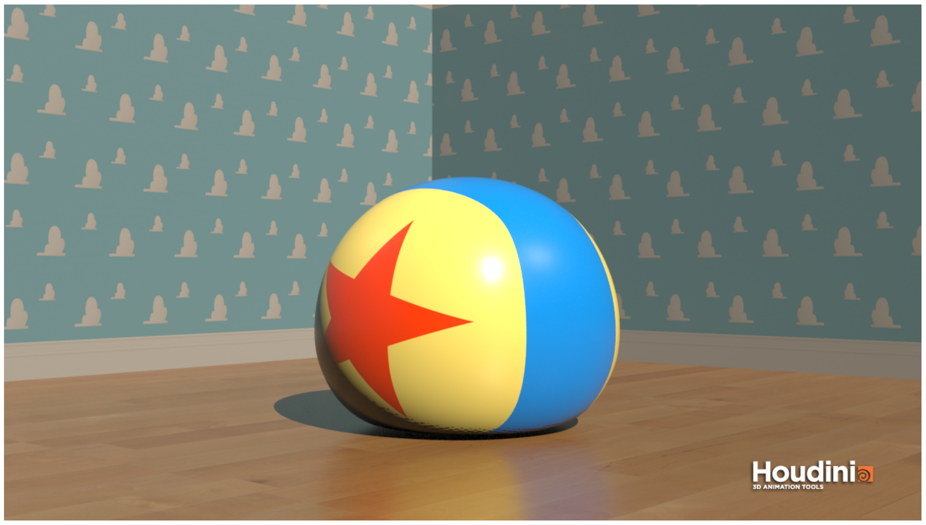
Tasty - 3D Deformable Solid Simulator (FEM)

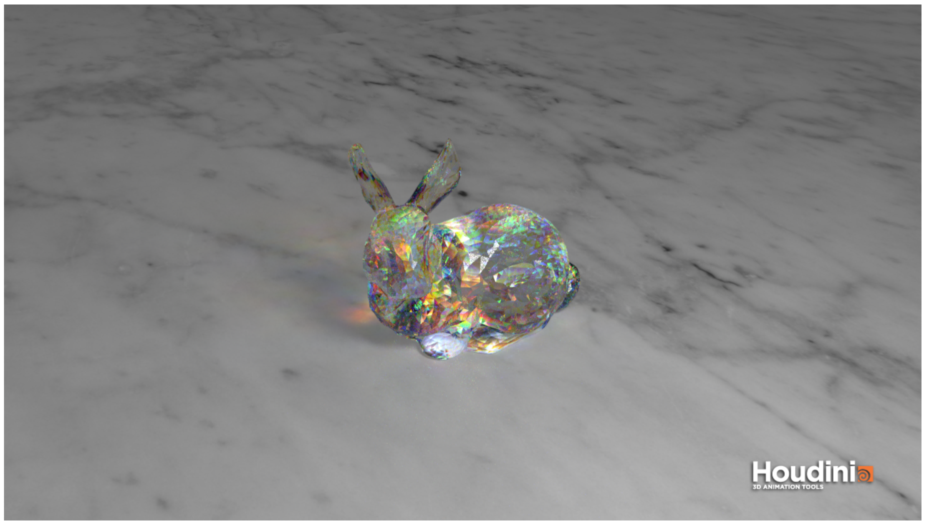
Alexander Chan, Tabatha Hickman, Jacob Snipes, Emily Vo

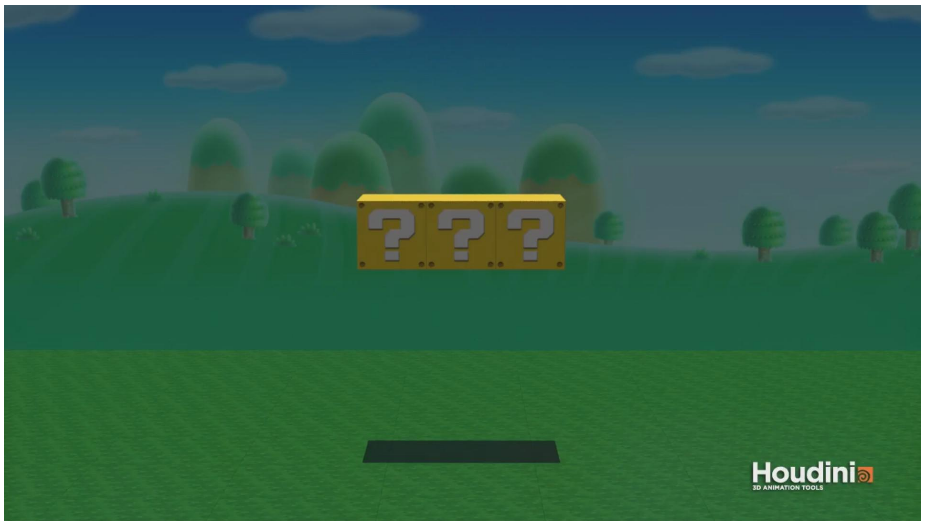
Demos

All demos were rendered in Houdini.

Click the gif to see the corresponding uncompressed video. Click here for the full video playlist.







Implementation Details

`particles.h`

The `Particles` class contains a vector of positions, velocities, accelerations, and forces. The `Particles::tick` function performs numerical integration to calculate updated acceleration, velocity, and position values for each particle. It also handles collisions with the ground and other objects in the scene.

Forces can be set to (0, 0, 0) via the `Particles::resetForces` function. A force can be added to a particular particle, and to all particles

(`Particles::addUniformForce`).

`tetra.h`

The `Tetra` class represents a tetrahedron. It contains 4 particle indices, a mass, and various member functions to compute *Ds*, *P*, and *H*. `Tetra::computeElasticForces` will calculate the elastic forces on each of the four vertices of the tetrahedron and add the forces to their respective particles in the contained `Particle` class.

`main.cpp`

The main procedure of the simulation is as follows:

- Tetrahedralize mesh
- Create a particles structure containing elements in the .node file
  - Initialize the particle's velocity, acceleration, and force to 0
  - Initialize the particle's mass to 0
- Create a tetra structure for each tetrahedron in the .ele file
  - For each tetra, compute *Dm* and *W*
  - Add *(density* W) / 4* to the mass of each particle
- At each timestep:
  - Reset all forces
  - Add gravitational force
  - Calculate internal elastic forces
  - Update kinematics and fix collisions

We use a forward Euler integrator with the Neo-Hookean elastic model for all the demos. Constraints are handled by simply not updating constrained particle's accelerations, velocities, or positions. Collisions are handled with planes and spheres by detecting if a particle's position is inside the object and its velocity is moving against the normal of the object at the point of intersection, then setting the velocity to 0.

Struggles

- Phantom elastic forces
  - Caused by using `==` instead of `=` when resetting forces
  - Caused the mesh to explode or collapse
- Inverting tetrahedrons
  - Caused by using elastic models other than Neo-Hookean
- Improper meshes with self intersections and holes
  - Manually fixed meshes in Maya
- NaN values

    - Timestep too large for forward Euler integration
    - Tiny forces caused by floating point error, used an epsilon to discard small forces
- Low Young's modulus values
    - Caused simulation to look bad

Contributions

- Alexander Chan
    - Setting up project and base code
    - Implementing all the elastic models in the 2012 SIGGRAPH course notes
    - Debugging
    - Scene setup and rendering
- Tabatha Hickman
    - Research
    - Tetgen data I/O
    - Rigid body collisions
    - Rendering
- Jacob Snipes
    - Research
    - Debugging
    - Constraints
    - Interobject collision
- Emily Vo
    - Particle system class
    - Force and energy calculation
    - Scene setup
    - Interobject collision

Resources

- *FEM Simulation of 3D Deformable Solids: A practitioner's guide to theory, discretization and model reduction*, E. Sifakis and J. Barbič, (2012).
- Ladislav Kavan's PBA 2014 Lectures
    - Part 1
    - Part 2

Credits

- Luxo Ball Wallpaper
- ? Block Background
- ? Block Ground

- Thwomp Background
- Thwomp Background - Lakitu
- Mario Symbol
- Marble Texture
- Stanford Bunny
  - Holes were manually filled in Maya