Developing a Rigid-Body Dynamics Physics Engine

Neal Milstein
Computer Systems Lab

Abstract

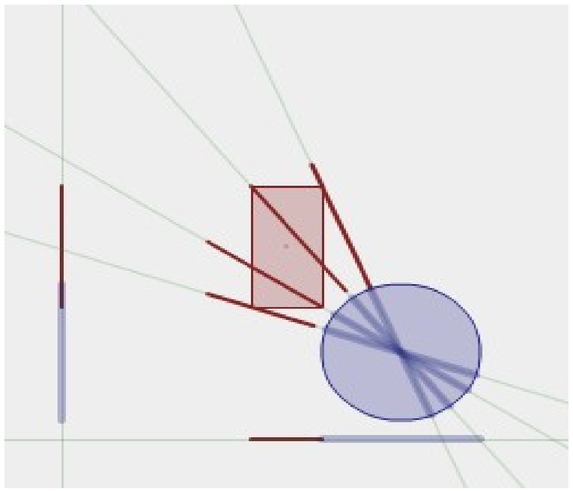
The goal of this project is to create a rigid-body dynamics physics engine in order to allow physics students to visualize and solve physics models. Rigid-body dynamics is the study of the motion of non-deformable, free-moving bodies through space. Such setups involving rigid bodies are prevalent throughout physics courses, leading to the value of such a simulation. The engine will improve upon many current models by focussing on approximating mathematic techniques with a small percent error, such as Runge-Kutta 4 integration and the Separating Axis Theorem. The physical responses to these collisions will be calculated by representing rigid bodies with grids of ellipses, approximating the physical responses to collisions. The engine interface will allow for the speedy input of a variable number of rigid body and interaction mechanisms, specifically optimized for an educational environment.

Background

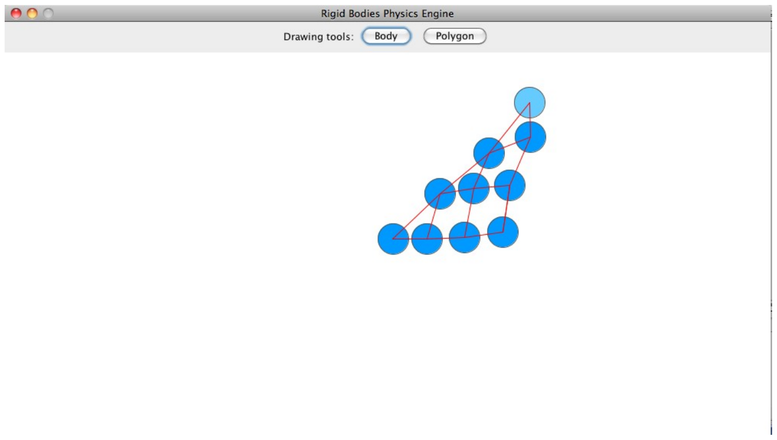
Physics engines present the unique problem of combining small-scale rigid-body oriented interactions with large-scale physics engine scalability. Research into rigid body dynamics engines can be broken down into the microcosmic lower-level interactions between rigid bodies, and the macrocosmic large-scale design structures of physics engines. On the microcosmic level, the simulation uses the Runge-Kutta 4 methods of integration and the Separating Axis Theorem to calculate impulse forces and collisions. On the macrocosmic level, the model-view-controller design is used by the engine to separate collision detection algorithms and other interactions from the rest of the engine.

Collision Response

To model the responses to collisions, my program represents bodies as grids of ellipses. If enough ellipses are used (around 64 per virtual square meter), the the collisions between the meshes will simulate collisions with a small degree of error.



The separate Axis Theorem used for Circular Objects. If all all of the separating axes (represented here by the thick lines) are intersecting, then the two objects are colliding.



A triangular shape represented by a grid of ellipses. This setup is used by the simulation to model responses to collisions.

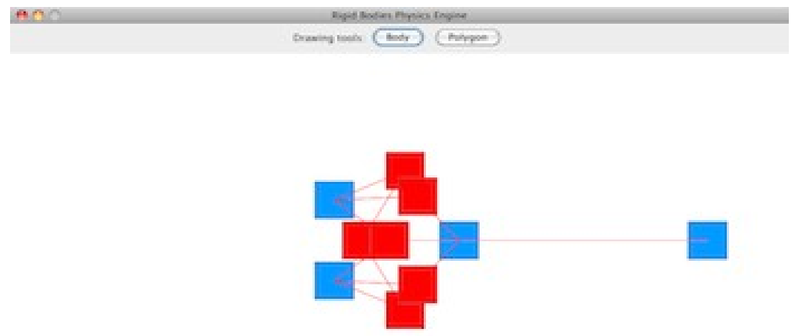
Developments

The engine uses the Model-View-Controller (MVC) programming paradigm to develop my project. The architectural program is used by a number of software systems, such as Microsoft's .NET framework, the open source Ruby on Rails framework, and a number of common software platforms, such as the Facebook application platform and twitter. The MVC design differs from the typical two-part design of physics simulators, where the interface is separated from the simulator, by adding a third module: the model, used to represent the underlying data of the engine.

Discussion of Results

The results of the physics engine are primarily graphical, with currently very little numerical output to measure, but enough data can be obtained to determine the accuracy of the physical interactions within the engine. Of primary concern is the accuracy of the grid model used to simulate collision responses. I found that I need to use at least 64 ellipses per virtual square meter (approximately 2 square inches on a standard screen) to believably simulate collision interactions. The simulation ran with no lag at this rate, and the collisions were still accurate. After about 20 seconds, however, the collisions begin to deviate signifi cantly from the expected results with only 64 ellipses per square meter. I upped this number to 256 ellipses per virtual square meter, but this caused a small amount of lag in the simulation, which required me in turn to lower the frame rate. The exact number of ellipses used by my simulation depends on both the computer architecture it is run on, and much processing power the computer can allot to the simulation at a given time. For this reason, I plan to add a scaling calculation that dynamically changes the number of ellipses used in the irregular objects, so the simulation is as accurate as possible.

In one physical setup found in a research article, an irregularly shaped "spaceship" is sent through a moving sea of "astroids," which are irregular objects designed to collide with the spaceship. The article discusses the outcomes of such a simulation, and it details how certain collisions will impact the overall setup. For example, the article tries to predict the number of collisions that will result when different shapes for the spaceship are used. I tried simulating this model in my simulation, and achieved results very similar to those predicted in the article. For example, when the "spaceship" was fi red into a "sea" at a rate of one new convex shape per second, I recorded 5 collisions, which was in this case the exact same number of collisions reported in the article.



An example setup, where a grid of bodies is connected via springs. I used this model to ensure mechanical energy was conserved in my program.