10531 Video Game Technologies Proj. 2 - 31/03/2017

Collision Detection

Supervisor: Abel Gomes *Scribe: A. Gomes*

In this second project, the goal is to write a program in C++ that detects a collision between two geometric objects (i.e., meshes) in the 3D scene. Detecting collisions between geometric objects is to be done by implementing bounding volumes, SAT, or GJK in the broad phase, together with triangle-triangle intersection in the narrow phase.

1 Specific Learning Goals

After completing this project, students should know and be able to:

1. To deal with broad phase collision detection and avoidance. This has to do with collisions between space regions in 3D.
2. To deal with narrow phase collision detection and avoidance. This has to do with collisions between triangles in 3D.

2 Requirements

This project must be built upon the geometric data structure of Project #1.

3 Web Links

General repository:
`https://github.com/jslee02/awesome-collision-detection`

Bounding volume collision detection:
`https://en.wikipedia.org/wiki/Bounding_volume`
`http://www.gamefromscratch.com/post/2012/11/26/GameDev-math-recipes-Collision-detection-using-an-axis-aligned-bounding-box.aspx`
`http://www.gamefromscratch.com/post/2012/11/28/GameDev-math-recipes-Collision-detection-using-an-axis-aligned-bounding-box-part-2.aspx`
`http://www.gamefromscratch.com/post/2012/12/12/GameDev-math-recipes-Collision-detection-using-bounding-circles.aspx`

Containment/Intersection Test (C/IT):
`http://www.cs.umd.edu/~djacobs/CMSC427/IntersectionContainment.pdf`
`http://www.tankonyvtar.hu/en/tartalom/tamop425/0046_algorithms_of_informatics_volume2/ch10s04.html`
`http://thebuildingcoder.typepad.com/blog/2010/12/point-in-polygon-containment-algorithm.html`

Separating Axis Theorem (SAT):
`https://en.wikipedia.org/wiki/Hyperplane_separation_theorem`
`http://www.dyn4j.org/2010/01/sat/`
`http://www.metanetsoftware.com/technique/tutorialA.html`

http://www.phailed.me/2011/02/polygonal-collision-detection/

Gilbert, Johnson and Keerthi algorithm (GJK):
https://en.wikipedia.org/wiki/Gilbert%E2%80%93Johnson%E2%80%93Keerthi_distance_algorithm
http://www.dyn4j.org/2010/04/gjk-gilbert-johnson-keerthi/
http://vec3.ca/gjk/implementation/

4 Project Steps

This project consists of the following main steps:

1. To construct a 3D scene on the geometric data structure coded in Project #1.
2. To code the broad phase of two colliding objects. At least one of them is moving in the scene.
3. To code the narrow phase of two colliding objects. At least one of them is moving in the scene.

After completing your project, send it by email to agomes@di.ubi.pt. The .zip archive name should include the student number (e.g., 12345-project2TJV.zip).

The deadline to deliver the project is 8 May (Monday, 23:59). **Projects not working or received after the stipulated deadline will not be evaluated.**

5 Algorithms to Students

Christian Lopes: SAT
Tiago Mendes: GJK
José Manteigueiro: SAT

António da Silva: C/IT
Fábio Machado: SAT
André Louro: C/IT

Miguel Rodrigues: C/IT
João Silva: GJK
José Nunes: GJK