|  |  |  |
| --- | --- | --- |
|  | **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 Spring**

**A9 - Axis Aligned Bounding Box from Oriented BB**

Due: April/08/2015

This is homework assignment is a continuation of the previous homework assignment and it is still meant to be completed in pairs. For this purpose the work on the repository needs to be continued. 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. As such both partners are expected to have at least 1 meaningful commit to the repository. 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 fulfilled.

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, “Because he or she showed up to the meeting” is not a valid reason for a grade.

The goal of this homework assignment is to get familiar with Axis Aligned bounding Boxes and Oriented Bounding Boxes (implementation and collision detection, no collision response is necessary); get familiar with teamwork through repositories and as practice before the final project.

For this homework assignment you will improve on the Bounding Box class and the Bounding Box manager. As this homework improves on the previous homework, the whole solution for that assignment is provided in the starting code (with the exception that some fields and methods changed their name for the AABB and the OBB can coexist in the same file).

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. Your previous homework should be completely functional at this point if you decide to continue it for this assignment, just base the methods names on the ones I’m providing.

Your grade will be as follows:

80% Translate the Axis Aligned Bounding Box into an Oriented Bounding Box and implement the new Axis Aligned Bounding Box of said OBB.

At a bare minimum the Axis Aligned Bounding Box class it should:

1. Draw the AABB (if you are working with MyEngine, add it to the render list)
2. Draw the OBB(your previous AABB)

20% of your grade is related to the Bounding Box Manager singleton.

At a bare minimum it should:

1. Check the collision of all the boxes in the manager using the new AABB (not the OBB)
2. Respond to those collisions (change the color of the box if there is a collision).

For either class you can implement as many extra methods and fields 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.

Controls provided:

WASD will move the camera in a first person shooter kind of movement.

Arrow keys will move the selected model horizontally and vertically, holding shift will move it backwards and forwards and rotate it in the z plane

F1 to F5 will change the selected model.

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.

Submit to the dropbox labeled A8 – Rough Axis Aligned Bounding Box