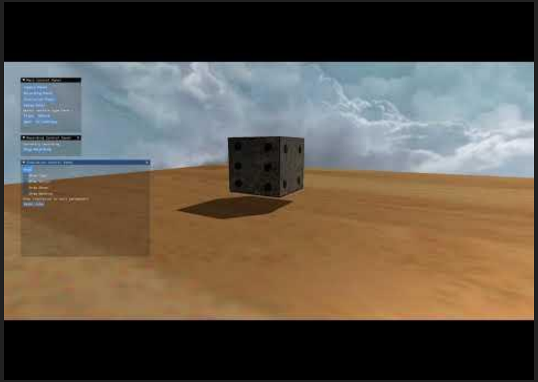
Particle System

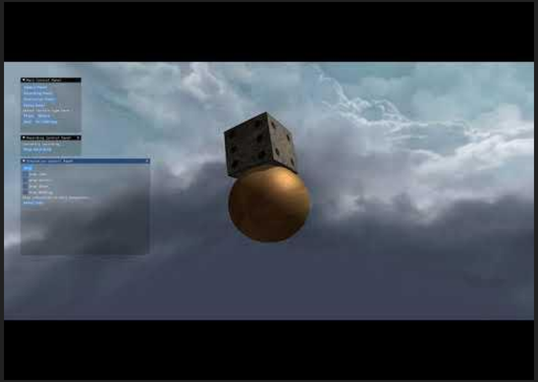
Soft Body Simulation

Outline

- Demo
- Project overview
- Scoring criteria
- Objective and explanation
- Submission detail
- Hint and reminder

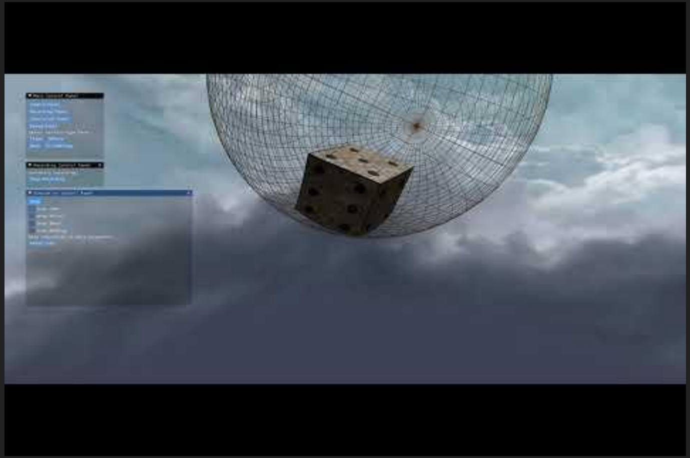
Demo



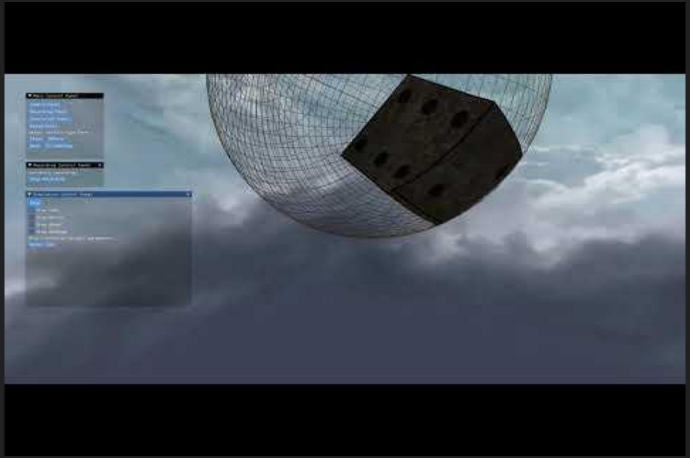


Plane	Tiled Plane
Sphere	Bowl
	frame rate 60

Demo (cont.)



Friction Coefficient = 0.3



Friction Coefficient = 0.0

Project overview

- Solution layout
    - bin
    - assets (textures and shaders)
    - [Screenshots] (will be created if record is turned on)
    - src (source code)
    - utility (tools for outputing video)
    - vendor (project dependencies)

Project overview (cont.)

- Environment
  - IDE: Visual studio 2017
  - Platform: Windows

  - Graphics API: OpenGL
  - OpenGL Loading Library: glad
  - OpenGL Toolkit: glfw
  - UI Library: dear imgui

  - Math Library: Eigen
    - Eigen::Vector3f

Project overview (cont.)

- src
  - gfx (a simple graphics library for rendering basic geometries)
  - simulation (code for running particle system simulation)
  - util (utilities including outputing screenshots as .jpg files)

  - everything you need to implement is in the simulation folder
    - but you can edit other components if you want

Scoring Criteria

- Construct the connection of springs - 10%
- Compute spring and damper forces - 20%
- Handle Collision - 25%
    - Plane and TiltedPlane - 5%
    - Sphere - 5%
    - Bowl - 5%

    - Contact force (resist and friction) - 10%

Scoring Criteria (cont.)

- Intergrator - 25%
    - Explicit Euler - 5%
    - Implicit and Midpoint Euler - 5%
    - Runge-Kutta 4th - 15%

- Report - 20%
- Bonus - up to 15%
    - Improve graphics?
    - Other type of terrain?
    - Other type of collision?

Objective and explanation

- Construct the connection of springs
  - void Cube::initializeSping()
- Compute spring and damper forces
  - void Cube::computeInternalForce()
    - Trace every spring and apply the force accordingly.
  - Eigen::Vector3f Cube::computeSpringForce(...)
  - Eigen::Vector3f Cube::computeDamperForce(...)

Objective and explanation (cont.)

- Handle Collision
  - void <differentTerrainClass>::handleCollision(...)
  - There are four terrains: Plane, Sphere, Bowl and Tilted Plane
    - constexpr float eEPSILON (ε) = 0.01f;
    - constexpr float coefResist = 0.8f;
    - constexpr float coefFriction = 0.3f;

  - You can assume the terrain will not move under any circumstances.
  - Related parameters can be found in class member.

Objective and explanation (cont.)

- Integrator
  - void <differentIntegrator>::integrate(...)
  - There are four integrators
    - ExplicitEuler
    - ImplicitEuler
    - MidpointEuler
    - RungeKuttaFourth

Objective and explanation (cont.)

- Report (below is a suggested outline)
    - Introduction/Motivation
    - Fundamentals
    - Implementation
    - Result and Discussion
        - The difference between integrators
        - Effect of parameters
    - Conclusion

Submission detail

- Compress all the files into a .zip file
  - Naming rule: CA1_StudentID.zip
    - e.g., CA1_309553010.zip
  - If the file size exceeds the limitation on new E3, upload only the "src" folder and main.cpp for the source code component in zip mentioned below.

- Your zip file should contain following components
  - Source code (ensure your project build successfully)
  - At least 2 videos (include parameters in your video)
  - Report in pdf format, no more than 10 pages

Submission detail (cont.)

- Upload all your materials to new E3
    - No limit to the number of times of upload
    - The latest version is your final submission

Submission detail (cont.)

- Late policies
  - Penalty of 10 points on each day after deadline
- Cheating policies
  - 0 points for any cheating on assignments
  - Allowing another student to examine your code is also considered as cheating
- Deadline
  - Monday, 2021/04/05, 23:55

Hint and Reminder

- Hint: course materials
  - Spring and damper forces
    - Review “particle.pptx” from p.9 - p.13
  - Explicit Euler integration
    - Review “ODE_basics.pptx” from p.15 - p.16
  - Midpoint Euler integration
    - Review “ODE_basics.pptx” from p.18 - p.20
    - “Physically Based Modeling” from B.5 - B.6

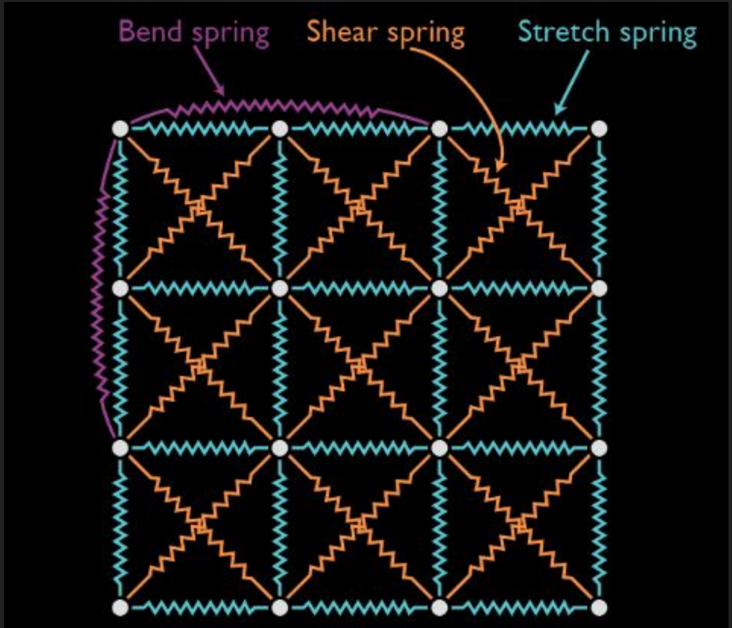
Hint and Reminder (cont.)

- Hint: course materials
  - Implicit Euler integration
    - Review "ODE_implicit.pptx" from p.18 - p.19
  - Runge-Kutta 4th (RK4) integration
    - Review “ODE_basics.pptx” from p.21
    - “Physically Based Modeling” from B.5 - B.6

Hint and Reminder (cont.)

- Hint: spring initialization
  - Initialize three types of spring
    - struct, shear and bending
  - “springStartID” and “springEndID” are the index in the “std::vector<Particle> particles”

```
Spring(
    int springStartID,
    int springEndID,
    float restLength,
    float springCoef,
    float damperCoef,
    SpringType type
);
```



Courtesy of Chris Twigg

Hint and Reminder (cont.)

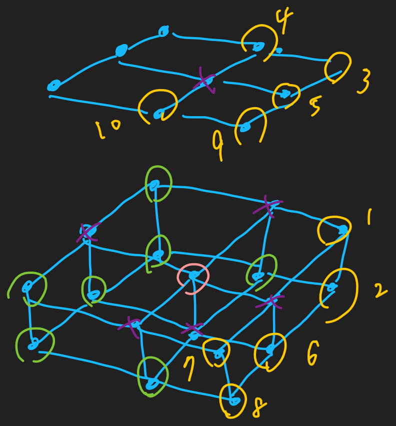
- Hint: spring initialization
  - In practice, it's better to reserve memory space for vector if the size is known, but we'll make it simple for now.
  - Each type of spring will have different number of connection directions
  - Assume a 3x3x3 cube with 27 particles and observe the center particle
    - Struct / bending: 3 directions
      - 6 directions: up, down, left, right, front and back. But if each particle is responsible for all 6 directions, there will have duplicate connection. Thus, each particle will only be responsible for 3 directions.
    - Shear: 10 directions
      - Center particle is surrounded by 26 particles. 26 - 6 (up, down, left, right, front and back) = 20, and each particle can be only responsible for half part of directions.

Hint and Reminder (cont.)

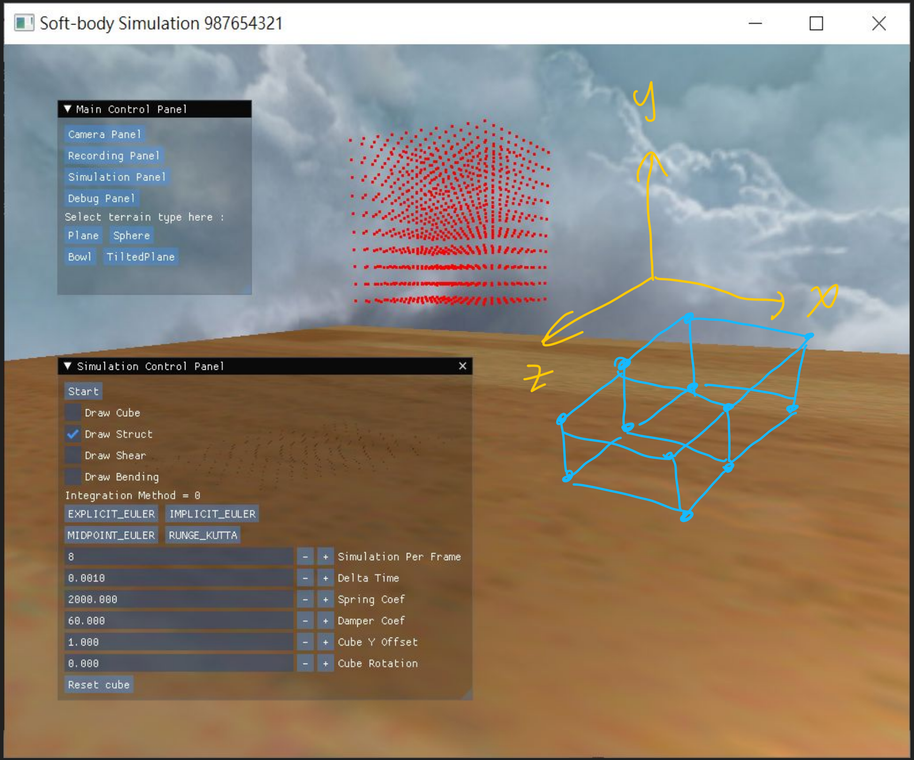
- Hint: spring initialization
  - Sample code of connecting struct spring on one axis (z-direction)
  - ```
    for (int i = 0; i < particleNumPerEdge; i++)
        for (int j = 0; j < particleNumPerEdge; j++)
            for (int k = 0; k < particleNumPerEdge - 1; k++)

        iParticleID   = i * particleNumPerFace + j * particleNumPerEdge + k;
        iNeighborID = i * particleNumPerFace + j * particleNumPerEdge + k + 1;

    springs.push_back(Spring(...));
    ```


    ```

Hint and Reminder (cont.)

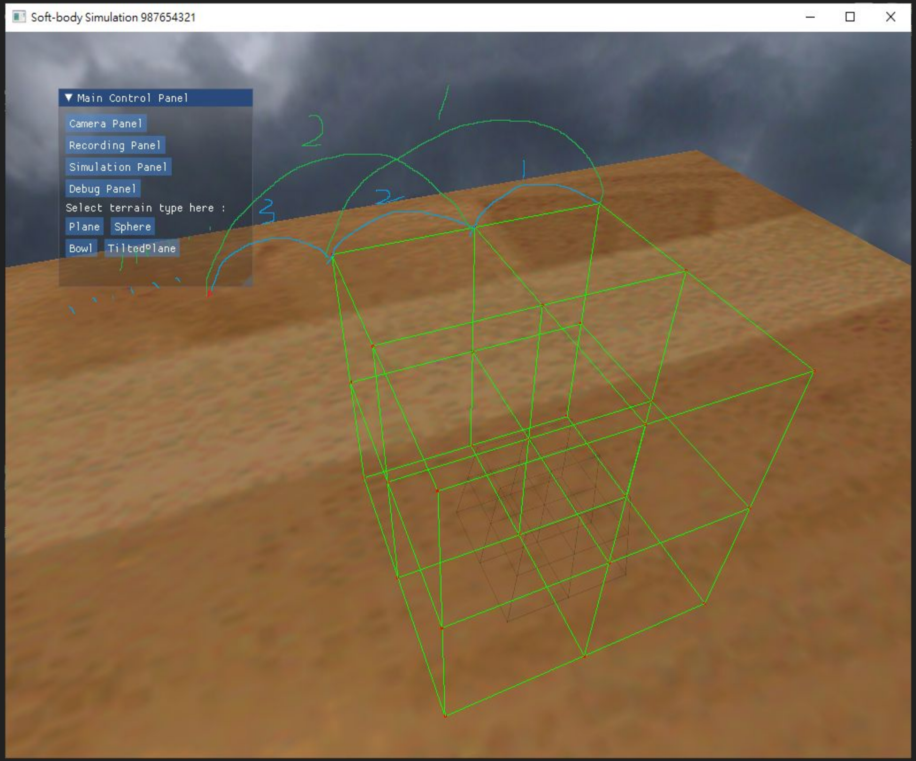


Hint and Reminder (cont.)

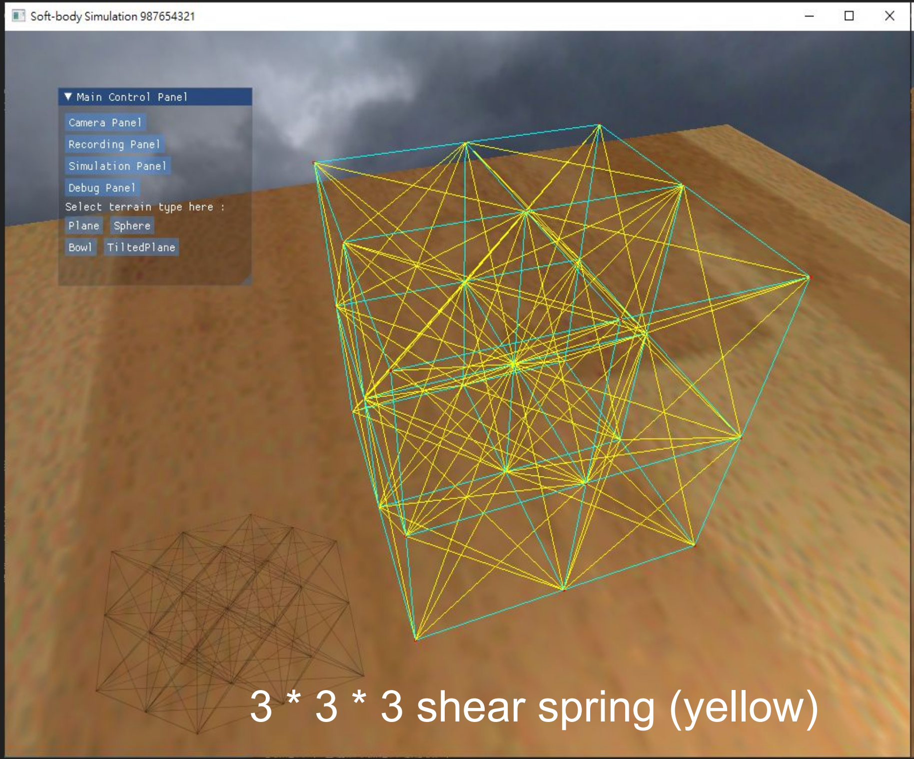
This image shows difference between bend spring and structural spring

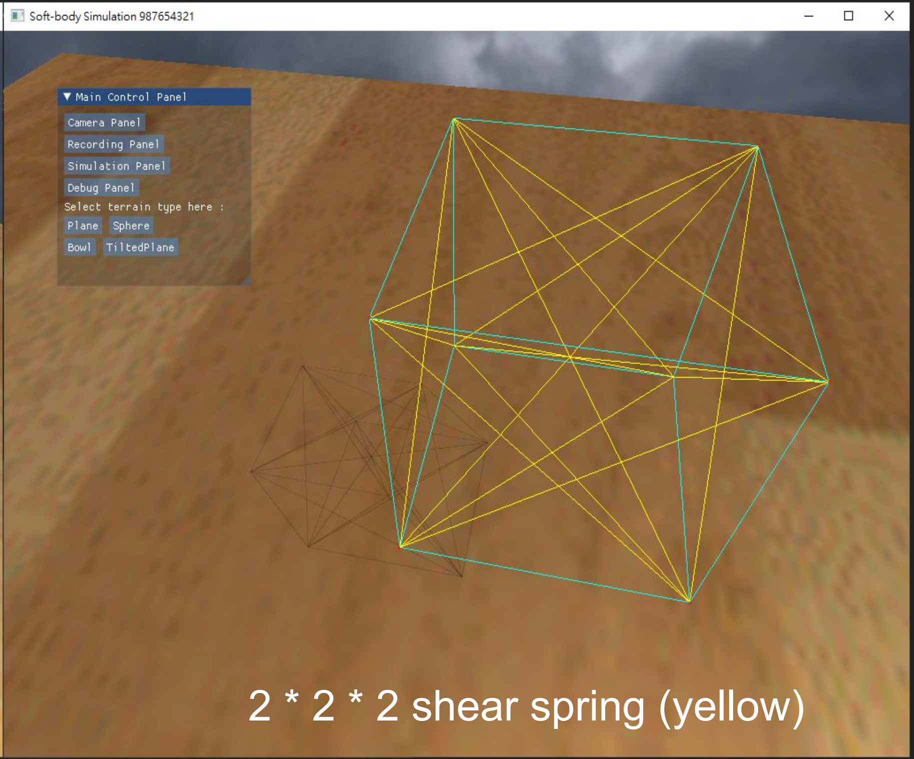
- Green: bend spring
- Blue: structural spring

The curves were drawn by MS paint.



Hint and Reminder (cont.)





Hint and Reminder (cont.)

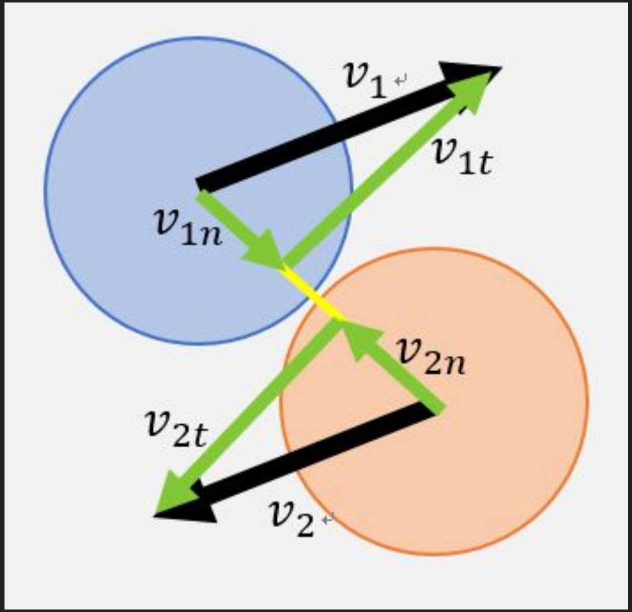
- Hint: point-plane collision
    - Review “particles.pptx” from p.14 - p.19
    - Should be the exact same as tilted plane.
    - If you only implement the horizontal plane collision detection, you will get a part of score.

Hint and Reminder (cont.)

- Hint: sphere/bowl collision
    - You can assume the terrain will not move
        - that is, its velocity is 0

$$v_1' = \frac{v_{1n}(m_1 - m_2) + 2m_2 v_{2n}}{m_1 + m_2} + v_{1t}$$

$$v_2' = \frac{v_{2n}(m_2 - m_1) + 2m_1 v_{1n}}{m_1 + m_2} + v_{2t}$$



Hint and Reminder (cont.)

- Hint: Integrator::integrate
  - You should update particles' velocity and position.
  - You probably need to call

    void MassSpringSystem::computeCubeForce(Cube &cube)

    for getting future information in implicit methods

Hint and Reminder (cont.)

- How to output video?
  - Press "start recording" in recording panel UI.
    - this will output screenshots to the "Screenshots" folder
  - Execute utility\gen_video.bat
    - this will combine the images in "Screenshots" folder and output a .mp4 video
  - You can edit the batch file to adjust frame rate or other parameters.
    - But for the uploading videos, please adjust the frame rate to 60 (default)
  - Also, please include your simulation parameters in your video.
    - By having some part of video showing the simulation panel.
    - offset, rotation, spring coef, damper coef ...

Hint and Reminder (cont.)

- How to properly report bonus?
  - Mention it in your report.
  - If your implementation violates with the original implementation, please make a toggle for switching.
  - If your bonus hides any original feature, you will get not get the score for the features that the TAs cannot test.

Hint and Reminder (cont.)

- How to contact TA?
  - Please ask your question on new E3 forum.
    - or send email to TAs via new E3.
  - If you need to ask question face-to-face, please send email for appointment.

  - IMPORTANT: please sort out and arrange your question, so we can help you without wasting time on trivial matters.