Prof. Dr.-Ing. Ralf Steinmetz
Multimedia communications Lab

Dr. Florian Mehm
Dipl. Inf. Robert Konrad

TECHNISCHE UNIVERSITÄT DARMSTADT

„Game Technology“ Winter Semester 2014/2015

Exercise 8

For bonus points upload your solutions until **Friday the 12th of December 2014, 11:40**

General Information

- The exercises may be solved by teams of up to three people.
- The solutions have to be uploaded to the Git repositories assigned to the individual teams.
- **The submission date (for practical and theoretical tasks) is noted on top of each exercise sheet.**
- If you have questions about the exercises write a mail to game-technology@kom.tu-darmstadt.de or use the forum at https://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=557

1. Practical Tasks: Physics (5 Points)

In this exercise, the overall task is to build a demo in which you can shoot spheres from the camera position by pressing the space key and have them interact with each other and a plane. The code is provided for you, your task is to fill out the respective functions.

1.1 Particle Systems - Billboards

You can see in the exercise code that the particles are being spawned, but when the camera turns, they are visible from the side and back. Instead, we want them oriented towards the camera. Implement this rotation. The view matrix of the camera is available to the particle system.

1.2 Particle Systems – Control parameters

Decide on an effect you want to create using particle systems that uses one control parameter that is not present in the code you are given. For example, you can implement the "fire" example from the slides. Other control parameter could be the rotation of the particle billboards (you might want to change the texture in this case as the provided texture is symmetrical), the size of the billboards or the mass. For rain, you could spawn a "splash" particle when the rain hits the ground.

1.3 Numerical Integration

Implement an Euler integrator for the physics computations. The necessary function can be found in "PhysicsObject.cpp".

1.4 Sphere-Sphere Intersections

Implement an intersection test for two spheres with each other. This task entails implementing the functions
bool IntersectsWith(const SphereCollider& other);
vec3 GetCollisionNormal(const SphereCollider& other);
float PenetrationDepth(const SphereCollider& other);

which are found in "Collision.h"

2. Theoretical Tasks: Physics (5 Points)

2.1 Integration constant acceleration

Imagine a sphere with mass m=1kg that is falling down in a vacuum. At t=0, the sphere is released. The velocity v at t=0 is 0. The acceleration a is constant at 10 m/s² in the negative y-direction.

Calculate the movement of the sphere by integrating the equations of motion using the Euler method as explained in the lecture (you can use a spreadsheet application or a script).

a) Use a time step of deltaT=1s and provide the values of height (z) and velocity (v) for the situation at time t=3s.
b) Use a time step of deltaT = 0.5s and provide the values. Compare the results and discuss them.

2.2 Integration under drag

The sphere of task 2.2 is now dropped in the earth's atmosphere. Now, drag is acting on the object.

Drag is a force that acts on an object based on the velocity of the object. The faster the object, the more drag it experiences. We can approximate drag with the following formula:

$F_{drag} = -v_{normalized} (k_1 |v| + k_2|v|^2)$

The result is a force that acts in the opposite direction of the object's movement (indicated by $-v_{normalized}$, the normal vector in the direction of movement). The amount of drag results from the coefficients k1 and k2. Choose k1 = k2 = 0.1.

Calculate the movement of the sphere by integrating the equations of motion using the Euler method as explained in the lecture (you can use a spreadsheet application or script). Make sure to add the acceleration caused by the drag in the formulas you use.

a) Use a time step of deltaT=1s and provide the values of height (z) and velocity (v) for the situation at time t=2s.
b) Use a time step of deltaT = 0.2s and provide the values. Compare the results and discuss them.

2.3 Billboard matrix

In the practical exercise, we implemented particle billboards that were set up to face the camera wherever it was positioned. Now, assume a billboard which is used to display a tree. This billboard should stay upright, i.e. it should only rotate around the global y-axis pointing upwards to face the camera.

Explain how the model matrix to align this billboard in this way can be derived from the camera's view matrix.