Spring Simulation

Simulation of a string of beads.

Purpose is to demonstrate:

- Use of coordinate class `pCoor` and vector class `pVect`.
- Simulation of a spring.
Simulated World

- *Point masses* connected by *ideal springs*.
- First and last ball can be frozen in space.
- Balls can collide with platform but not each other.
- Each ball has its own mass.
- All springs are identical.
Hooke’s Law

Describes an ideal spring.

The force, $f$, is

$$f = h(l - l_r)$$

where $h$ is the *spring constant*,

where $l$ is the current length of the spring,

$l_r$ is the *relaxed* length of the spring.
Variables for Physical State

**Ball class**

No changes between *demo-1-simple* and *demo-2-springs* to the *ball* class were necessary to model the spring. But some members were added to make the simulation more interesting. The ball class in demo-1-simple had members only for position and velocity. Here we add members *mass* and *radius* so we can look at the effect of different masses.

**Ball Array**

Variable *World::balls* holds an array of *Ball* objects. This is allocated in the *World::init* function, and the ball objects are initialized in routines *ball_setup_1, ball_setup_2*, etc. (The different routines put the balls in different patterns, for example, *ball_setup_2* forms them into a pendulum.)
Spring Modeling

There is no special class used for springs. A spring is assumed between each pair of adjacent balls. (Adjacent based on the position in the balls array.) The length of the spring is just the distance between the respective balls. Each spring has a relaxed length, at which it gives no force (it will just stay at that length if no external forces act on it). Variable World::relaxed_length is the relaxed length for all springs.

Variable World::opt_spring_constant holds the spring constant \( k \). To model spring friction two different spring constants are used. When a spring is being stretched (its length is larger than its relaxed length and getting larger) or compressed (its length is smaller than its relaxed length and getting smaller) spring constant World::opt_spring_constant is used to compute forces. When a spring is relaxing (its length is larger than its relaxed length but getting smaller, or its length is smaller than its relaxed length and getting larger) then World::opt_spring_constant \* 0.7 is used to compute forces.
Coordinate and Vector Classes

The code uses a coordinate class `pCoor` and vector class `pVect` for the respective quantities. These classes are defined in file `coord.h`, and were written for this course. (There are many similar libraries one can use.)

File `gpup/demo-0-coor.cc` shows examples of how to use these.