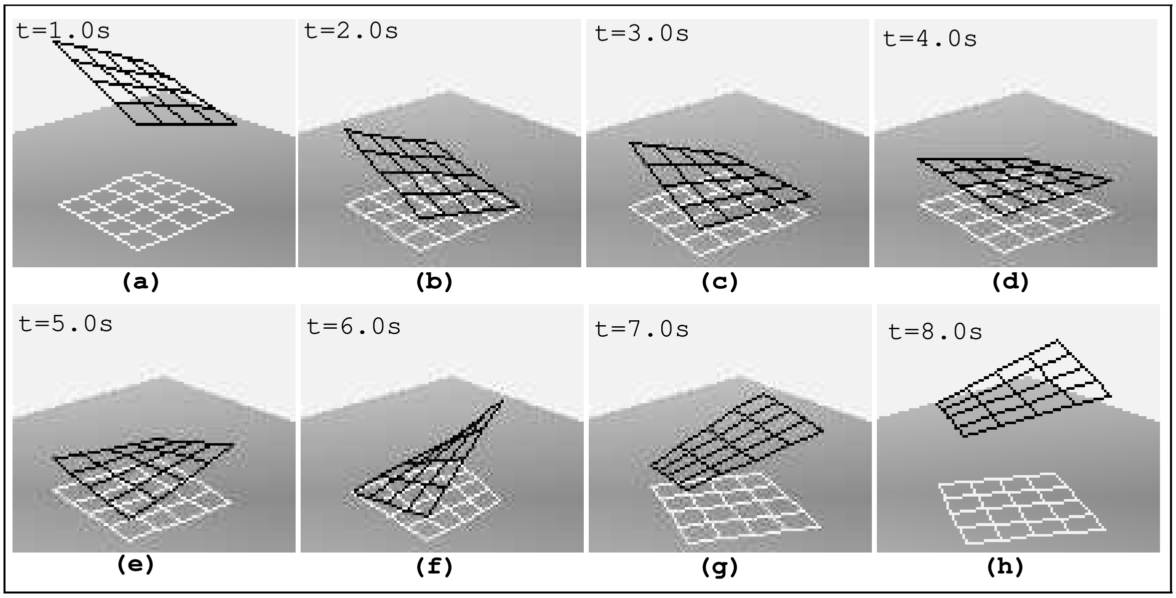

Figure 13: *Rigid mesh bouncing off floor. 100% of the force acts upon the coarsest level.*

based on a weighted average scheme that maintains the total mass at each level (i.e. each level has the same total mass). Currently, the mass of each individual segment is evenly distributed (the inertia tensor matrix for each line segment is the identity matrix). Thus, if all segments in the original mesh are of equal mass, then the mass is evenly distributed to each segment in all levels.

Applied forces are distributed between levels by assigning some percentage of each force applied to level $k+1$ to specified mesh points at level k. Since each level k is half the resolution of level $k+1$, only some of the mesh points will correspond between levels. If a mesh point in level $k+1$ lies between two mesh points in level k, the force is evenly distributed among the neighbouring mesh points of level k as shown figure 5(a) where the force is transferred to mesh points, $p1$, $p2$, $p3$ and $p4$. A portion of each force is also applied to the center of each segment, as described in section 3. The forces on the four segments connecting the mesh points $p1$, $p2$, $p3$ and $p4$, act only on the center of gravity of each segment because equal forces were applied to both endpoints of the segment. Figure 5(c) shows the state of the coarser level after the dynamic phase. The final configuration of the mesh after solving the constraints is shown in figure 5(d).

In the last example, the original force was divided equally between levels. However, the animator can choose to apply any portion of force, from 0% to 100%, to any particular level, or change the correspondence rule to distribute the forces to a wider or narrower portion of the mesh.

4.2 Prolongation

The prolongation operator transfers the solutions (positions and orientations) from level k to level $k+1$ and provides the initial configuration at level $k+1$ for another round of dynamics and constraint satisfaction.

The prolongation operator is intended to be the exact adjoint of the restriction operator (i.e. if level k is unchanged by the dynamic and constraint phases at that level, the original level $k+1$ is restored). However, this is not the case if linear interpolation, one of the standard prolongation operators in multigrid formulations [3], is used. The difficulty arises when a mesh point in level $k+1$ lies between two mesh points in level k. Figure 7 illustrates this problem. Point $p1$ in level $k+1$ lies between points $p2$ and $p3$ in the lower resolution surface at level k. After the completion of the dynamic and constraint phases at this level, prolongation using linear interpolation will generate $p1'$ at the midpoint between $p2'$ and $p3'$. Thus, interpolation eliminates the kink in the original mesh on the left side of figure 7, and in general will tend to smooth out the surface and inhibit localized responses.

Linear interpolation is quite reasonable in a standard multigrid setting where it simply provides the initial values for the next round of calculation, but this is not the case in our intended application for animators where shape preservation may be crucial.

To alleviate this situation we employ the *reference plus offset* form developed in [4] to represent the surface at each level. In this formulation the position of $p1$ is coded as an offset vector, $\overline{h}$, from the midpoint of segment $\overline{p2p3}$ (note that this will not always be a right angle). The restriction operator is unchanged, but during prolongation, (Figure 8) the offset vector $\overline{h}$ is used to determine the position for the new mesh point, $p1'$. This formulation will preserve shape for any affine transformation of level k and will tend to retain local details during the simulation.

Quaternions are used to update the orientation of the offset vectors during the prolongation phase. For example, the rotation between segments $\overline{p2p3}$ and $\overline{p2'p3'}$ is converted to a quaternion and applied to $\overline{h}$ to get $\overline{h}'$. The cross product of the two line segments defines the axis of rotation, $\overline{r}$, and the magnitude of $\overline{r}$ is set to the angle between the two segments.

4.3 Example Animations

Figures 9, 10 and 11 contain frames taken from four separate animations. The error tolerance used when solving the constraints is set to 0.4% of the length of the smallest line segment; gravity is turned off and the animation is initiated by applying an upward impulse force (for one timestep) to one corner of the mesh. The initial surfaces are identical in terms of their physical properties, but the distribution of the forces between levels is different in each.

Figure 9 shows the behaviour of the surface when 100% of the force is applied to the finest mesh and 0% to all other levels. This behaviour is equivalent to that calculated using Gascuel's original DC technique.

Figure 10 shows the behaviour of the surface when 100% of the force is applied to level 0 and 0% to all other levels. The surface acts as a stiff sheet and after the first time step has constant linear and angular velocity. No local response is observed.

In figure 11, the forces are distributed equally between all levels in the multilevel representation. The behaviour of the surface is intermediate between that of figure 9 and figure 10.

The final three examples (figures 12, 13 and 14) show a mesh bouncing off the floor using different force distributions. The error is set to 0.4% of the length of a line segment and the only external forces acting on the mesh are gravity and contact forces from the floor. Simulated shadows are included for clarity and are shown in white on the surface of the floor.

In figure 12, 100% of the forces are applied to the finest level. This mesh is very flexible and its behaviour is equivalent to that modeled by the original DC algorithm.

In figure 13, 100% of the forces are distributed to level 0 to produce behaviour characteristic of a rigid surface. Figure 13(b) shows the mesh colliding with the floor and figures 13(c) through (f) show the new angular velocity (counter clockwise) resulting from the collision. In figure 13(f), another corner of the mesh collides with the floor and counteracts the angular velocity.

In figure 14, the forces are distributed equally to all the levels of the mesh producing a behaviour that is intermediate between that of figure 12 and figure 13, with both local and global characteristics.

5 Conclusions

The displacement constraints method has been extended using a multilevel representation of the surface where the distribution of forces within the levels of the surface is used to control the local/global behaviour of that surface. The reference plus offset form used for each vertex in each level of the representation allows surface details to be retained during any global deformations of the surface.

This technique allows displacement constraints, a simple local approach to surface deformation, to respond globally to imposed forces without evaluating or enforcing some global characteristic of the surface.

6 Future Work

The multilevel technique presented here is independent of the formulation used to determine the response of any specific level in the representation, and is compatible with many of the techniques presented in Section 2. Thus displacement constraints could be replaced by a more physically faithful formulation, or one such as [17] which includes the modeling of elastic objects. In this latter case, the reference plus offset form already encodes the rest-shape of the surface at all levels in the representation.

One challenge will be to generalize the multilevel approach to surfaces of arbitrary topology and to surfaces that can split apart, tear, or break into pieces.

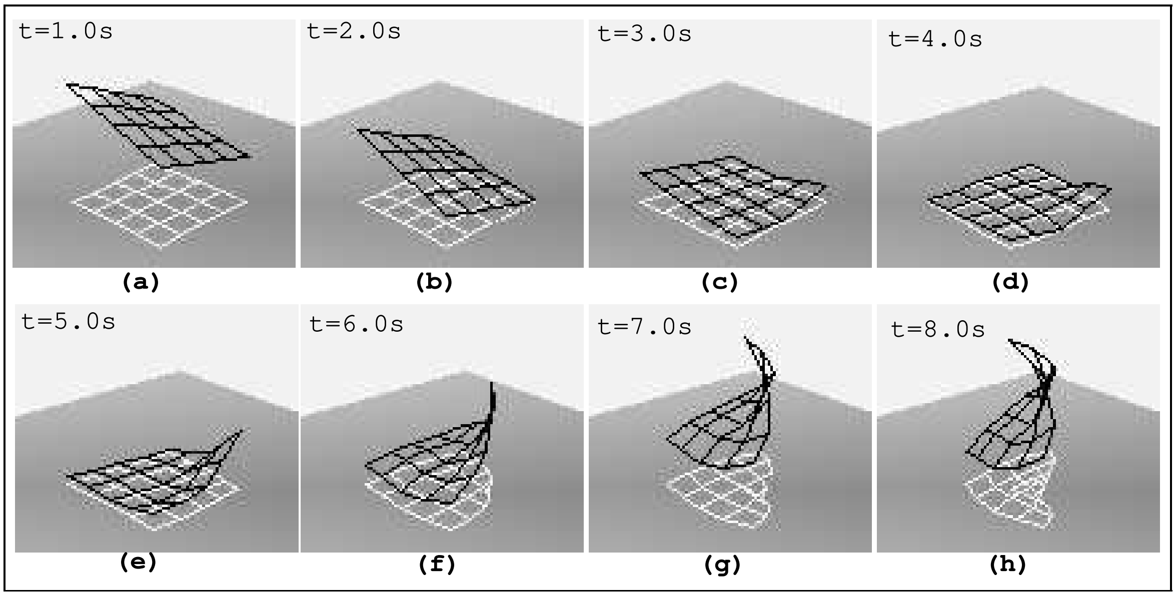

Figure 14: *An intermediate behaviour. The forces are distributed equally among levels.*

References

[1] Baraff, D. and Witkin, A. Dynamic Simulation of Non-penetrating Flexible Bodies. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2), pp. 303–308, July 1992.

[2] Barzel, R. and Barr, A. H. A Modeling System Based On Dynamic Constraints. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4), pp. 179–188, August 1988.

[3] Briggs, W. L. *A Multigrid Tutorial.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987.

[4] Forsey, D. R. and Bartels, R. H. Hierarchical B-Spline Refinement. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(3), pp. 205–212, August 1988.

[5] Gascuel, J.-D. and Gascuel, M.-P. Displacement Constraints: A New Method for Interactive Dynamic Animation of Articulated Bodies. In *EUROGRAPHICS '92 Proceedings*, 1992.

[6] Gascuel, M.-P. An Implicit Formulation for Precise Contact Modeling between Flexible Solids. In *Proceedings of SIGGRAPH 93, Annual Conference Series, 1993*, pp. 313–320, August 1993.

[7] Goldstein, H. *Classical Mechanics.* Addison-Wesley, Reading, MA, second edition, 1983.

[8] House, D. H., Breen, D. E., and Getto, P. H. On the Dynamic Simulation of Physically-Based Particle-System Models. In *EUROGRAPHICS '92 Proceedings*, 1992.

[9] Metaxas, D. and Terzopoulos, D. Dynamic Deformation of Solid Primitives with Constraints. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2), pp. 309–312, July 1992.

[10] Metaxas, D. and Terzopoulos, D. Shape and Nonrigid Motion Estimation through Physics-Based Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6), pp. 580–591, June 1993.

[11] Platt, J. C. and Barr, A. H. Constraint Methods for Flexible Models. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4), pp. 279–288, August 1988.

[12] Szeliski, R. and Tonnesen, D. Surface Modeling with Oriented Particle Systems. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2), pp. 185–194, July 1992.

[13] Terzopoulos, D. and Fleischer, K. Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4), pp. 269–278, August 1988.

[14] Terzopoulos, D. and Metaxas, D. Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7), pp. 703–714, July 1991.

[15] Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. Elastically Deformable Models. *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21(4), pp. 205–214, July 1987.

[16] Terzopoulos, D., Platt, J., and Fleischer, K. Heating and Melting Deformable Models (from Goop to Glop). In *Graphics Interface '89*, pp. 219–226, June 1989.

[17] Terzopoulos, D. and Witkin, A. Physically-Based Models With Rigid And Deformable Components. In *Graphics Interface '88*, pp. 146–154, June 1988.

[18] van Overveld, C. An Iterative Approach to Dynamic Simulation of 3-D Rigid-Body Motions for Real-Time Interactive Computer Animation. *The Visual Computer*, 7, pp. 29–38, 1991.