Interactive Thin Shells – A Model Interface for the Analysis of Physically-based Animation

James Skorupski
Computer Science Dept.
UC Santa Cruz
Santa Cruz, CA, USA
jskorups@cs.ucsc.edu

Zoë Wood
Computer Science Dept.
Cal Poly, San Luis Obispo
San Luis Obispo, CA, USA
zwood@csc.calpoly.edu

Alex Pang
Computer Science Dept.
UC Santa Cruz
Santa Cruz, CA, USA
pang@cse.ucsc.edu

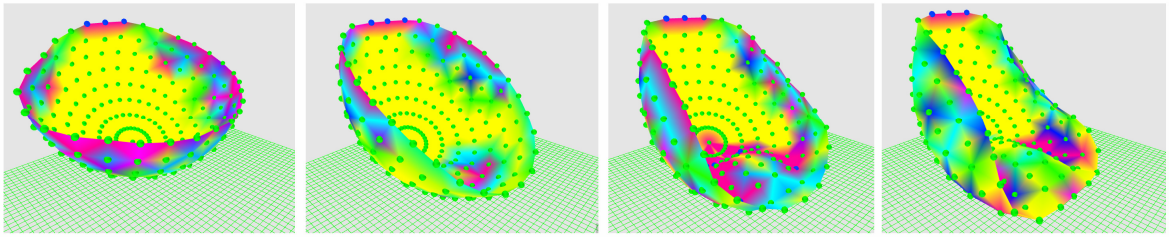


Figure 1 - Simultaneous Experiments. A pseudo-colored hollow cloth bowl with three pinned vertices is simulated collapsing under the force of gravity. Each image depicts the same moment in time for the same mesh, with decreasing membrane and bending energy constants, from left to right.

Abstract

Realism has always been a goal in computer graphics. However, the algorithms involved in mimicking the physical world are often complex, abstract, and sensitive to changes in experimental parameters. We present an interface to a physically-based algorithm, a thin shell animation, which focuses on visualization, experimentation, and control. Through the use of dynamic surface coloring, abstract visual cues, robust user interaction, and full control over the algorithm parameters, our system facilitates experimentation and the process of discovery. The system is targeted at enhancing the user's learning experience by clarifying interactions between various components of many physically-based animations.

1 INTRODUCTION

One of the driving goals in the field of computer graphics is to artificially mimic reality. While the ultimate purpose of the resulting imagery may vary from entertainment to scientific application, the underlying algorithms are all mathematically intensive. Graphics algorithms that are based on real phenomena, such as fluid dynamics, rigid body dynamics, and the transport of light, are known as *physically-based models* [1]. As technology advances, more complex physically-based algorithms continue to develop, and computer scientists wishing to design physically-based algorithms encounter an increasingly varied amount of scientific theory. In addition, many modern computer science students encounter only a limited selection of math courses in their curriculum [3], [6], [11].

We present a model application designed to help the user understand the complex workings of a physically-based algorithm and augment their learning experience with direct, visual interaction, through the use of dynamic coloring, abstract visual cues, mouse interaction, and full control over the algorithm parameters. The system encourages the exploration, discovery, and understanding of the mathematically intense concepts that underlie a physically-based algorithm. Our interface is specifically designed for interacting with a *physically-based animation* algorithm. These types of algorithms facilitate a natural graphical interface, because they mimic physical motion which can be observed and readily confirmed in the real world.

The program we created to demonstrate our interface is called Interactive Thin Shells (ITS). The underlying physically-based algorithm simulates the dynamics of *thin shells*, which are flexible structures that have a high ratio of width to thickness and have an initial three dimensional non-flat shape that affects its energetic reaction to change from that initial shape [9]. The ITS environment allows us to directly demonstrate how our interface can be used to investigate the properties of an algorithm and interact with it in an intuitive and educational manner.

2 RELATED WORK

The thin shell algorithm implementation in ITS is based on a standard physically-based animation model, described in detail in the work of Baraff and Witkin [1]. Our implementation of the physical system is based on a simplified constraint model based partially off of previous work of in the area of thin shells [9] and mass-spring cloth simulation [4]. Fundamentally, the ITS implementation is equivalent to a cloth animation model with some modifications that support non-planar initial configurations and the stiffer internal forces of thin shell materials.

The ITS system also employs backward Euler step implicit integration to progress the simulation, as described by Baraff and Witkin [2]. The implicit method provides for numerical stability that is critical in this particular situation [1]. Thin shell materials typically exhibit very little deformation within the surface of the material itself, and require high resistance to these local changes. Because of this, our simulation will experience regions of high energy in response to deformation, where implicit differentiation allows for reasonably sized time steps [1]. Our particular implementation of the implicit method is based on the work of Dean Marci of the Intel Corporation [12], [13], [14].

The user interface of the ITS environment takes the attributes of the underlying physically-based animation model and provides a simple, intuitive interface that is designed for an individual who wishes to understand the capabilities and theoretical components of the model. Similarly, Burgoon [5] demonstrated an interface to a thin shell simulation based on origami folding and the discrete shells model of Grinspun et al [9]. In general, there is little previous research that addresses an interface design to physically-based animation, however, the field of computer-based scientific simulation provides another source of research.

The general capabilities of the ITS environment are based on the work of Michael Rooks [15], who defines a set of requirements for general visual interactive simulation (VIS) software systems. VIS systems, as defined by Rooks, simulate real world physical phenomena as accurately and completely as possible. In contrast, physically-based animation methods aim to achieve convincing visual realism without a requirement for accuracy. However, the goals for VIS systems remain accurate as it is strongly based on experimentation. Rook describes a complete VIS system as one that facilities (1) Intervention, (2) Inspection, (3) Specification, and (4) Visualization [15]. The ITS environment satisfies each of these requirements by providing direct control of the meshes involved and procession of time (Intervention), access to and customization of all relevant material and simulation attributes (Inspection and Specification), and visual feedback of the resulting simulation and its effect on the dynamics of the thin shell model (Visualization). The ITS environment is designed such that a curious student or computer science practitioner is able to discover all aspects of the thin shell model, including its efficiency, capabilities, limitations, and resulting level of visual realism.

3 THE ITS SYSTEM

To allow the user to experiment with physically-based animation, ITS provides an animation algorithm, user interaction with that animation system, visualization of animation parameters, and a playback system to store and repeat animations. The following section highlights these major features of the interface. For complete details about the entire system, see [16].

3.1 Animation algorithm

To facilitate the experimental capabilities of the ITS interface, a number of animation features are included. The most important feature of the simulation is its dynamic material and global parameters. The ITS interface is able to, at any time in the simulation, allow modification of any of the thin shell membrane or bending parameters, as well as the time step size, gravity force, integration mode, and any environmental collision objects. This modification does not adversely affect the progress of the simulation, and ensures that users can experiment with many simultaneous parameters. Other features of the animation system include constraints on vertices, which can disable up to three degrees of freedom and the ability to switch between explicit and implicit integration modes without any errors in the simulation. To allow for a large number of varying thin shell shapes, the simulation is also able to load arbitrary mesh files. In addition, to introduce a varied environment for the thin shell interactions, the system supports collisions between the thin shell and a sphere or cube objects. Figures 4 and 6 illustrate collision objects and constrained vertices, respectively.

The forces and constraints that act on the underlying physical system in ITS are a simplified version of the internal forces that are present in previous work on cloth and thin shells [2], [9]. The membrane, or in-plane forces in our algorithm are based on the length of edges between vertices and the bending force is a simplified form of the piecewise geometric bending energy in [2], [9]. This bending force simplification, which is based on a simple linear constraint across the shared edge of a pair of triangles, is similar to the bending forces of traditional mass-spring particle-based cloth models [4].

The ITS environment supports adaptive time steps to help ensure stable and real-time interaction at all times. Upon each iteration of the simulation, rapid changes in position or velocity invoke an automatic 50% reduction in the time step size down to a fixed lower limit. If this divergent behavior continues, the simulation proceeds without reducing the time step, and notifies the ITS interface of the problem. However, if a stable iteration occurs, a lowered time step is subsequently increased incrementally, up to a user-defined upper bound.

3.2 User interaction

Live and paused interactions with the simulation are treated independently. When an animation is live, or playing, the user is able to select and move any vertex in any experiment using the mouse. The selected vertex is moved by a spring force between the projected mouse and vertex locations along a plane that is perpendicular to the camera and intersects the original vertex location. This movement method allows for smooth and natural interaction that is compatible with any camera rotations or translations (See Figure 6). To avoid numerical instability, the vertex of interest is not directly moved by the mouse. When an animation is paused, the user may click and select any vertex and choose to "pin" or "unpin" it. Pinning a vertex enforces a constraint with zero degrees of freedom on the vertex of interest, and unpinning a vertex releases any constraints. A pinned vertex cannot change velocity or position in the virtual world. As an example, the rear rim vertices of the bowl in Figure 1 have been pinned using this technique. The user is not allowed to move the positions of any vertices while the animation is paused, because this might introduce numerical instability caused by instantaneous changes in position.

To allow useful comparative analysis, the ITS system supports simultaneous live or paused user interaction of multiple experiments in parallel, since all experiments share the same mesh structure. When a user performs a live or paused interaction with any of these common vertices, the ITS environment attaches simultaneous constraints and forces on all meshes. A screenshot of the process of synchronized experiment interaction is displayed in Figure 4.

3.3 Visualization

The ITS environment provides two visualization enhancements, dynamic force histogram coloring and a temporal cache, to complement and enhance real-time interaction with the physical model.

3.3.1 Dynamic force histogram

In the ITS environment, it is important that the user be able to visually distinguish between the various forces acting in the simulation, so that he or she may readily explore the effects of various types of interactions, and recognize changes in the resulting simulation. To this end, the user may choose to view color representations of the force values for the membrane, bend, or total forces for each vertex within the system. When any of these views are chosen, each vertex is colored according to a histogram with a discrete set of colors that vary in hue attributes, as pictured in Figures 2 and 3. This mapping from the large range of possible force values to a series of discrete colors ensures that resulting coloring model exhibits sufficient variations to be perceived by the human eye. This is important for determining areas of interest and performing comparative analysis. The difference between a traditional histogram and the one in the ITS environment is its dynamic range and force-to-color mapping capabilities, which are accomplished through compression and equalization algorithms, respectively.

3.3.2 Histogram compression

The histogram compression algorithm, outlined in Figure 2, attempts to analyze a histogram and adjust the upper and lower ranges so that the force values are distributed evenly. To distribute the values evenly, if the boundary segments contain more than twice as many values than the average number of values per segment the algorithm iteratively expands the range. Expansion occurs by widening the range boundaries to the average value in the edge range segments. Alternatively, if non-edge buckets in the histogram have more than twice the average number of values in each segment, the range is slowly compressed. The boundary value compression occurs in half segment increments.

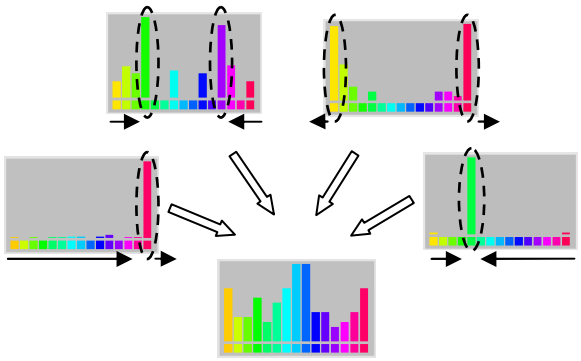


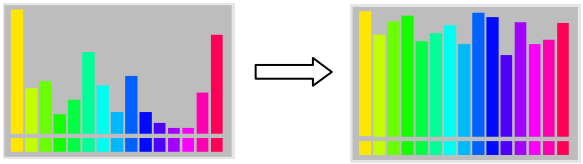
Figure 2 - The ITS histogram compression algorithm.

Due to the fact that the compression algorithm analyzes only the resulting histogram table segments and their distributions during each iteration, our algorithm is simple and fast, but limited in precision. It does not necessarily converge on an ideal range size due to the heuristics used in expanding and contracting the range. As a result of this imprecision, there is a chance that the algorithm will oscillate the distribution of range values about an ideal location. To prevent this, boundary value adjustments are buffered and limited to 50 iterations.

3.3.3 Histogram equalization

Like the histogram compression algorithm, histogram equalization attempts to evenly distribute force values across the entire histogram, to allow for utilization

of the full discretized color spectrum for comparative force analysis. However, this algorithm performs a nonlinear transform on force values based on the cumulative probability distribution of those values. The resulting color values reveal difference in range values, but the ranges are no longer of a uniform size, and comparisons across range segments in the same image cannot be made easily (See Figure 9).



$$f(D_A) = \max(0, round[D_M * (n_k / N^2)] - 1])$$

Figure 3 - Histogram equalization

The histogram equalization algorithm is based on previous work in image processing, and the theory behind its continuous and discrete formations can be found elsewhere [8]. Figure 3 shows the discrete equation that is used in the ITS implementation of histogram equalization. In the equation, D_A represents an arbitrary force value, D_M is the number of color levels in the histogram, n_k is the number of values at force value k or less, and N is the total number of force values in the data set.

When requested, both the compression and equalization algorithms can analyze a single frame of force values or all frames and therefore all force values that have been recorded. The analysis of all past and present frame data results in a histogram that is optimized for an entire run of a simulation, and has the ability to show, on average, an adequate distribution of color for any given frame in the animation. In order to analyze all frames of force data, the temporal simulation cache is accessed.

3.3.4 Temporal cache

The ITS application stores a circular, fixed-size buffer of previous simulation data in a cache so that the user may navigate to a previous time step and analyze the state of the animation. A slider bar in the user interface controls the playback of the cache. The histogram-based force value pseudo-coloring feature may also be enabled when viewing the cache, so that previous force values can be observed and analyzed. The buffer keeps track of the locations of all vertices in the animation, as well as per-vertex force values. In addition, the material parameter settings for each experiment are stored in this cache, as well the time step and gravity settings. In this way, the user is able to see the exact progression of the animation and determine the cause of various behaviors.

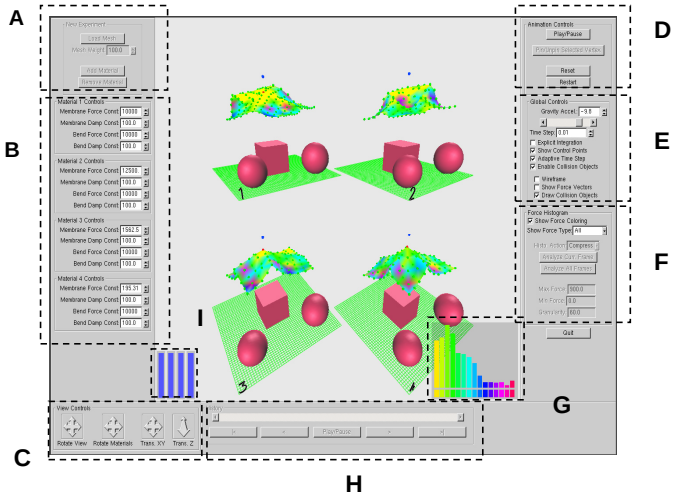


Figure 4 - The ITS Graphical User Interface. The spheres and cube are obstacles with which the material can collide.

4 RESULTS

In this section, we will highlight some of the important features of the ITS environment that allow it to act as a truly free form experimental environment.

4.1 User interface overview

The main ITS user interface is displayed in Figure 4. In this screenshot, a user is interacting with four simultaneous experiments with varying strengths of membrane and bending forces, and has histogram force coloring enabled. Regions A-G contain buttons for user interaction's described in the previous section. For specific details see [16]. Region G highlights the visual representation of the force histogram, as discussed in Section 3. At the bottom, region H outlines the group of controls that allow the user to play back cached animation data, and select any frame of interest for further analysis. Finally, region I marks the visual cues for the current adaptive time step status. Each of these bars represents the size of the current time step for each experiment on screen, in relation to the targeted time step indicated in the global preferences panel on the right side of the screen.

4.2 Animation features

As expected, the explicit mode requires an extremely small adaptive time step, on the order of 0.00001 seconds, $1/100^{th}$ the size of the implicit mode time step, in order to keep the animation stable. Figure 5 demonstrates a set of simultaneous experiments with varying membrane (k_b) constants and bending force (k_m) constants. From left to right, $k_b = k_m = 100000$, 12500, 1562, and 195, respectively. Each displayed frame of the

experiment is shown at the same moment in time, and demonstrates varying reactions to collisions or pinned vertex constraints.

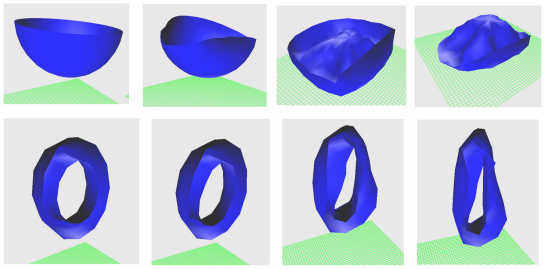


Figure 5 - Two mesh experiments: A falling half sphere impacting an invisible cube and a ring, pinned at a single point, shown at the same moment in time, with decreasing membrane and bending force constants.

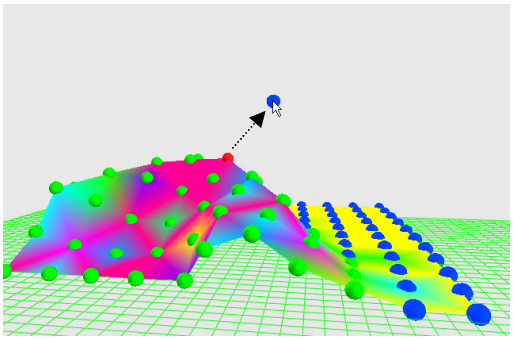


Figure 6 - Live user interaction. The blue control points resting on the plane represent vertices constrained in one dimension.

4.3 User interaction

The screenshots in Figures 1 and 6 demonstrate the paused and live interaction modes, respectively. The hollow bowl in Figure 1 has three rim vertices pinned, while the rest of the mesh is left to succumb to gravity. Each displayed mesh has a varying level of bending force, and is shown at the same moment in time. From left to right, the bending force constants, $k_{b,}$, are 100000, 12500, 8000, and 2000, respectively. As is expected, the bowl loses its structural rigidity when its bending force is reduced. Figure 6 demonstrates live user interaction using a spring force. Here, the user has selected the vertex colored by a red control point, and is dragging the cursor towards the blue control point, which represents the target constraint location. In addition, force coloring is enabled, revealing the redder regions of high force. The arrow in the screenshot shows the direction of force.

4.4 Visualization

In the screenshot in Figure 7, a hollow cylinder lies flat on the floor, and its surface is colored according to the histogram coloring scheme. Force vectors are also visible on its surface, which augment the coloring by indicating the direction of the force currently being viewed.

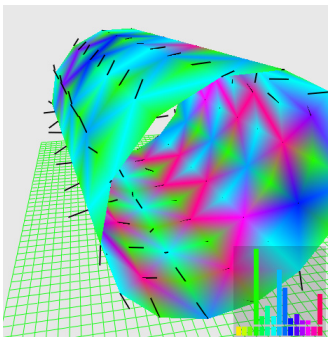


Figure 7 - Visible force vectors and force-based vertex pseudo-coloring

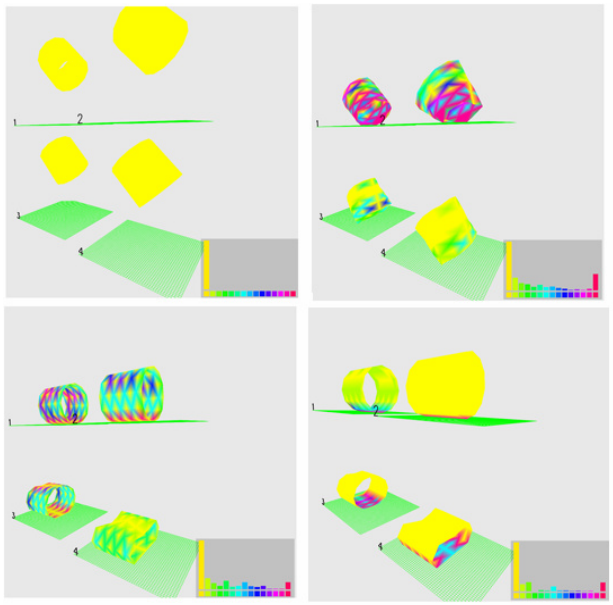


Figure 8 - The progression of forces in four dropped cylinders with varying internal force contributions.

Similarly, Figure 8 shows the progression of force coloring at various frames of an animation. In this example, four simultaneous experiments with a mesh cylinder of varying membrane and bend constants are analyzed, with membrane forces only enabled in the upper left, upper right, and lower left frame, and total forces rendered in the lower right frame. The final frame demonstrates the membrane energies canceling out the gravitational force on the top of the cylinder, and residual vibration between the floor boundary and the bottom of the cylinder introducing a small amount of force on the lower side of the object.

The histogram compression and equalization algorithms are displayed in Figure 9. The plane mesh in this screenshot has its upper left vertex pinned. Initially, the force histogram distribution is insufficient for revealing the force variations on the mesh at this stage in the animation. In the middle frame, the histogram compression algorithm has altered the range as much as it could while maintaining fixed size range segments. In this state, the image has a larger contrast and the variations in the forces across the upper region of the mesh are more apparent, but much of the lower region shows very little visual variation. In the rightmost frame of this figure, the equalization algorithm properly distributes the force values across the histogram, at the expense of fixed color range segment sizes. In this final stage, the force variations are very visible, but judgments about their relative force intensities would be inaccurate, due the nonlinear force value mapping.

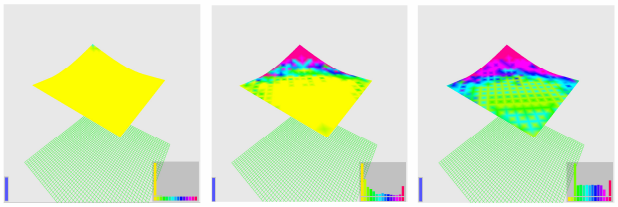


Figure 9 - Histogram compression and equalization. The original histogram range (left), the compressed range (middle), and the compressed and equalized range (right).

4.5 Thin shell model weaknesses

Due to the visualization and control features of ITS, we easily and directly observed a weaknesses in our thin shell model. As mentioned in Section 3, the bending forces in our physical model are simple linear constraints across the shared edge of two triangles. Given a rest condition in which the angle between a pair of triangles is close to 180 degrees, any bending that occurs will not be resisted strongly until the bending angle has extended far from that nearly flat configuration. This occurs because the linear bending constraints are nearly parallel to the pair of triangles, and imbue little force along the normal of each of the triangles until a large amount of deformation occurs. The weakness in this approximation is readily observable within ITS as structural weakness in certain meshes, such as the cylinder mesh in Figure 8. Even with extremely high bending force constants, the cylinder deforms easily during collision or user-initiated interaction, due to the nearly parallel angles between each adjacent polygons in the mesh.

The ITS interface also reveals another inherent weakness which stems from the discrete nature of the animation. This weakness is not unique to our implementation, but extends to any physically-based animation model that relies on a discrete geometric formulation of an object. The weakness is illustrated in Figure 10, where a v-beam is constrained on an entire side and left to hang under the force of gravity. Both corners of the beam should exhibit symmetric force distributions but they do not due to the discrete triangulation of the mesh. This structure results in one corner vertex that has three membrane constraints to neighboring vertices, as seen on the right frame of Figure 10, while the other corner vertex in the left frame has connections with two neighboring membrane constraints and a single, weaker bending constraint across to the neighboring triangle. Therefore, the inherent discrete geometry of the model prevents it from accurately mimicking the symmetric forces that would have resulted from a similar real world experiment with a thin shell material in a similar configuration.

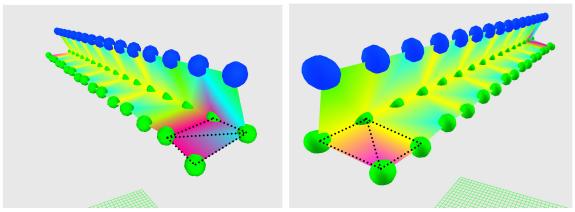


Figure 10 - Unrealistic Forces. Two panels (left, right) show bending force views of two sides of the same experiment on a v-beam with pinned vertices. The forces are asymmetric due to the underlying triangulation of the mesh. The black lines indicate triangle edges.

4.6 User feedback

The ITS interface was tested by several expert researchers working in the field of physical simulation from two different research labs. Users reported that the open, experimental framework encouraged them to play with simulation parameters, which they found to be valuable. In particular, they found the side-by-side experiments with varying parameters and the temporal cache play-back features to be useful when exploring a simulation [10], [17]. A thorough user study is left for future work.

5 CONCLUSIONS AND FUTURE WORK

The Interactive Thin Shells application provides an experimentally-focused, open, informative and very accessible interface to a physically-based animation algorithm. The careful research of Michael Rooks resulted in specific system requirements and framework for VIS applications [15]. These specifications served as a basic guide for the construction of our system. Ultimately, by providing features that allow for thorough intervention, inspection, user-driven specification, and visualization of the underlying physical model, we satisfied each of the VIS requirements in multiple ways, so that the user has a

large variety of useful visualization and interaction mechanisms available at all times.

The ITS visual feedback worked so well, it allowed us to identify weaknesses in the chosen thin shell model. While the bending angle constraint simplification was known to be imperfect, the subtle behavior of weak bending forces at extremely obtuse angles and their results on the animation as a whole were only obvious after carefully exploring simultaneous experiments on multiple meshes while varying specific parameters. In addition, the force coloring patterns in specific pinned mesh configurations were another clear indicator that our simplistic bending force was not a completely adequate model in many cases. The additional discovery of asymmetric forces due to the triangulation of the mesh was another phenomenon that was found only after use of the ITS interface. In this case, the histogram compression algorithm was essential in allowing us to perceive the force asymmetry in the v-beam mesh in Figure 10. Due to the fact that many physically-based animations utilize discrete representations, such as triangles meshes, the ability to discover and analyze the flaws in these approximations is an extremely valuable feature of the ITS interface, and further exhibits the usefulness of the tool in situations outside of thin shell animation.

Future work includes improving the force coloring scheme by implementing a form of intelligent surface shading that does not excessively obscure the force coloring, yet preserves the surface shading. To make the ITS program widely available, ideally, its visualization and analysis components could be generalized into an API for a large assortment of mathematically intensive animation models.

6 REFERENCES

[1] Baraff, D. & Witkin, A. (2001), 'Physically Based Modeling', *Siggraph 2001 Course Notes*.

[2] Baraff, D. & Witkin, A. (1998),Large steps in cloth simulation, *in* 'SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques', ACM Press, New York, NY, USA, pp. 43--54.

[3] Beaubouef, T. & Mason, J. (2005),'Why the high attrition rate for computer science students: some thoughts and observations', *SIGCSE Bull.* 37(2), ACM Press, New York, NY, USA, 103--106.

[4] Breen, D.E.; House, D.H. & Wozny, M.J. (1994),Predicting the drape of woven cloth using interacting particles, *in* 'SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques', ACM Press, New York, NY, USA, pp. 365--372.

[5] Burgoon, R.J. (2005),'Discrete Shells Origami', Master's thesis, California Polytechnic State University San Luis Obispo.

[6] D'Antonio, L.; Baldwin, D.; Ford, F.; Henderson, P. & Wyatt, R. (2002),'Panel: is there too much math in the computer science curriculum?', *J. Comput. Small Coll.* 17(3), Consortium for Computing Sciences in Colleges, , USA, 97--102.

[7] Feynman, C. (1986),'Modeling the Appearance of Cloth', Master's thesis, Massachusetts Inst. of Technology.

[8] Fisher, R.; Perkins, S.; Walker, A. & Wolfart, E. (2003),'The Hypermedia Image Processing Reference', http://homepages.inf.ed.ac.uk/rbf/HIPR2/.

[9] Grinspun, E.; Hirani, A.N.; Desbrun, M. & Schröder, P. (2003),Discrete shells, *in* 'SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation', Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 62--67.

[10] Grinspun, Eitan. Personal Communication regarding lab's use of ITS, November 2006.

[11] Konvalina, J.; Wileman, S.A. & Stephens, L.J. (1983), 'Math proficiency: a key to success for computer science students', *Commun. ACM* 26(5), 377--382.

[12] Marci, D. (2006), 'Simulating Cloth for 3D Games', http://www.intel.com/cd/ids/developer/asmo-na/eng/20413.htm.

[13] Marci, D. (2000),'Real-Time Cloth', *in* 'Game Developers Conference 2000 Proceedings'.

[14] Pritchard, D. (2006), 'Implementing Baraff & Witkin's Cloth Simulation'.

[15] Rooks, M. (1991),A unified framework for visual interactive simulation, *in* 'WSC '91: Proceedings of the 23rd conference on Winter simulation', IEEE Computer Society, Washington, DC, USA, pp. 1146--1155.

[16] Skorupski, J., Interactive Thin Shells - An Interface for the Analysis of Physically Based Animation. Technical Report CPSLO-CSC-06-02, California Polytechnic State University, 2006.

[17] Smith, Adam & Scher, Steve. Personal Communication, May 2007.