Amar Saini - Computer Animation/Simulation Project 1 Report

Numerical Solvers Implemented

Euler Solver: This solver was implemented exactly how it was done in the example particle system. It stores the next position and velocity in each state. 6 variables are calculated during every integration step. The position in the x, y, and z direction, as well as the velocity in the x, y, and z direction. These are calculated according to the equations in the lecture slides.

Verlet Solver: This solver is similar to the Euler method. It holds 6 variables, but rather than position and velocity, it holds the new current position, and the previous position (in the x, y, and z direction). I use the new current position and the previous position to estimate what velocity would be for a particle (take the difference in position and divide by change in time). I needed to estimate velocity because my particle system relies on a spring-force collision system.

Spring-Force Collison System 1: To hold the net together, I simply applied a spring force to every pair of particles horizontally, vertically, and diagonally (top-left to bottom-right, and top-right to bottom-left). The spring constant is the same for all springs for all particles in the net (to give it its deformable, stretchable look).

Spring-Force Collison System 2: To have bigger particles, or balls, collide with the spring, I implemented another spring-force collision system. This system basically checks if any of the big particle balls are in contact with any of the net particles. We can check if they are colliding based off of the distance between the particle ball and net particle (see if the distance is less than the sum of the particle ball's radius and the net particle's radius). The spring constant for the force pushing against the net can be adjusted through the "impact" slider on the UI. This will give the net the ability to deform more upon impact.

Geometric Constraint + Verlet Integration: Since we store the current and previous positions in the state when using the Verlet integration method, we can use this to put a maximum cap on the amount of displacement in a single frame. For example, we can check the displacement of a given integration step (current position – previous position), and if it's bigger than some threshold, we can reduce the amount displacement by a certain "stiffness" factor. This can be adjusted via the "stiffness" slider on the UI.

Benchmarks

Note: N represents the number of particles used to create the particle system for the deformable net. And M represents the number of big particle balls that will collide with the N-sized deformable net. "Verlet Integration" refers to using the Verlet Integration method with the geometric constraint described above.

	Euler Integration	Verlet Integration
N = 100, M = 2	-2.996	-1.374
N = 100, M = 5	-4.320	-1.779
N = 400, M = 5	-5.908	-4.241
N = 400, M = 15	-8.172	-5.316
N = 900, M = 15	-16.621	-10.021
N = 900, M = 30	-21.550	-13.7465

Benchmark #1 tests the stiffness of each integration method by observing how far down the net will stretch. To measure this metric, we simply look at the minimum point that any particle on the net ever reaches (on the y-axis).

	Euler Integration	Verlet Integration
N = 400, M = 10	1 ball lost	0 balls lost
N = 400, M = 15	8 balls lost	5 balls lost
N = 400, M = 20	12 balls lost	11 balls lost
N = 900, M = 20	18 balls lost	10 balls lost
N = 900, M = 30	28 balls lost	23 balls lost
N = 900, M = 50	50 balls lost	43 balls lost

Benchmark #2 further tests the stiffness of each integration method. This is measured by capturing the number of balls that slip through the net. Note that in this benchmark, we have a stronger gravity (a vertical wind of -5), and a stronger impact (3x stronger) to simulate strong spring forces against the net.

Conclusion

As expected, I found that the Euler integration method yields a more "stretchy" deformable net as compared to the Verlet integration method. Both integration methods have roughly the same runtime (since they each require 1 pass through all the particles & spring forces). Benchmark #1 shows that the Euler integration method causes a bigger stretch in the net for different size nets and amount of big particle balls. For all combinations of net sizes and number of colliding balls, the Verlet integration method was always stiffer (since the Verlet net always stretched less distance than the Euler net). Benchmark #2 also shows that the Euler integration method causes more rips/slips through the net when under heavy force due to multiple balls. Since using the Euler integration method creates a non-stiff net, the net can stretch and deform more. Because of this, the big particle balls can fall/slip through the gaps created from big stretches (particles are loosely held together). The Verlet integration method doesn't stretch as much, so it yields a lower amount of balls lost from the net stretching.

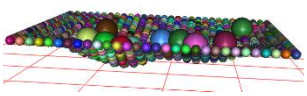

Image 1: N = 900, M = 10 Euler Integration

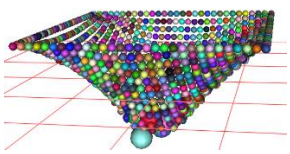

Image 2: N = 900, M = 10. Strong impact and increased gravity (vertical wind) to demonstrate the deformable net tearing, causing balls to slip out of the net.