Institut für Informatik — Technische Universität München SS 2016
Scientific **C**omputing in **C**omputer **S**cience
S. Rettenberger, R. Wittmann

Bachelor Lab Scientific Computing (Game Physics) Worksheet 4: Collisions (sphere-box, plane-box), create your own scene

Assignment 1: Implement a sphere-box collision detection

File: *cPhysicsIntersections.cpp*,
Method: *CPhysicsIntersections::sphereBox*
Similar to a previous assignment, you should transform the center of the sphere from world space to the object space of the box. Then, all collision tests can be implemented by testing collisions with axis aligned surfaces. Fill the collision data with appropriate values!

Assignment 2: Implement a plane-box collision detection

File: *cPhysicsIntersections.cpp*,
Method: *CPhysicsIntersections::planeBox*

There are different ways to handle these intersections. The easiest one (even if this is not the most efficient one - see next worksheet) is to transform the box into plane space:

$$M_{to\_planespace} = M_{plane}^{-1} \cdot M_{box}$$

Start by checking all vertices interpenetrating the surface of the plane.

Then test all box edges for an interpenetration with the edges of the plane. Further, the box should be able to collide with the edges of the plane. Introduce respective tests for this scenario!

Assignment 3: Create your own scene

File: *cScenes.cpp*,
Method: *CScenes::setupScene21*
It's time to show us what you can 'design'! This assignment is quite easy: Setup a new *fancy looking* scene. This scene may show limitations of the engine in its current state of implementation. We are looking forward to seeing your experiments! Remark: Feel free to take over pieces of code from other scenes.

Assignment 4: Validate the collision response

File: *cPhysicsIntersections.cpp*,*CPhysicsCollisionImpulse.hpp*

For the collision response we used a couple of assumptions to find formulas for the change of velocities. We can now check if we fulfill these assumptions in order to find out if our implementation is correct. I.e., momentum conservation is fulfilled by making the collision forces sum up to 0. Energy conservation is not fulfilled, because we allow a loss of energy due to the coefficient of restitution. However, this energy loss can be computed analytically and amounts to

$$\Delta E_{kin} = -\frac{1 - C_r^2}{2} \frac{((v_1 - v_2) \cdot n)^2}{m_1^{-1} + m_2^{-1}}$$

Include an assertion in your collision impulse computation to verify that this is the amount of energy lost during the collision. If momentum conservation is not obvious in your code, you can check for it too.

Another assumption we used was that the collision normals are always perpendicular to the surface of both objects. To assert that this condition is fulfilled, execute the following steps for the sphere-box collision:

- Check if all points of object 1 lie in or below the plane that is perpendicular to the collision normal and passes through collision point 1.
- Check if all points of object 2 lie in or above the plane that is perpendicular to the collision normal and passes through collision point 2.

Submission

Please submit a **compileable** snapshot of your (team-)work to Moodle along with a visual proof (e.g. set of pictures, short length video) to demonstrate the progress of your work.

Good luck,

Roland & Sebastian