|  |  |  |
| --- | --- | --- |
|  | **Rochester Institute of Technology**  **Golisano College of Computing and Information Sciences**  **School of Interactive Games and Media**  **2145 Golisano Hall – (585) 475-7680** |  |

**Data Structures & Algorithms for Games & Simulation II**

**IGME 309, 2014 Fall**

**A7: Bounding Box Manager**

Due: Nov/09/2014

This is homework assignment is meant to be completed in pairs. For this purpose a repository needs to be setup. The way you will deliver your assignment is by providing me and the grader access to your repository, this could be done freely in github or bitbucket, you will also need to provide your final solution to the labeled dropbox in MyCourses. The purpose of the repo is not only to facilitate your workflow but also to see who is doing what in the code, so we are expecting to see at least 1 commit from each team member. I do not tolerate a slacker and neither should you!

There should be a Readme file in your .zip file submission to my courses and in your repo, stating the name of the team members and what part of the assignment each one fulfill.

Only one submission to MyCourses is required, if you have more than one submission (if you submitted something and improved on it after, for instance) this new submission needs to be done in BY THE SAME team member.

10% of your grade is a peer-evaluation, this evaluation is personal and needs to be submitted in the same dropbox in MyCourses as the homework assignment. In it you will grade from 0 to 10 how useful your partner was AND WHY. Failing to provide a reason for the assigned grade or failing to submit this file will result in losing this 10% from YOUR grade.

The goal of this homework assignment is to get familiar with Axis Aligned bounding Boxes and Oriented Bounding Boxes (implementation and collision detection); get familiar with teamwork through repositories and as practice before the final project.

Using the provided code for this homework assignment you will implement a Bounding Box class which will be able to create Oriented Bounding Boxes and Axis Aligned Bounding Boxes; calculate their position in the world and finally draw it on the screen surrounding different objects.

After the creation of the Bounding Box class you should create a box manager through a singleton which will let the user create new boxes for different objects and destroy existing boxes, manage the different transformations of those boxes and calculate intersections among all the boxes in the class.

You can base your code on the Bounding Sphere Class and the Bounding Sphere Manager singleton class provided.

For your collision response changing the color of your OBBs and AABBs is more than enough but if you want to go ahead and implement other kind of collision response I can take that as extra.

There is a binary executable example under \_Binary that demonstrates what I’m expecting of this homework assignment.

It is not necessary to use “MyEngine” for this homework assignment, you can implement your own code for this, as usual “MyEngine” is meant to give you a head start on all the functionality that is not relevant to the homework assignment but it is completely optional. If you decide to implement your own code instead of working with it, at the very minimum I will need a framework able to load any number of .obj files and display them with textures on the screen; be able to provide a “model to world” matrix and that can display your bounding boxes. Everything in your code should be commented in such a way that the grader does not take much time reading and understanding the code. There are no extra points for implementing your own framework.

45% of your grade is related to the Bounding Box class.

At a bare minimum it should let the user:

1. Switch the visibility of the OBB
2. Switch the visibility of the AABB
3. Get the Centroid of the OBB
4. Get the Centroid of the AABB
5. Get the Minimum and Maximum of the OBB
6. Get the Minimum and Maximum of the AABB
7. Get and Set the “Model to World” matrix of the Box
8. Set the color of the AABB and have a different color for the OBB
9. Draw the OBB
10. Draw the AABB

45% of your grade is related to the Bounding Box Manager singleton.

At a bare minimum it should let the user:

1. Add a box based on a model
2. Remove any box
3. Get the number of boxes in the manager
4. Set the color of a box
5. Set the visibility of a box
6. Render any specific box or all of them
7. Check the collision of all the boxes in the manager
8. Respond to those collisions.

For either class you can implement as many extra methods as you need.

If you decide to go for any of the extra challenges you should write in the readme file what you did as extra and explain and why do you think you deserve the extra points.

Extra:

15% - Have a different collision response other than changing the color of the shapes, like moving the other objects as well or impeding the movement of the shapes in a spot that is already occupied.

15% - Implementing other collision detection like Box to Sphere or ray to Box, etc.

15% - Implement any spatial partitioning optimization. (Only 15% even if you implement more than one)

Submit to the dropbox labeled A7 - Bounding Box Manager