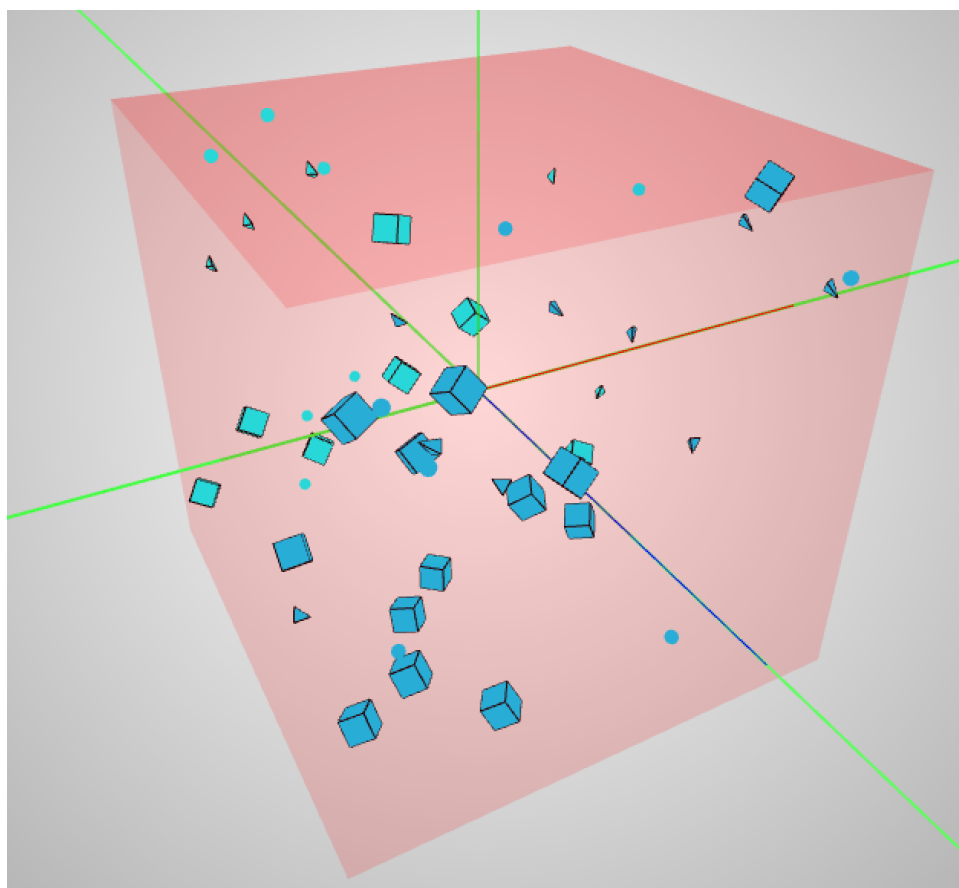


PHYSICALLY-BASED MODELING, SIMULATION AND ANIMATION
ASSIGNMENT 2: COLLISION DETECTION BETWEEN RIGID BODIES

ZHEN WEI

October 24, 2016

More results and demo can be found at <http://cs.unc.edu/~zhenni/courses/UNC/COMP768/>



Problem A: Implementation Write a program to detect collisions among many sphere-like, ‘fat’ polyhedra of different shapes and sizes flying inside a confined region (e.g. a cube, a table with boundary, etc); Each object is given an initial random velocity and angular velocity. The velocity stays the same unless an object hits an obstacle (e.g. a wall or another object), in which case a component of the velocity should be flipped so that the object stays within the confinement. State your assumptions about the objects! You may wish to reuse and modify the source code available at: <http://gamma.cs.unc.edu/software/#collision>

Problem B: Analysis Analyze performance of your algorithm (or implementation) in different situations. Change the following parameters and measure computation time:

1. Number of objects (1-50)
2. Complexity of objects
(The number of polygons for each object. Try at least 3 different data sets)
3. Sizes of objects
(The relative size of each object w.r.t. the extension of the bounding cube.)

Your algorithm (implementation) should exhibit different characteristics to the variation of these parameters. Present data in tables and graphs, and explain the differences in performance.

Solution:

In this assignment, I use **SWIFT** library to implement the part of collision detection .I use three kinds of polyhedra, namely, tetrahedron, cube and sphere. Another thing should be noticed is that all the computation time mentioned below is the time consumed by 10000 times of updates.

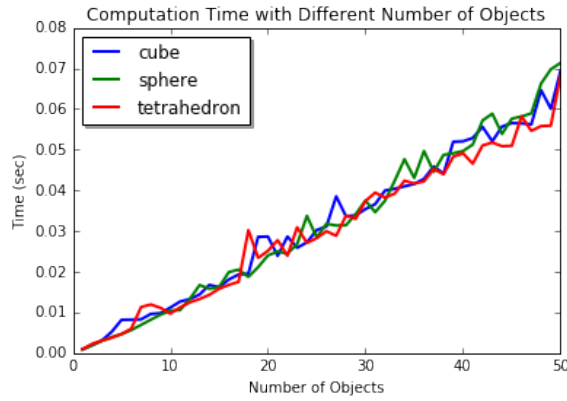


Figure 1: Number of Objects

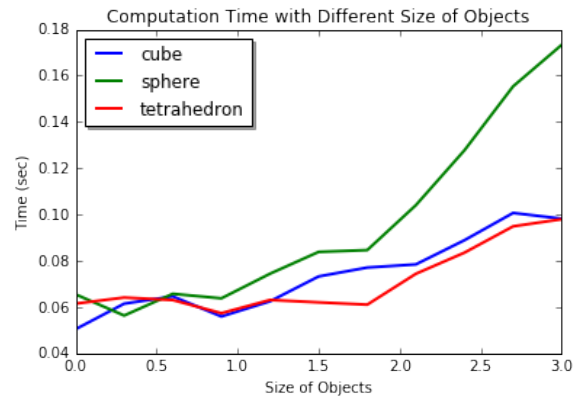


Figure 2: Sizes of Objects

1. Number of objects (1-50) :

Figure 1 shows that, with fixed size and complexity, cases with more objects will consume more time while computing the detections. .

2. Sizes of objects :

From Figure 2, we can know that, computation time increases along the size of objects, which have fixed amounts and complexity.

3. Complexity of objects : Figure 3a and Figure 3b

Figure 3a is the results with respect to the number of polygons in each object with the same size and amount. For Figure 3b level of complexity means that, a object of Level $(i + 1)$ is the results object by equally dividing all the faces of the object of Level i into four faces. Examples can be found in Figure 4a and Figure 4b. By this method of dividing faces, we can get a bunch of objects with the *exactly* same size. Figure 3b illustrates the results with respect the level of complexity. Figure 3a and 3b demonstrate that same amount of objects with the same size

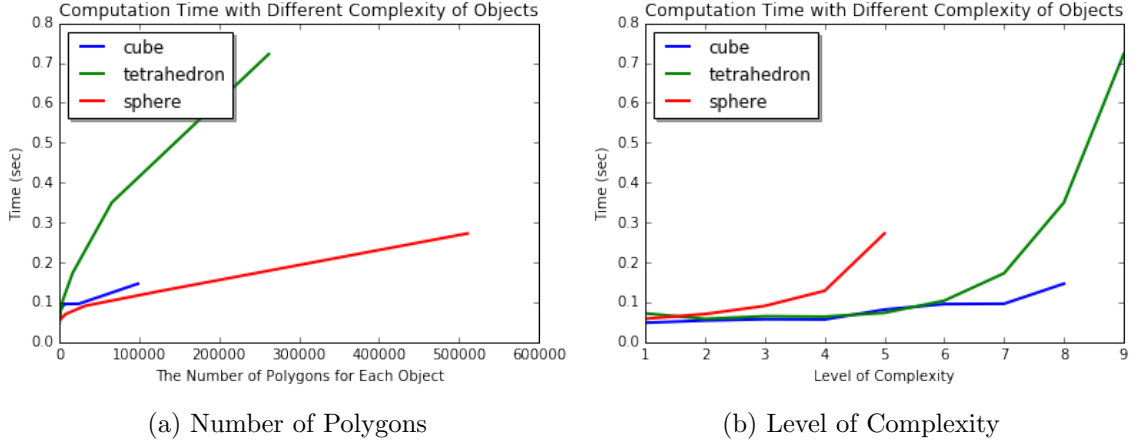


Figure 3: Complexity of objects

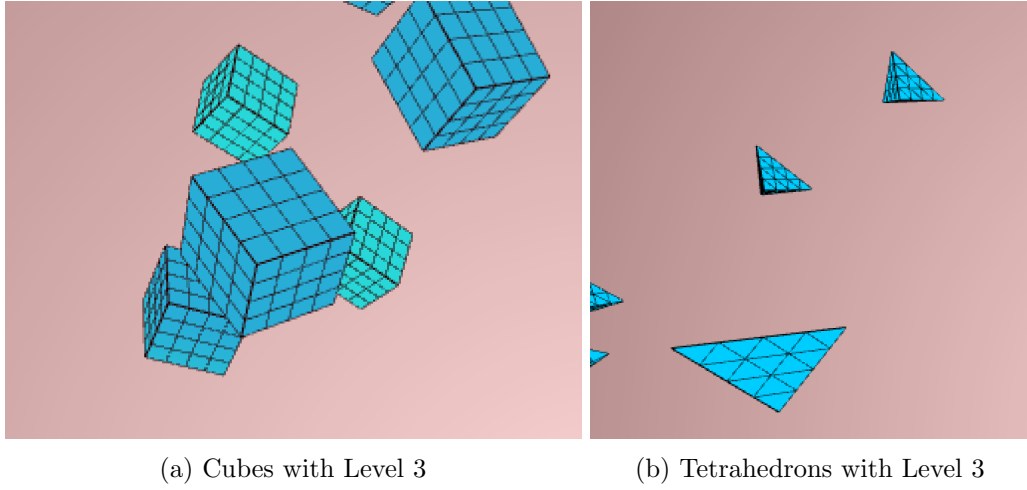


Figure 4: Example

Problem C: Varying Parameters Same as Problem A, except the objects have the same size. What would you do differently? Show a simple prototype implementation to compare and contrast the performance of two different algorithms (implementation).

Solution:

In this problem, since the objects have the same size, we can easily implement a bounding sphere or cube algorithm to coarsely detect the collision and simulate. Figure 5 shows that the computation time using different methods while the number of objects are changing. From the figure, we can know that , although the detection is some kind of coarse, our new simple algorithm runs much faster than the origin algorithm we used.

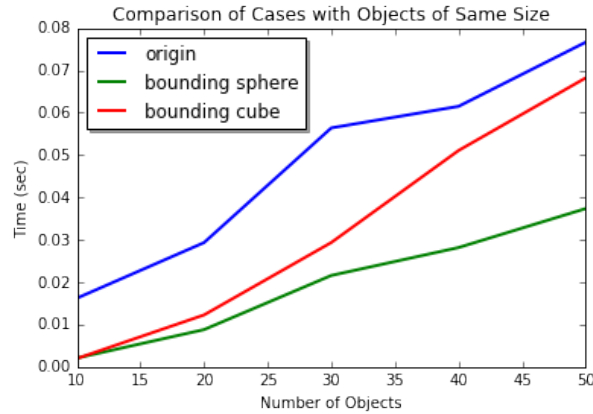


Figure 5: Complexity of objects