Cloth Simulation

ABSTRACT

This paper covers a simulation of cloth-like materials implemented as a mass-spring system.

1. INTRODUCTION

This simulation aimed to simulate a variety of objects that could be classified as soft-body planes, such as cloth, paper, and plastic packaging. Additional goals were real-time rendering, creasing, tearing, and permanent stretching of the objects.

2. IMPLEMENTATION

The cloth is modeled as a mass-spring system, and is simulated via Ordinary Differential Equations (ODEs) which are integrated using a Runge-Kutta-Fehlberg 4(5) integrator.

2.1 Model

The model (depicted in Figure 1) is constructed as an n x m lattice of masses, with each mass connected to its adjacent masses via a spring, and each spring connected to its opposite and adjacent springs by a radial spring. The visualization of the model uses the center of the 4 surrounding masses as each vertex.

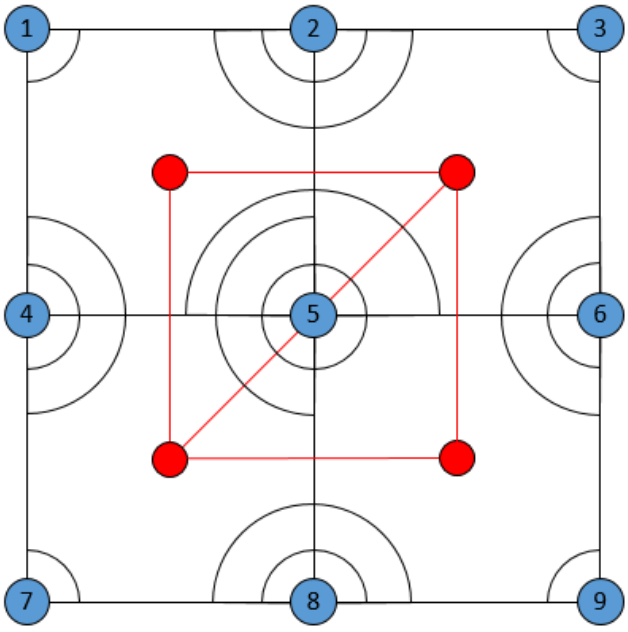

Figure 1. Diagram of model

In the diagram depicted in Figure 1, the blue dots represent the masses, the red dots represent the vertices, the black straight lines represent the springs connecting the masses, the smaller black rounded lines represent the radial springs connecting adjacent springs, and the larger black rounded lines represent the radial springs connecting opposite springs. The red lines depict the triangles constructed by the vertices, which is what gets displayed while running the simulation. The radial springs connecting opposite springs have been enlarged for viewing purposes. (See section 2.2 for spring size details)

2.2 ODEs

The ODEs that govern the model are based on Hookes Law. The force of each spring connecting the masses is: $F = -kx - cv$. Figure 2 shows a diagram for how the forces of the radials springs are obtained.

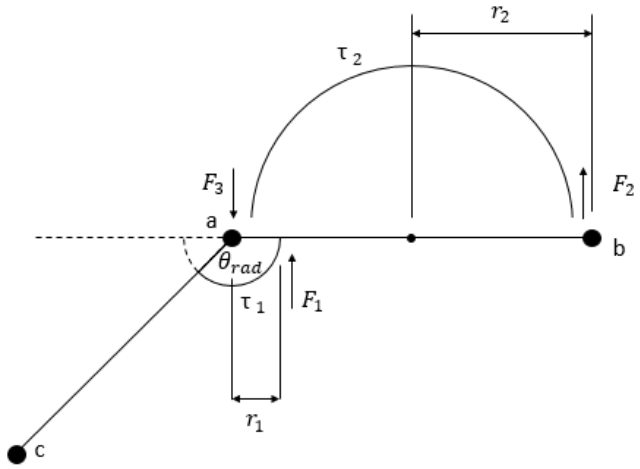

Figure 2

From Figure 2 the following can be derived:

$\tau_1 = \|r_1\|\|F_1\| \sin\theta$

$\tau_2 = \|r_2\|\|F_2\| \sin\theta = \|r_2\|\|F_3\| \sin\theta$

$\tau_1 = \tau_2 \Rightarrow$

$\|r_1\|\|F_1\| \sin\theta = \|r_2\|\|F_2\| \sin\theta = -\|r_2\|\|F_3\| \sin\theta$ eq. 3

F_1 is defined as the force of the radial spring, and since the radial spring is centered on the mass, F_1 is always perpendicular to the connector spring, as is F_2 and F_3. Also for simplicity, r_1 is set to always be 1.

F_1 is obtained by adapting Hookes Law to use the arc length and angular velocity of the radial spring. The angular velocity is defined as $\omega = \frac{d\theta_{rad}}{dt}$.

$$F_1 = -k(\theta_{default} - \theta_{rad})\left(\frac{\pi}{180}\right) - c\omega\left(\frac{\pi}{180}\right)$$

Where $\theta_{default}$ is the resting angle of the radial spring. In all of the simulation runs $\theta_{default}$ is 90° for adjacent springs and 180° for opposite springs.

F_2 and F_3 are derived by substituting F_1 into eq. 3 with $r_1 = 1$ and $\sin\theta = 1$:

$$\left\| -k\left(\theta_{default} - \theta_{rad}\right)\left(\frac{\pi}{180}\right) - c\omega\left(\frac{\pi}{180}\right)\right\| = \|r_2\|\|F_2\| = -\|r_2\|\|F_3\|$$

$$\Rightarrow$$

$$\|F_2\| = \frac{\left\| -k\left(\theta_{default} - \theta_{rad}\right)\left(\frac{\pi}{180}\right) - c\omega\left(\frac{\pi}{180}\right)\right\|}{\|r_2\|}$$

$$\|F_3\| = -\|F_2\|$$

Where F_2 is the force of the radial spring acting on mass 'b', and F_3 is the force of the radial spring acting on mass 'a'. Dividing these forces by the mass of the points, the equations for acceleration that are used in the integrator are obtained.

2.2.1 Addition Note

Since the model is in 3 dimensional space, θ_{rad} is the smallest angle between two springs, and is calculated using the dot product, and therefore its possible value ranges from 0 to 180.

2.3 Runge-Kutta-Fehlberg 4(5) Integrator

For integrating the ODEs, the Runge-Kutta-Fehlberg 4(5) method was used. The Butcher tableau for this method is shown in Figure 3 below.

0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	−7200/2197	7296/2197			
1	439/216	−8	3680/513	−845/4104		
1/2	−8/27	2	−3544/2565	1859/4104	−11/40	
	16/135	0	6656/12825	28561/56430	−9/50	2/55
	25/216	0	1408/2565	2197/4104	−1/5	0

Figure 3

2.4 Forces Applied on Each Mass

For each mass, three sets of forces are calculated. The first set is the force of all springs connecting the mass to its adjacent mass. The second set is the force of the adjacent masses' radial springs connected to the spring connecting it to the mass. The third set is the force of the mass's own radial springs acting on the mass.

Using Figure 1 as an example, the forces calculated for point 4 are the forces of the springs connecting it to points 1, 5, and 7, the radial spring of point 1, the outer-most radial spring of point 5, the top left and bottom left inner-most radial springs of point 5, the radial spring of point 7, and the three radial springs of point 4.

2.5 System Variables

The state of the system is stored in an array with the point specific variables stored consecutively, followed by all the connector spring variables. The following subsections list the point specific variables, of which there are 31 in total.

2.5.1 Input Variables

Mass, k values of each radial spring, c values of each radial spring, $\theta_{default}$ of each radial spring.

2.5.2 State Variables

(x, y, z) position, (x, y, z) velocity, θ_{rad} of each radial spring.

2.5.3 Output Variables

(x, y, z) position

3. SAMPLE RUNS

In all sample runs the center point is anchored at position (0, 0, 0).

In the implementation the model is built with the arguments:

Number of vertical points.

Number of horizontal points.

Mass in kg. (Every point has the same mass)

k of radial springs connecting adjacent springs.

c of radial springs connecting adjacent springs.

k of radial springs connecting opposite springs.

c of radial springs connecting opposite springs.

k of connector springs.

c of connector springs.

length of connector springs in meters.

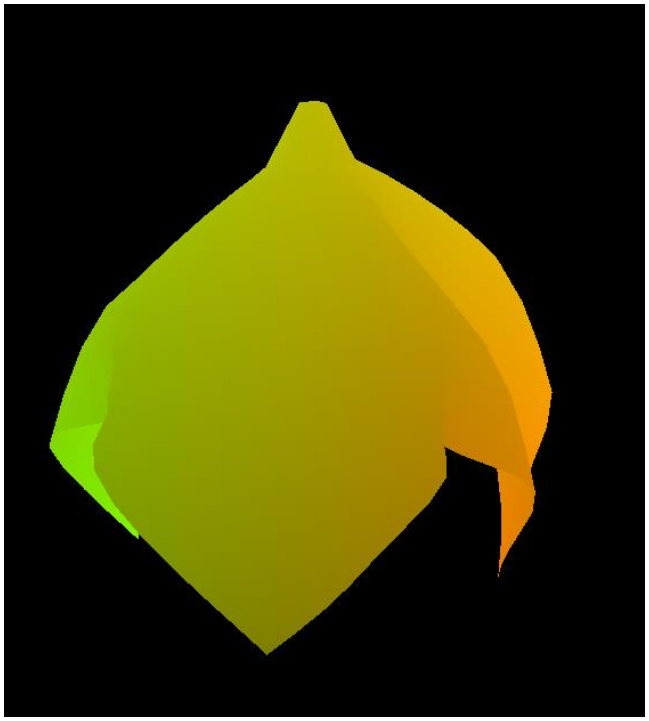

Figure 4

Figure 4 depicts a simulation with values [21, 21, 1, 4, 1, 1400, 200, 80, 5, 1]. The image is of the object at rest. As can be seen, the object is folding in on itself as one would expect.

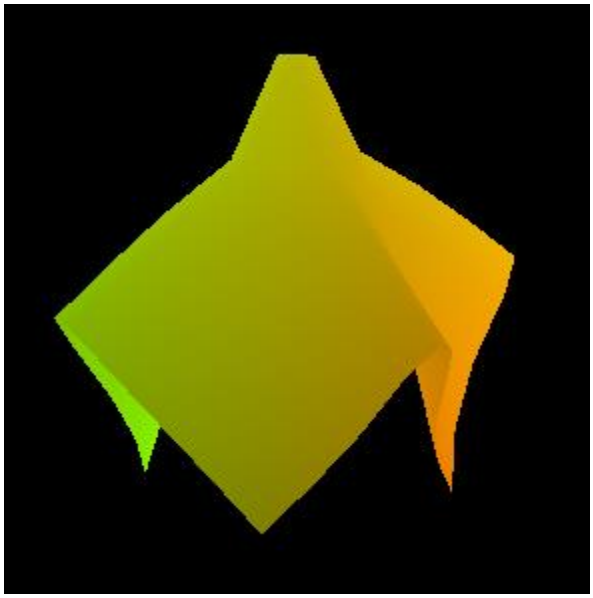

Figure 5. Values: [11, 11, 1, 4, 1, 1400, 200, 80, 5, 1]

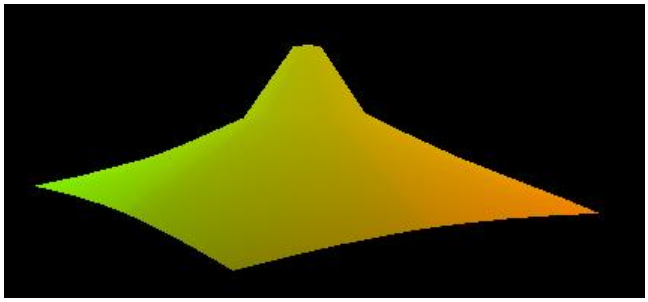

Figure 6. Values: [11, 11, 1, 400, 10, 1400, 200, 80, 5, 1]

4. REMARKS

4.1 Accuracy

The simulation does not do collision detection, which means that the simulation is only accurate when the object is stiff enough to not intersect itself.

4.2 Performance

With one of the goals of this simulation being real-time rendering, the simulation was implemented using the Vulkan Graphics API. Vulkan was chosen based on it's cross-vendor hardware support. With the use of GPU computing, cloths up to 25x25 points were able to render in real time. The compute kernel ran one time step for each point. Due to OS time restrictions, if the kernel did not complete in 5 seconds, the OS would terminate the kernel. This led to needing to sacrifice some accuracy in the integrator, and decrease the size of the time steps. Due to earlier plans of doing collision detection on the CPU, after every time step all the data is copied to the GPU to be computed, then copied back to the system memory, which causes slowdowns.

4.3 Visualization

The model that gets rendered to the screen doesn't have any shading, which can make it difficult to see some of the contours.

Since the vertices are in the centers of the masses (see Figure 1), the point that the objects are anchored to in the sample runs are not visible.

5. REFERENCES

[1] Figure 3 obtained from https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta%E2%80%93Fehlberg_method