CS 205b / CME 306

Application Track

Homework 4

Description

In this homework assignment, you will write a very simple rigid body simulator that integrates a rigid body with very simple forces.

This assignment will require a fair amount of time, so it would wise to start the assignment very early. The rigid body material is in lecture 8. It is likely that this material will not be covered in much detail in lecture, but it will be covered during section. It would be a good idea to start working on the assignment before section.

Input

The program should be invoked with the commandline

```
./evolve <n> <t>
```

where the first argument (n) is the number of steps to take, and the second argument (t) is the amount of time over which the rigid body should be evolved during those n steps. Thus, each step will be made with $\Delta t = t/n$. The rest of the information is read from stdin, which will consist of lines, each of which will take one of the forms:

```
# This is a comment.  The next line is blank and should also be ignored.

d <density>
v <x> <y> <z>
w <x> <y> <z>
f <x1> <y1> <z1> <x2> <y2> <z2> <x3> <y3> <z3>
g <x> <y> <z>
s <ks> <kd> <x0> <x1> <y1> <z1> <x2> <y2> <z2>
```

Blank lines and lines starting with a # character (not necessarily followed by a space) should be ignored. All other lines will begin with one of the six characters followed by a space: g, d, v, w, f, or s. The rest of the line will consist of floating point numbers separated by spaces. These lines may contain trailing spaces, but no line will contain leading spaces. Anywhere a space can occur, multiple spaces may occur instead. The individual line formats are described below.

d: This line contains just one floating point number, which is the density of the body. There will be exactly one of these lines.

v: The arguments specify the initial velocity. There will be exactly one of these lines.

w: The arguments specify the initial angular velocity. There will be exactly one of these lines.

f: There are nine numbers on this line, which are the coordinates of the three vertices of a triangle. Each such triangle is one face of the rigid body's boundary surface. There will always be multiple lines of this form in the input. The three vertices occur in the "standard" orientation (counterclockwise, looking at them from the outside). All of the examples in this document should yield a positive volume.

g: This line indicates a body force (such as gravity). Its three arguments specify the acceleration vector the body would feel if no other forces acted upon it. Multiple forces of this type may occur.

s: This line provides a description of a simple spring. The first three parameters correspond to the spring parameters k_s, k_d, and x_0. The remaining six parameters are two vectors. The first vector is a point the rigid body, given by its position in space at the beginning of the simulation. The second point is a fixed position in space. Note that the first point will move along with the rigid body, and this point need not line inside the rigid body's surface mesh. The spring connects these two points. There may be multiple lines of this type.

An example input file is

```
g 0 0 -9.8
d 1.2
v 0 0 1
w 0 3e-1 0
f 1 0 0 0 1 0 0 0 1
f 1 0 0 0 0 0 0 1 0
f 1 0 0 0 0 1 0 0 0
f 0 1 0 0 0 0 0 0 1
s 1 .2 .5 0.1 0.2 0.1 0.1 -.3 0.1
```

This corresponds to a rigid body (a tetrahedron) falling under gravity (where up and down is the z direction), density $\rho = 1.2$ and moving up. The body is initially spinning around the y axis. A spring, which is in its rest configuration, is attached to the rigid body. The volume of this body should be $\frac{1}{6}$, and its mass should be $\frac{1}{5}$.

These faces correspond to a cube with edge lengths 2 centered at the origin:

```
f -1 -1 -1 -1 1 -1 1 -1 -1
f 1 1 -1 1 -1 -1 -1 1 -1
f -1 -1 -1 1 -1 -1 -1 -1 1
f 1 -1 1 -1 -1 1 1 -1 -1
f -1 -1 -1 -1 -1 1 -1 1 -1
f -1 1 1 -1 1 -1 -1 -1 1
f -1 -1 1 1 -1 1 -1 1 1
f 1 1 1 -1 1 1 1 -1 1
f -1 1 -1 -1 1 1 1 1 -1
f 1 1 1 1 1 -1 -1 1 1
```

```
f 1 -1 -1 1 1 -1 1 -1 1
f 1 1 1 1 -1 1 1 1 -1
```

These faces correspond to the unit cube in standard position:

```
f 0 0 0 0 1 0 1 0 0
f 1 1 0 1 0 0 0 1 0
f 0 0 0 1 0 0 0 0 1
f 1 0 1 0 0 1 1 0 0
f 0 0 0 0 0 1 0 1 0
f 0 1 1 0 1 0 0 0 1
f 0 0 1 1 0 1 0 1 1
f 1 1 1 0 1 1 1 0 1
f 0 1 0 0 1 1 1 1 0
f 1 1 1 1 1 0 0 1 1
f 1 0 0 1 1 0 1 0 1
f 1 1 1 1 0 1 1 1 0
```

Note that the input state does not include an initial position, orientation, inertia tensor, or mass. This information must be computed from the surface mesh and the density. The surface mesh is needed only to compute these initial properties and should not be stored during simulation. The input mesh description does not provide the surface mesh's topology, since this information is never required.

Output

The program outputs its initial state and its final state to stdout. Each state dump should contain the following lines, in the order shown below. The output should consist of a state dump immediately after initialization, followed immediately by a state dump of the final state. Nothing else, even debugging information, should be printed to stdout. Debugging information may be printed to stderr, as it will be ignored. The program should not read or write any files.

```
x <x> <y> <z>
v <x> <y> <z>
R <r11> <r12> <r13> <r21> <r22> <r23> <r31> <r32> <r33>
w <x> <y> <z>
m <mass>
I <m11> <m12> <m13> <m21> <m22> <m23> <m31> <m32> <m33>
p <x> <y> <z>
L <x> <y> <z>
K <kinetic-energy>
E <total-energy>
```

The vector quantities x, v, w, p, and L are the position, velocity, angular velocity, momentum, and angular momentum of the rigid body. The scalars m, K, and E are the mass, kinetic energy, and total energy. The total energy includes the kinetic energy, the potential energy of all body forces and springs, and the potential energy should shifted so that the total energy is zero in the initial configuration.

Time Integration

The time integration scheme used is not fixed, except that it must satisfy a few requirements.

- The position of the center of mass should be evolved with second order accuracy so that objects fall under gravity with only roundoff error. The evolution of velocity, angular velocity, and orientation need only be first order accurate.
- Orientation should be evolved using a scheme that keeps the orientation normalized (up to roundoff) without the need for normalization. Thus, a scheme like Runge-Kutta is not suitable for evolving orientations, but the scheme in question 2 of the third theory homework is.
- Orientation should be stored internally and evolved as a rotation matrix.
- If the rigid body experiences no forces, it should exactly conserve momentum (excepting for roundoff).
- If the rigid body experiences no torques, it should exactly conserve angular momentum (excepting for roundoff). In particular, angular momentum should not receive first or second order errors.
- If the time integration achieves second order accuracy (second order accurate position, velocity, orientation, and angular velocity), two extra points will be awarded. It is much more important to focus in correctness, so this should be undertaken last if at all.

Submission

Each submission should be a single tar.gz file. This compressed archive should contain all of the code and a makefile for building the executable. The archive should not contain subdirectories; everything should be at the base level. The archive should also contain a README file containing your name, in addition to any other information you would like. The tar.gz files should be emailed to the application track CA by the deadline. The file may be attached to the email, or the file may be hosted elsewhere, and the email may contain a link to the file. In this case, the uploaded file should not be changed after the deadline and must be accessible at the deadline.

The program should be written in C++ and use no libraries beyond the language libraries, and the program should build under a gcc. It is acceptable to work with a partner for the purposes of planning the assignment and working out technical details. Each student must write a unique program. In particular, it should be difficult to determine who partners were by examining the source code.

Testing

The following is a list of tests that may be helpful to perform and some assumptions that may be made. I will be using at least some of these when checking correctness.

- Homework 3 discusses invariance of the simple spring. Each of those applies to the rigid body as well. If units are scaled in the input (including possibly the commandline arguments), the same output should be obtained as if the units in the output of the original simulation were scaled instead. The same is true of rotating and translating space. You will need to work out how orientation, angular velocity, angular momentum, and the inertia tensor should be transformed.

- The input time may be zero, in which case the two states printed to stdout should match. The number of steps taken will always be a positive integer, and the input time will never be negative.
- Orientation should remain an orthogonal matrix, up to roundoff.
- Mass should be positive and never change. The input density will always be positive.
- The inertia tensor should always be symmetric and positive definite.
- Kinetic energy should never be negative. Don't forget that there is a linear contribution and an angular contribution.
- If there is gravity but no springs, the analytic solution is known and can be compared against. One of the requirements of the time integration scheme is that this solution should be obtained correctly up to roundoff error.
- If there are no forces, kinetic energy should remain constant, except perhaps for up to a first order error that should vanish as the number of steps taken is increased. Momentum and angular momentum should not change by more than roundoff error, and uniform translation should be observed. Note that angular velocity may still evolve in a complicated way.
- If a body experiences no forces and has an angular velocity that coincides with an inertial axis, then the angular momentum should be aligned with the angular velocity, and both remain unchanged. In such a case, the evolution of the body can be computed analytically.
- If there is no gravity and only a single spring attached to the center of mass of the rigid body, the system will follow the motion of a simple spring, whose analytic solution can be compared against. This is easiest to do if the initial velocity and displacement are aligned, so that the spring can be treated in 1D.
- Convergence can be tested by changing the number of steps taken and leaving all other inputs unchanged. Every simulation should converge.
- If multiple forces (especially multiple springs) are present, the results obtained should not depend on the ordering of those forces.
- A system with a damped spring should eventually reach steady state, and the steady state solution will be easy to determine analytically if there is only one spring. In particular, the force of gravity should be equal and opposite to the force of the spring, and the spring's attachment point should be directly over the body's center of mass.
- If no springs are attached to the rigid body except at its center of mass, the rigid body should experience no torques. The linear and angular dynamics of the rigid body should remain independent, so varying angular velocity, for example, should not affect velocity. The linear and angular contributions to kinetic energy should also be independently conserved, up to simulation accuracy.