Kyle Engelmann

KWE039

Project A: Lots o’ Dots

**User’s Guide:** In this project, several different types of particle systems were created, each acting under various types of forces and constraints, and run under various types of solvers. The goal was to create four systems: A cloth, a flame, a flock, and small objects orbiting a planet, forming a ring.

The particles in the cloth can be influenced by either spring forces or rigid constraints, with the user being able to switch between them using the F key. The user can also apply a force to the corner of the cloth using the arrow keys. The planet orbiting particles have a planetary gravitational force (Towards the planet), a world gravitational force (Always down), and a spinning force applied to them, with the world gravitational force turned off at startup. The Z key toggles the planetary gravitational force off and on, the X key toggles the spinning force, and the C key toggles world gravity.

The user can select between solvers using the number keys at the top of the keyboard (not sure if num pad keys work as I don't have one), with 1 selecting an explicit Euler solver, 2 selecting an explicit midpoint solver, 3 selecting an implicit Euler solver, and 4 selecting a velocity Verlet solver.

The scene’s camera can also be moved around. Dragging the mouse while clicked will rotate the camera. The WASD keys move the camera forwards, left, back, and right. The Q and E keys move the camera up and down, and holding LEFT SHIFT will speed up camera movements.

**Code Guide:** The Code consists of various types of classes, many of which take care of many basic functions as long as their update functions are called each frame. The classes used are the Particle class, the particleState class, the particleSystem class, the Force class, and the Constraint class.

The Particle, particleState, and particleSystem classes assist in the creation and use of particle systems. The Particle and particleState instances needed are created by the particleSystem class on construction, and both of these classes are used to modify the particles. The particleSystem class contains an array of Particle instances called particles. The system also contains a few particleState instances, called s1, s2, and dt. S1 contains the current state of all the particles, s2 contains the next state, and dt contains information about how to calculate s2 from s1. s2 and dt are only needed in the solvers, and changes to them will not affect results at all. All changes to particles should be made to the S1 state. The particles array can be used to access s1 through functions like getPosition() and setVelocity(vel). S1 can also be accessed directly through functions like getParticlePosition(i) and setParticleVelocity(i, vel). The particleSystem class also contains an init function for initializing rendering objects, an update function for solving for particle states, and a render function for drawing particles on screen. There are 2 classes derived from the particle system class, a springSystem class and a Cloth class, which make small changes to functions and initializers, but otherwise work the same.

The force class only contains 2 methods, a constructor and an apply function, and cannot be instanced directly. Instances of derived classes must be made, to create a force. Derived force classes can have their own apply function, but the base force class has its own apply function which will call a perParticle function for every particle, so for forces that affect each particle the same, a perParticle function should be written. Constraints work the same way, with a base class called Constraints and an apply function that calls a perParticle function. Each particleSystem instance has a list of Force instances, called forces, and a list of Constraint instances, called constraints. To apply a force or a constraint to the system, push the object onto the appropriate list. The apply functions will be called in the system’s update function. Spring and cloth systems do not need the springs pushed into the list, and cloths do not need rod constraints pushed either.

**Results:** Overall, the project was a success. Multiple particle systems were created, and were able to act independently of each other, each with a different set of forces and constraints. Figure 1 shows the scene upon opening.

The flame system was created by using a single texture applied to each particle. The rotation of the texture was randomly selected per particle. The particles had a single force acting up, and their masses, sizes, and colors would change over their lifetimes. The system also had a single constraint to reset a particle’s positions after a certain amount of

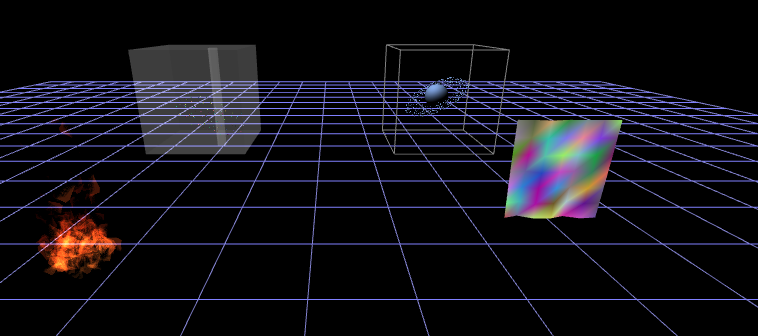


Figure : The project upon opening

time. Also, the particles were rendered using an additive blend function, making the flame get brighter colors in places where many particles overlap. The resulting flame is shown in Figure 2.

The cloth was created with a grid of interconnected particles. The connections could be switched between spring forcers and rigid constraints, creating different properties in the cloth. Figures 3 and 4 show these differences. Figure 3 shows the behavior of the cloth with spring forcers. The cloth acted much more elastic in this case, looking a lot like a rubbery material. Also, the cloth requires a large amount of damping to remain stable. Figure 4 shows the behavior of the cloth with rigid constraints. The cloth exhibits much less elasticity, resulting in a much more realistic cloth simulation, and it requires no damping to remain stable, but some damping was still applied to create more realistic movements.

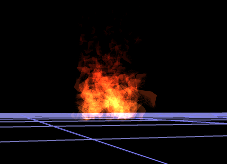


Figure : Flame created from a particle system with textured particles and an additive blend function

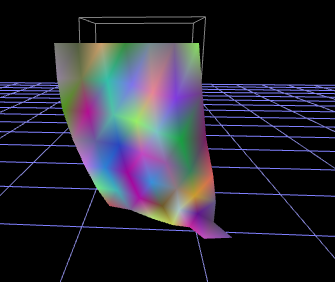
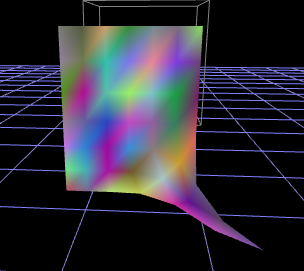


Figure :Elastic cloth behavior created with springs

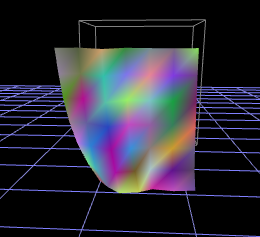
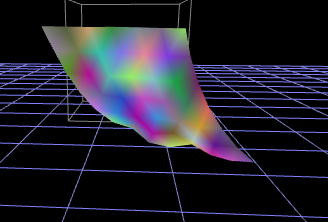


Figure : Inelastic cloth created from rigid constraints

The flock, shown in Figure 5, consisted of many freely moving particles influenced by Boids forces. The particles form into a single cluster and avoid obstacles. The obstacles for them to avoid were 6 walls trapping them in a cube, along with one column, which the flock would split into two to avoid. The particles were also under the influence of a circular force to keep them moving indefinitely.

The ringed planet is shown in Figure 6. The particles are under the influence of a gravitational pull from the planet, a circular force to keep them orbiting, and a world

gravitational force, which pulls them down. All of the forces can toggled on and off, and the world gravitational force starts turned off, as it disrupts the particle’s orbit and just sends them crashing to the ground. Turning off the gravitational force results in the particles moving further and further from the planet, while turning off the spinning force results in the particles crashing into the planet. The particles have constraints to prevent them from leaving a boxed area around the planet, and another constraint to prevent them from entering the planet.

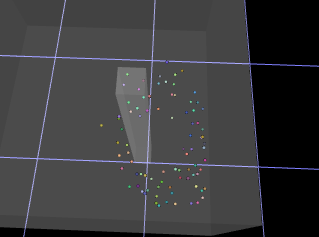


Figure : Flocking particles splitting around obstacle

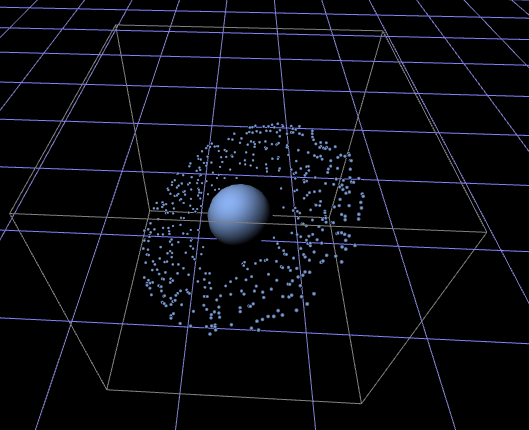


Figure : Particles orbiting a planet