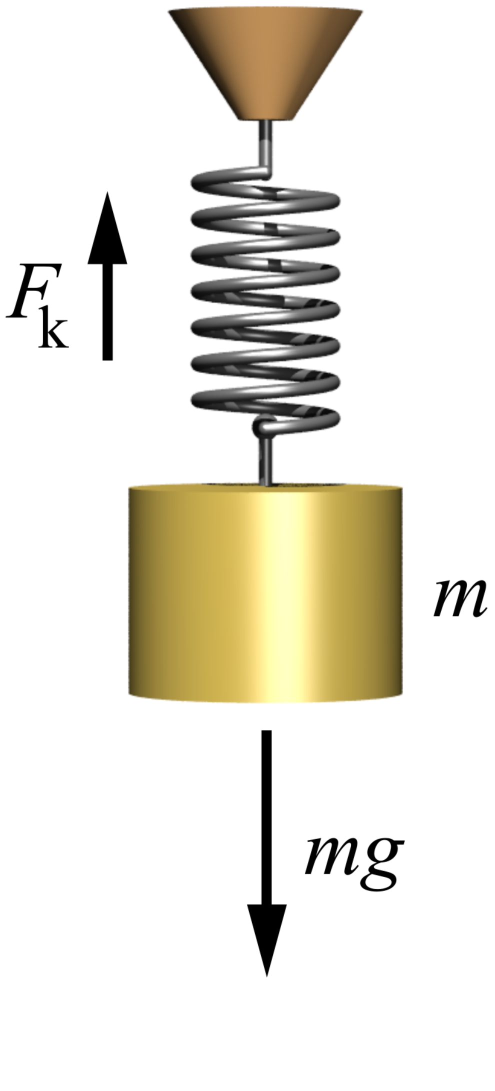
Practical 4: Mass-Spring Systems, part 1

Babis Koniaris



Introduction

The goal of this practical are to:

- Revise our architecture to support the implementation of mass-spring systems
- Develop a practical understanding of constrained particle systems
- Implement 1D mass-spring system simulations

Architecture changes

To assist with the requirements of constrained particle systems, a slightly modified is provided for you: “03_constraints_framework”. There are only a few key differences compared to the previous framework:

- The Particle class now stores accumulated (per update iteration) forces and impulses. During simulation, you first clear the forces and impulses (Particle::ClearForcesImpulses), the process all your particles and apply and required forces or impulses (Particle::ApplyForce and Particle::ApplyImpulse), and finally, at the integration step you will get the accumulated forces and impulses in order to compute acceleration, new velocity and new position for each particle.
- The forces are now more explicit, and we provide a function for each. You can see the function declarations in Force.h. Only the gravity implementation has been implemented for you, so you can implement the rest using the formulas in the lecture materials. When you calculate a force, you should call .ApplyForce(...) on the particle, to add the force into its list of forces. You can see that Hooke force is a bit special, as it applies (opposing) force to two particles. You will notice that the particles are passed in by reference, so that we can modify them (their accumulated force, via ApplyForce).

You can copy over relevant code from your previous project, so that your timestep and integration functionality is already implemented. The part that we will be changing in these two practicals is the application and evaluation of forces and impulses.

Tasks

Task 1: Port integration and timestep to new project

Before you start working on mass-spring systems, make sure that this new project includes integration (one method will suffice) and timestep from the last project, so that you have a single particle bouncing around a cubic room, or just off the ground. While the code for integration and timestep should be mostly identical (and copied over), the part that changes is how do you get a vec3 for the force, and a vec3 for the impulse. The previous section describes the changes that need to be done to achieve that, and can be summarised as:

- At the beginning of a particle's simulation step, call particle.ClearForcesImpulses()
- During the particle's simulation step, resolve collisions as needed, and call particle.ApplyImpulse(impulse) for a calculated collision impulse
- During the particle's simulation step, apply forces as needed, e.g. Force::Gravity(particle);
- At the beginning of the particle's integration step, evaluate accumulated forces and impulses using particle.AccumulatedForce() and particle.AccumulatedImpulse(), before proceeding with your integration formulas

Task 2: Hooke's law implementation

Once your updated architecture is in place, you can add the implementation of Hooke's law.

Task: Implement Force::Hooke(...) in a way that simulates the behaviour of a spring-damper force

Tips:

- Test this on two particles connected by a spring. Keep one stationary, and run the simulation on the other one.
- Ignore the damping force to begin with and add it once your spring force works.
- The size of your timestep will impact the stability of the system. Start with a timestep (dt) of 0.01 and see what happens when you increase/decrease that value.

- The stiffness of the spring will also significantly affect the behaviour of the system. You shouldn't have any problem with a single spring, but later on, as you connect many together, you will have to tune both the timestep and stiffness to achieve stable and believable results ...

Task 3: Chain of particles connected by springs

Produce a simulation that looks a similar to mass_spring_systems_1d_a.exe, included in the github repository folder "examples". Note that the goal is not to emulate the specific behaviour of that simulation, but to create a physically believable simulation of a chain of 5 particles connected by springs.

Task: Simulate a chain of 5 particles connected to each other sequentially by a spring. One extremity of the chain should be fixed. The system is assumed to be under the influence of gravity.

Further tasks

You have connected particles together in 1D chains and you have seen in the lecture that this can be extended to 2D or even 3D structures. Feel free to anticipate next week's lab and experiment with 2D structures.

Deliverables

Nothing is assessed this week, but part of the work (task 3) will be part of the assessed deliverables of next week's practical.