Realistic Fluid and Cloth Collision Simulation

Christian Grise Matthew Nebel

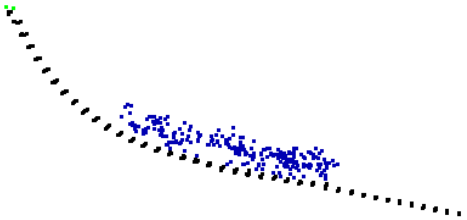

Figure 1: Single drop of fluid onto strip of cloth.

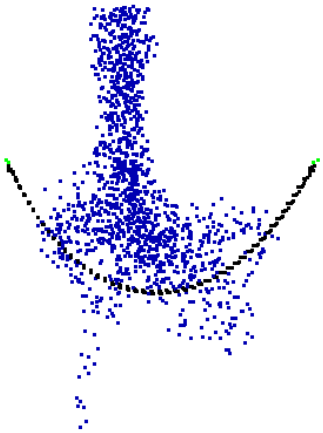

Figure 2: Pouring source onto a strip of cloth

Abstract

This paper seeks to extend previously implemented simulations of cloth [6] and fluid [2] to account for collisions between different sections of cloth as well as between fluid and cloth. The voxel grid representation used in fluid simulation is extended to contain the particles making up the cloth in order to speed up collision detection and provide direct control over both sides of the simulation. The results don't quite achieve realism, but they are fast enough to display a scene within a minute or two.

1. Introduction

Upon starting this project, two individual simulations had been implemented separately. For cloth, the mass-spring architecture as described in [6] uses three different types of springs for each particle. Structural, shear, and flex springs apply force to cloth particles as the cloth is stretched in order to realistically contort the cloth. In addition, particles stretched beyond a given threshold are pulled back for further realism. For fluid, a voxel grid of cells containing velocity information and fluid particles is used in conjunction with the Navier-Stokes equation to provide a realistic fluid simulation as described in [2].

These two simulations, while certainly impressive on their own, have potential to be expanded and combined in order to have cloth and fluid react with each other in a realistic manner. Previous approaches to this problem and the collision of cloth with itself are summarized in section two. Section three discusses our implementation of cloth-cloth and fluid-cloth collisions. Results are discussed in section four. Finally, specific challenges are described in section five and bugs in section six.

2. Previous Work

The first problem to consider is cloth-cloth collision. This has been addressed by Fedkiw and others in [1]. They started with a similar mass-spring system to what we use. To more efficiently calculate collisions, they use an axis-aligned bounding box hierarchy built bottom-up at the beginning of the simulation. They build it by first sweeping one way, pairing off adjacent triangles to get parent nodes, and repeating this in the opposite direction until they have a root node.

Cloth-fluid collision is arguably a more complicated problem and the main focus of this paper. Fedkiw has again tackled this problem with an emphasis on preventing fluid particles from leaking through infinitesimally thin rigid and deformable solids [3]. Essentially, rays are cast from each fluid particle to detect whether they are occluded, visible, or within a buffer region around the polygonal cloth surface. If the particle is occluded or inside of the buffer, it is considered invalid. Its values are replaced with the values of surrounding valid points. Special attention is paid to fluid particles bordering objects so that they do not compress and appear to remove volume. A major advantage of this paper's implementation is that it can be used with just about any solid simulation model. While this ray-casting approach is effective, it is also extremely expensive, and we will seek a faster way to implement collisions.

Another approach to cloth-fluid collision is discussed in [4], where two uniform voxel grids are used: one for fluid and one for cloth. A voxel in the fluid grid stores the particles it contains and a voxel in the cloth grid stores the polygons for which the center of mass is present in the volume of the voxel. Distance between fluid particles and cloth polygons to determine collision is estimated using these center of mass values. The paper goes on to detail equations for how particles exert force on the cloth and vice versa. The second half of the paper details the use of the parallel nature of a GPU to efficiently calculate the forces from each side and the resulting positions of the particles and vertices and execute a simulation in real time. While this GPU implementation is outside the scope of what we want to do, the idea of using a voxel grid for cloth in order to speed up computation is a very appealing idea.

3. Implementation

The first step in our implementation is combining both fluid and cloth into one scene. Our code initially included support for cloth and fluid scenes separately, and we merely extended this to load both at the same time. The only further work required to is to scale the scene files appropriately.

In order to fully display the power of our collision detection, we needed to add support for sources and sinks in the fluid simulation. This is a relatively simple matter of creating fluid particles in a specified location each frame for sources and removing particles in specific locations for sinks. Sources can have an optional velocity value, which gives an initial velocity to cells in which source particles are created.

Inspired by the idea of using a voxel grid for cloth as described in [4], we decided to simply use the fluid voxel grid that had already been created to store cloth particles. After the fluid scene is initialized, the cloth scene is created and its constituent cloth particles are added to the fluid's voxel grid. After cloth movement has been performed each frame, the cells are updated to contain the proper cloth particles.

With each cell in the voxel grid now containing both fluid and cloth particles, checking collisions is straightforward. The entire grid is iterated through, and each fluid particle in each cell finds the closest cloth particles in the surrounding voxels. If this particle is within a certain threshold, push both sides according to the following equations (where p is the cloth particle, and f is the fluid particle):

$p_v = p_v * p_m$
$f_v = f_v * f_m$

$sum = p_v.x + f_v.x,\ p_v.y + f_v.y,\ 0$
$p\_velocity = sum$
$f\_velocity = sum$

```
set the fluid particle's cell's velocities to be the current velocities times
(x-1)/x
```

While this method is somewhat slow, the usage of the fluid's voxel grid to store cloth particles allows for a considerable speedup as each fluid particle will not have to check against every cloth particle every frame.

4. Results

Considering the main feature of our work is fluid-cloth collision, we need several scenes to test the performance of these interactions. We created two scenes, both in 2d, that produce results similar to our original goals.

One scene starts with a drop of fluid at the top of the scene and an angled strip of cloth below it (Figure 1). This cloth has its two highest points fixed in place on each side. The fluid drops faster than the cloth, colliding and staying on the top side of it. The fluid then applies force on the cloth, causing it to fall faster.

Our other scene starts with a piece of cloth stretched horizontally, with both ends fixed in place (Figure 2). Fluid particles spawn above it and fall due to the force of gravity. When they collide with the cloth, most will stay above and push down on the cloth, stretching it further. Some particles will proceed to pass through the cloth once it has been overstretched. This represents of a piece of cloth becoming soaked to the point that fluid is almost freely able to pass through it.

5. Challenges

One of the biggest problems encountered from the beginning was a residual problem from our original fluid simulation implementation. Occasionally fluid particles would become stranded in the scene with no surrounding full cells to give them velocities, This would cause patches of particles to simply float in the air. We adjusted for this by detecting cases where surface cells aren't bordered by any full cells and forcing them to have a velocity due to gravity. This generally causes stranded particles to quickly fall and catch up with the larger body of water, which while not perfect is good enough for our purposes.

Another large problem we encountered was getting cloth to cloth collision reaction correct. The detection was not an issue, but when we tried to correct for the collision we could not get the right equation.

This project took about 30 total hours between us. Christian contributed a great deal to cloth-cloth and fluid-particle collision, writing the code to detect collisions and react appropriately. Matthew worked on refining the fluid simulation and creating and combining scenes, including incorporating cloth particles into fluid voxels.

6. Bugs

A remaining bug exists when scenes are created in full 3d with cells extending into the Z plane. There is no code to account for surface cells in all 64 configurations in the 3d plane due to uncertainty about the necessary calculations and excessive copying and pasting, so simulations have largely been constricted

While sources initially seemed easy to implement, they proved to be more trouble than they were worth. Due to the way cell velocities are handled, most attempts at sources resulted in extremely erratic behavior as randomly generated points tended to favor one side over another and bias velocities into odd trajectories.

7. References

[1] Bridson, R., Fedkiw, R. and Anderson, J. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *SIGGRAPH 2002, ACM TOG 21*, 594-603.

[2] Foster, N., Metaxas, D. 1996. Realistic Animation of Liquids. *Graphical Models and Image Processing, Volume 58, Number 5*, 471-483.

[3] Guendelman, E., Selle, A., Losasso, F., and Fedkiw, R. 2005. Coupling Water and Smoke to Thin Deformable and Rigid Shells. *SIGGRAPH 2005, ACM TOG 24*, 973-981.

[4] Harada, T., Koshizuka, S., and Kawaguchi, Y. 2007. Real-time Fluid Simulation Coupled with Cloth. *Theory and Practice of Computer Graphics*.

[5] Huh, S., Metaxas, D., and Badler, N. 2001. Collision Resolutions in Cloth Simulation. *Computer Animation, IEEE 2001*, 122-127.

[6] Provot, X. 1995. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. *In Proc. Graphics Interface '95*, pp 147-154.