DM815 Computer Game Programming III: Physics

Spring 2009

Department of Mathematics and Computer Science
University of Southern Denmark

February 10, 2009

Introduction

The purpose of this project is to try some of the methods from the course during actual programming.

The project is to be done in groups, preferably of size two. Groups of size one and three are allowed.

Requirements

You are to implement one or more programs testing and visualizing some of the methods from the textbook. The visualization should be in 3D.

At least one non-trivial techniques from each of the areas of collision detection and physics simulation should be included. For the area of collision detection, this means some techniques from Chapter 6 and onwards (with necessary elements from earlier chapters). For the area of physics simulation, this means physics based object movement and collision handling, including angular momentum.

You may build on a project from DM809/DM810 by adding collision detection and physics simulation to the program, or you may create stand-alone programs testing and visualizing the chosen techniques in some relevant setting defined by you.

If you choose the latter, it may be beneficial to make a suite of programs building up to one or more containing the required complexity. A concrete suite, based on moving objects in a box, could be:

- Start with rotationally invariant objects such as spheres. Do a simple particle simulation (no angular momentum, simple collision response of the type angle-in-equals-angle-out), and naive $O(n^2)$ all-objects-against-all-objects collision testing. Do the collision testing just based on distances (no triangle-triangle tests)..
- Change the spheres to a more complicated polygonal model, and include the triangle-triangle test (all-triangles-against-all-triangles for each object pair tested).

- Add a spatial structure (Chapter 7) to get rid of the all-objects-against-all-objects testing, and/or (one of these is enough) a hierarchical structure (Chapter 6) to get rid of all-triangles-against-all-triangles testing for object pairs.
- Add angular momentum and related collision response to the physics simulation (maybe using simple rotational non-invariant objects like boxes, and reverting to one of the simpler collision testing schemes from previous programs).

As an extra (not required), it could be nice to compare experimentally the various levels of sofistication of collision detection in the program suite with respect to how many objects can be simulated with a given frame rate.

Formalities

You should hand in: Some kind of executable program or installer (must run on either Imada computer lab machines, or on Windows (XP or Vista), with no special actions needed for installment), source code files, and a report of 8-10 pages (excluding any appendices) in pdf-format. The main aim of the report should be to describe the design choices made during development, the reasoning behind these choices, and the structure of the final solution, as well as give a simple user manual for the program. The user manual should include installation instructions. The source code should not be included in the report, but just be supplied as raw files.

The project will be evaluated by pass/fail grading. The grading will be based on:

- The clarity of the writing and of the structure of the report.
- The ability to apply the concepts of the course.
- The amount of work done.

You are welcome to incorporate public code on the web for significant parts of the project (e.g. for triangle-triangle tests), but must clearly indicate in the report text which parts of the code that is your own.

The material should be handed in using the `aflever` command on the Imada system: Move to the directory containing your code, executable, and report, and issue the command `aflever DM815`. This will copy the contents of the directory to a place accessible by the lecturer. Repeated use of the command is possible (later uses overwrite the contents from earlier uses). In the directory, you must for identification purposes have an ASCII file named `names.txt` containing the names of the group members, with one name per line.

You must hand in the material by

Monday, March 23, 2009, at 12:00