CPSC 687 A3 Description

Xi Wang, 30057535

Mar.26 2020

Program description: This program animates 5 mass-spring systems, including 4 basic parts and bonus-2.

Compilation [1]

1. How to Install Dependencies (Ubuntu)
   sudo apt install cmake build-essential
2. How to Build
   cmake -H. -Bbuild -DCMAKE_BUILD_TYPE=Release
   cmake –build build
3. How to Run
   (a) $./build/simple to run the program.
   (b) Press **TAB** to switch the model.
   (c) Press **R** to replay the animation of the current model.
   (d) Use **left mouse button** to rotate the model.
   (e) Use **SHIFT + left mouse button** to zoom.

Math/ physics [2]

1. **Net force**
   Given a spring the connects two particles $\boldsymbol{p}_i$ and bmp_j. Their positions and velocities are $\boldsymbol{x}_i$, $\boldsymbol{x}_j$, $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$, respectively. The net force is composed of spring forces, damping forces, and gravity.

(a) Spring force. The spring force is proportional to the displacement of the particle positions. The spring force acting on $\boldsymbol{p}_i$ is

$$\boldsymbol{F}_{ij}^s = -k_s(\|\boldsymbol{x}_i - \boldsymbol{x}_j\| - l)\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|},$$

where k_s is the spring stiffness coefficient and l is the rest length of the spring.

(b) Damping force. The damping force is proportional to the relative velocity of $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$, projected to the direction of $\boldsymbol{x}_i - \boldsymbol{x}_j$. The damping force acting on $\boldsymbol{p}_i$ is

$$\boldsymbol{F}_i j^d = -k_d(\boldsymbol{v}_i - \boldsymbol{v}_j) \cdot \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|},$$

where k_d is the damping coefficient.

(c) Gravity $\boldsymbol{F}_i^g = m_i \boldsymbol{g}$.

2. **Semi-implicit Euler integration**
The velocity and position of the particle $\boldsymbol{p}_i$ at the $n+1$ time step are updated by the semi-implicit Euler integration:

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \frac{\boldsymbol{F}_i^n}{m_i}\Delta t,$$

$$\boldsymbol{x}_i^{n+1} = \boldsymbol{x}_i^n + \boldsymbol{v}_i^{n+1}\Delta t,$$

where F_i is the net force and m_i is the mass of p_i.

3. **Collision**
Collisions are handled using the penalty method. If a particle $\boldsymbol{p}_i$ collide with the ground (i.e., $y_i < 0$), then the vertical speed is updated as

$$v^+ = -c_r v^-,$$

where $0 \leq c_r \leq 1$ is the *coefficient of restitution* and $v^- < 0$ is the original vertical speed calculated in the numerical integration.

4. **Bonus**
The bonus part animates a sheet of cloth falling on a table.

(a) Collision detection. Assume the radius of the tabletop is R, and the center of the tabletop is (X, Y, Z). Given a particle position (x, y, z) and its velocity v_x, v_y, v_z, collision happens if

$$\begin{cases} (x - X)^2 + (z - Z)^2 < R^2, \\ Y - h < y < Y, \end{cases}$$

, where h is the thickness of the tabletop.

(b) Collision handling. For the sake of simplicity, the cloth and the table are vertically aligned at the beginning. If a particle collides with the tabletop, then we let the particle "stick" to the table, which means the velocity of the particle is set to 0. This is equivalent to applying an extra upward force (or in the horizontal direction if the particle collides with the side of the table) to the particle, because they are both reducing the speed. Setting the velocity to 0 also avoids perturbations of the cloth, as observed. This is also close to the real world scenario where friction avoids the cloth to move (or stretch) horizontally.

Acknowledgement

The technical document of this assignment provided useful pseudo-code and math reviews. The material was well-explained by our TA Andrew Owens. The table model is retrieved from https://free3d.com/3d-model/round-glass-table-set-317777.html.

References

[1] Givr api, https://lakin.ca/givr/.

[2] Adam Runions Przemyslaw Prusinkiewicz Andrew Owens, Jeremy Hart. *CPSC 587/687 Assignment 2 Notes*. University of Calgary, 2020.