Adaptive Mesh Refinement for Use in Cloth Simulation

Wesley Miller Viktor Rumanuk

May 1, 2014

Abstract

In this paper, we discuss the importance of accurate, real time cloth simulations. Additionally, we explore adaptive mesh refinement for complex and accurate real time cloth simulation. We demonstrate the methods used to subdivide the cloth, the way to determine where subdivision is necessary to achieve optimal simulation results, and ways to simulate physically realistic internal and external forces to the cloth and achieve a realistically animating fabric model.

Our implementation covers the subdivision method and where to do so; however, the scope of the project was cut to not fully implement the physical simulation as time constraints did not allow us to completely implement that aspect. Mass calculations were implemented, and the beginnings of the physical systems were modeled, but it was not taken to the point where it is useful to demonstrate. This paper demonstrates that the implementation is robust enough to turn any well-formed triangle mesh into a cloth for animation.

1 Introduction

As computer animated films and three-dimensional video games have become very common, the utility of accurate simulations of physical phenomena has increased drastically. Among these simulations is that of cloth. Accurate cloth simulation is very useful for things such as clothing, flags, bags, parachutes, banners, tent structures, and tablecloths.

Without simulation techniques, animators must hand animate any motion of cloth in films and interactive games and applications, and as these film and game projects become bigger and bigger, the animators in a studio must increase both in numbers and in skill. Clearly, there is a need for techniques to simulate things and remove the need to animate them. When these simulations are inaccurate, however, the audience will not be able to fully enjoy the experience as much because lack of realism in physical behavior stands out very much to humans. Furthermore, if simulations are accurate but not efficient enough, they cannot be used in video games and other interactive applications because they cannot be rendered in real time. This presents an interesting problem to solve, balancing realism and efficiency in a method that will create realistic cloth motion without necessary user intervention. Further complicating the problem is that there are many external forces that can effect fabric motion, such as gravity and wind resistance, in addition to the internal forces which must be accounted for.

2 Motivation

After implementing the mass-spring method of cloth simulation shown by Provot [1], we felt that it was significantly lacking in desired realism because of the lack of any collision detection and because of the fact that it was only capable of using a regular grid pattern. As robust as the system is, if the initial cloth mesh is only a very coarse mesh of apparent quads, it will look more like a folded piece of paper than a rounded draped cloth. When looking through papers on different areas of research in graphics, we decided that this was one of the most interesting areas of work to us and that this method in particular was something we wanted to work on because it seemed to fix all of the problems we had with the Provot method while still being performant enough to allow real time rendering.

3 Related Work

Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior by Xavier Provot [1] presents a mass-spring method of points on a regular grid in the mesh and introduces a correction for springs extending beyond a realistic threshold. This is a precursor method to the one used in this paper.

We explored *Adaptive cloth simulation using corotational finite elements* by Jan Bender and Crispin Deul [2] for the adaptive refinement scheme. In the paper, the authors discuss a method for refining any triangular mesh on a per-triangle basis and restructuring edges in certain cases to maintain a highly regular form. Mass is adaptively calculated at each vertex, based on the areas of surrounding triangles. The core simulation method relies on solving a non-linear system of equations based on the mass matrix, damping matrix, and stiffness matrix and the position, velocity, and acceleration of the current state of the mesh.

Simulating Real-Time Cloth with Adaptive Edge-based Meshes by T. J. R. Simnett, R. G. Laycock, and A. M. Day [3] uses a similar adaptive refinement scheme to simulate cloth in real time; however, it also includes methods to stitch together multiple cloth meshes to create articles of clothing for simulation. We used this paper mostly as a second perspective on adaptive refinement.

Large Steps in Cloth Simulation by David Baraff and Andrew Witkin [4] looks at methods more similar to the mass spring method detailed by Provot [1] but can use irregular grid patterns and uses differentials to remove inaccuracies caused by the Euler method of integration commonly used in Provot-like simulation schemes.

$\sqrt{3} - \mathit{Subdivision}$ by Lief Kobbelt [5] is the original paper describing the refinement technique used, with some modification, by Bender and Deul.

4 Technique

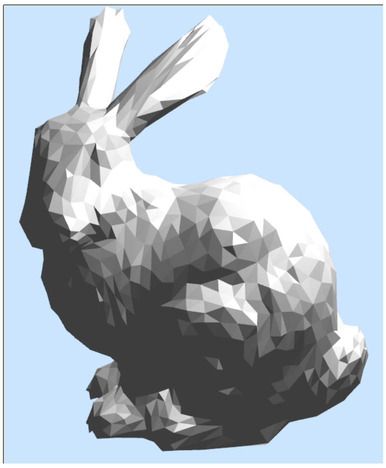

Figure 1: Two subdivision on the 1k bunny

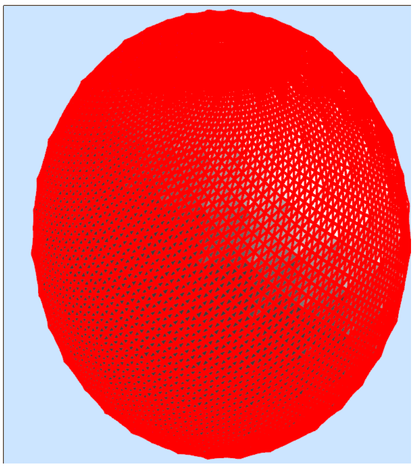

Figure 2: Multiple subdivisions on the sphere

The adjacency data structure used for meshes in the homework assignment is what was used for this implementation. Additional values were added to the Vertex class to hold masses, accelerations, velocities, and rest positions for the objects that were used to prepare to allow physical motion and deformation.

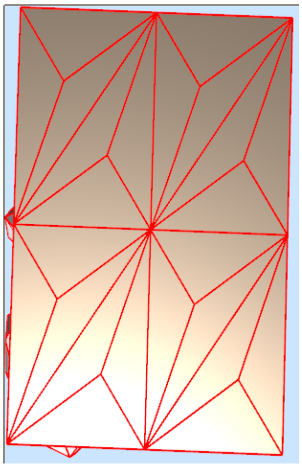

Figure 3: One subdivision on the cloth

The subdivision framework is a slight modification of the original $\sqrt{3}$-Subdivision technique described in [5]. One full round of subdivision is implemented as two successive rounds of slightly varying "sub-subdivisions". Starting from an unrefined mesh, the subdivision first inserts a new vertex into the center of each existing triangle, or equivalently all triangles with generation index equal to 0. All generation 0 triangles are then deleted and replaced with 3 new triangles, equally splitting the original triangles into thirds. At this point, the resulting mesh looks very rough due to the elongation of the generation 1 triangles. This irregularity is fixed with an edge-flipping procedure; specifically, every original, non-boundary "double edge" is flipped so that it connects the other set of opposite vertices of the quad formed by its two adjacent triangles. This operation splits the elongated generation 1 triangles into two equilateral, or close to equilateral, triangles of generation 2.

The mesh is now partially subdivided and has retained a high degree of regularity, the one remaining possible issue is that, due to the edge flipping, the inner group of new triangles appears rotated with regard to the boundary structure of the original mesh. The method solves this slight issue by running the target back through a similar subdivision method.

For this second runthrough, the boundary triangles are split laterally into thirds as opposed to radially. This replaces the edge flip trick for the previously mentioned elongation problems for boundary edges since edge flipping a boundary edge is an invalid operation. Afterwards, all inner triangles are subdivided normally and all "generation 1" non boundary edges are flipped once again.

There are many options for criteria for determining when to refine a specific triangle. The criterion used by Bender and Deul is based on the averaged mean curvature of the mesh over the three vertices of the target triangle.

$$c(x_i) = \frac{1}{2A_{hybrid}} \sum_{j \in R_1(i)} (cot\alpha_{ij} + cot\beta_{ij})(x_i - x_j)$$

Mass calculation for every vertex in the mesh is done using partial areas of the triangles around it, these triangles are said to be in the point's one-ring. For each triangle, a partial area is added to the vertex's effective area, a value that starts at zero. This partial area is computed in one of two ways depending on whether or not the triangle under consideration is obtuse or not. If it is obtuse, then half of its area is taken if the angle containing the given vertex is obtuse, and a quarter of its area is taken if the angle containing the given vertex is not obtuse. If the triangle itself is not obtuse, then the Voronoi area is taken. This Voronoi area is computed with the following formula, where x_i, x_j, and x_k are the three vertices of the triangle, x_i is the vertex of interest, and θ_n is the angle of the triangle at the vertex x_n.

$$A_V = \frac{1}{8}(||x_k - x_i||^2 cot\theta_j + ||x_j - x_i||^2 cot\theta_k)$$

Movement calculations are made by solving the following system of equations, where M is a diagonal matrix containing the masses of the points, a is the matrix containing the accelerations of the vertices, D is the damping matrix, v is the matrix containing the velocities of the vertices, K is the stiffness matrix, x and x_0 are the current and the rest positions of the mesh, respectively, and f_ext is the matrix containing the external forces being applied to each mesh particle.

$$Ma + Dv + K(x - x_0) = f_ext$$

The corotational formulation used to augment the physical simulation is important to avoid undesirable effects when the simulated cloth is undergoing massive rotational deformations. This method is generalized as computing values in an unrotated coordinate system and reapplying the rotation once the values have been found.

Collision detection is done by calculating trajectories for the masses in the mesh and determining which are bound for collision in the frame. Once this is determined, necessary impulses and repulsive forces are applied to correct the model. Friction determines the movement of surfaces in contact with each other as well, under this model.

5 Results

We achieved very good results for the subdivision process and the mass calculation and see a clear path to continuing to implement the rest of the real time cloth simulation. We had initially planned to implement the whole simulation in the time we were given, but we quickly realized that it was not a reasonable goal. We had to readjust the scope of the project to deal primarily with the subdivision scheme.

Despite not being able to realize the ultimate goal of the full cloth simulation, the implementation is robust enough that the current broken physics simulation or a future working physics simulation is applicable to any distinct mesh in the scene. The bunny or cube object could be loaded in and remain rigid as usual but be turned to cloth at the discretion of the user. This could make for some interesting scenes.

6 Conclusions

We spent roughly 150 hours on the project, but at about 50 hours of that were spent on building a custom OpenGL 4 viewer and trying to implement the technique on top of the Homework 2 codebase, which both ended up being scrapped ideas.

Viktor has spent most of his time generating the initial cloth mesh from given dimensions and working on the subdivision scheme, while Wesley spent time reworking the code base to allow for the creation and rendering of the cloth and built up the mass calculation system. Wesley also began on implementing the robust physics and collision detection, but as of the writing of this paper, this has been relegated to future work and is not included in the scope of the project. Both team members helped each other through their work by thinking through problems together and implementing smaller subsections of each other's focus areas. Everything was tested everything by rendering it in the viewer and inspecting it for flaws; additionally, values were printed out to test more hidden things, such as vertex masses.

We would like this method to be further explored in the future and possibly expanded to include support for other physical phenomena acting on dynamic cloths.

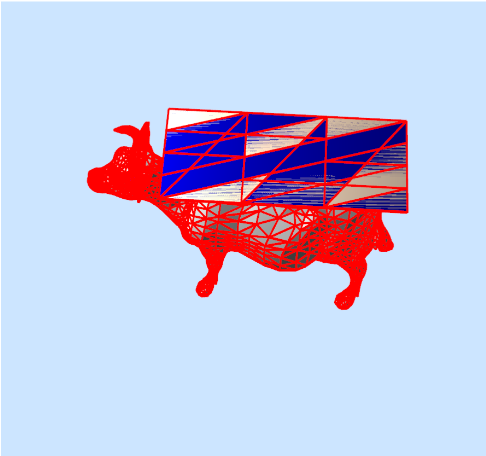

Figure 4: Incorrectly created cloth mesh over a cow

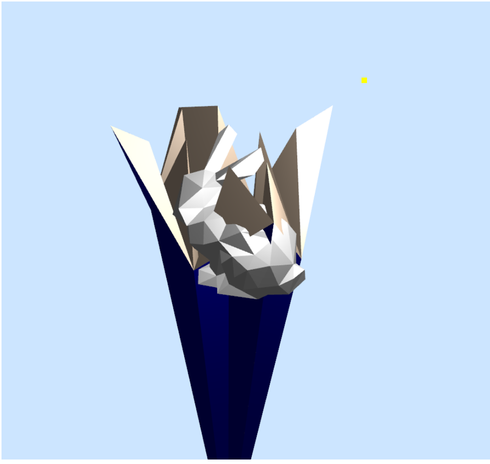

Figure 5: Broken physics simulation severely warps the mesh and allows it to intersect with the bunny

References

[1] Xavier Provot *Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior* 1995.

[2] Jan Bender and Crispin Deul *Adaptive cloth simulation using corotational finite elements* 2013.

[3] T.J.R. Simnett, R.G. Laycock, and A.M. Day *Simulating Real-Time Cloth with Adaptive Edge-based Meshes* 2010.

[4] David Baraff and Andrew Witkin *Large Steps in Cloth Simulation* 1998.

[5] Leif Kobbelt $\sqrt{3}$-*Subdivision* 2000.

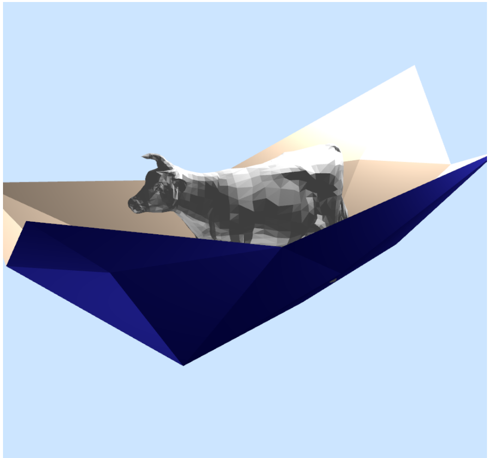

Figure 6: This result of an incorrect physics simulation on the cloth looks remarkably similar to a cow standing in a boat