Stephen Ranger
Computational Geometry
Homework 3

Physics, Simulation, and Computer Animation
Adam Bargteil

“Physics, Simulation, and Computer Animation,” a presentation by Adam Bargteil, talked about two specific algorithms as well as an overview of previous work the presenter had done. First was the finite element destruction algorithm followed by a digression, as he put it, in defining a liquid surfaces algorithm. Then, he talked about particle skinning and how he modified an existing algorithm using constrained optimizations and the union of spheres. Finally, he talked about the future of the field and where he saw physics-based animation moving in the future.

The special effects problem; how to define an object that can be easily deformed and manipulated in an intuitive way for animators. He spoke about his finite element destruction techniques and how the elastic and plastic deformations affected the virtual objects they worked with. When an object was elastic, the deformation was temporary, when plastic, the deformation became permanent. Combining these two gave you a wide range of interactions from brittle glass to mold-able clay. Depending on material properties, these deformations can bend, twist, stretch, and even tear, crack, and rip.

Next, his first “digression,” he talked about tracking liquid surfaces using his finite element method. These liquids were modeled to take the elastic deformation properties into defining viscoelastic surfaces. Later, the algorithm was modified to incorporate a finite difference method to define the liquid surfaces. Finally, he quickly talked about particle systems and tetrahedral mesh generation.

Next, the “particle skinning” portion of his presentation and the most involved, talked about defining a way to use particles to loosely define an object (solid or liquid) and then approximating its surface using a constrained optimization of the union of spheres. The main goal of the algorithm was to give static images a smooth and pleasing feel but while under animation, they would preserve temporal coherence and give the animator the ability to process each frame independently for parallel processing. The main piece of the algorithm takes a set of particles that makes up an object, defines a pair of spheres for each, one the size of the particle and one slightly larger, unions each set (smaller, then larger) and finds a surface that fits in between each set. This representation is implicit and good for boolean operations. The initial algorithm first initialized the scalar field, then extracted surface mesh, and finally ran the constrained optimization. The result was dependent on the mesh; as the mesh changed, the surface flickered. The output was inconsistent. Then, the level set method was used for surface definition and the constrained optimization step was done before extracting the mesh. With this change, the animations became much smoother. Finally, using a marching cubes or tetrahedra extracts the surfaces. This algorithm is integrated into OpenVDB.

Finally, the last portion of the presentation discussed the future of algorithms relating to physics-based algorithms. The main focus will be on more and better tools for animators, the ability for more interactive animation techniques, artistic authoring, and predictive animation. The presenter then highlighted some of his current work. The first was a system for defining large liquid systems. Currently, liquid is fairly well done when it is a discrete system and is only considered liquid. However, in real world systems, liquids aerosol and mix with air when agitated. Usually, fluid in == fluid out, however with real systems, fluid in <= fluid out. Adding these modifications gives liquid interactions a more lively feel. Then, he spoke of an unsampling algorithm which takes a coarse input simulation and outputs something more realistic without explicitly subdividing the surface and computing physically-based interaction models. Then, he spoke of work by students on a dynamic sprites algorithm which defines two rest states of an object and then deforms between the two; a modification of the algorithm allowed the objects to reverse their deformation after time. Finally, he went over predictive animation which he described as “not scientific animation” and which might require a new graphics pipeline. This new pipeline might contain a self-feedback loop. These loops could be something along the lines of simulation → geometry → meshing → simulation. Also, this idea could be taken into the real world and used for iterating physical objects in virtual space such as modeling → simulation → fabrication → capture → modeling.