Physically Based Animation Assignment 2

Yinan Wang 48-176630 Enea de Bollivier 48-179717

Introduction

In this assignment, we implemented an animation system without using any external animation libraries for calculation. This work is done in Java using Jogl for visualization in 3D. Jogl is a java library that implements OpenGl.

To compile the programs, You will need to download the required libraries jogl and Apache Commons Math 3.6.1 available here :

http://ftp.riken.jp/net/apache//commons/math/binaries/commons-math3-3.6.1-bin.zip

(If the above link does not work you can find jogl here (Binaries) :

https://commons.apache.org/proper/commons-math/download_math.cgi

http://jogamp.org/deployment/archive/master/gluegen_646-joal_408-jogl_930-jocl_756/archive/jogamp-all-platforms.7z

(If the above link does not work you can find jogl here (Downloading the latest aggregated autobuild) :

http://jogamp.org/wiki/index.php/Downloading_and_installing_JOGL

Then decompress the libraries and put the entire directory in the lib/ directory in the root of the project. The Makefile does automatically look for the libraries in this directory.

To know how to use the different classes, refer to the README.txt in the root of the project.

Task 1

Based on the Task2 of assignment 1, we change the object from particle to cube. We simulate the cube as a cluster of particles and add meshless shape matching algorithm.

Task 2

Reusing the clustering algorithm of Task4 of the first assignment to find the possible collision pairs, we add a checkCollision method for cube in class Square for both checking and responding collisions.

For each possible colliding pair, we check if there’s at least one summit inside the other cube. If yes, we then apply two opposite spring forces to cubes respectively. Based on physics, we translate the force to the center of mass of each cube, this act also generates a torque which we use the equation of angular motion to calculate the angular velocity. We also use the method for border response.

Task 3

For this task, we modified the shape matching algorithm of task 1 to add quadratic deformations. We compute the cube as a large amount of particles to allow different type of deformations.

Task 4

In this Task, we use the Sallow water equations to simulate the field of the depth of water.

Task 5

For this task, we compute the rope as many elastic cubes with particles in common. We then shape match every one of them. This way, the rope is divided in clusters. And each one of them compute a quadratic deformation. We managed to simulate the rope but when adding the cube at each extremity. To do so, we replace a particle at each extremity by a cube. When we did that, the cube became unstable. We think that this is because of a mass problem in our matrix. Also the rope does not react to collision with other objects.