Homework 3
Particle physics simulation

Advanced computer graphics 2021/22

1 Introduction

The goal of this homework is to get familiar with particle dynamics in physically based animation. Your task is to implement a particle dynamics simulation and visualization. The world, forces, particle types and particle emitters will be supplied externally.

2 Input

The input to your simulation will be a text file, containing descriptions of the simulation space, forces that act on the particles, particle types, and particle emitters. In the input file, distances and locations are given in meters, masses are given in kilograms, forces are given in Newtons, velocities are given in meters per second, and accelerations are given in meters per second squared. A line in the input file can define one of the following:

1. **Simulation space.** In this homework, the simulation space will be bounded by an axis-aligned bounding box, given by two of its opposite corners.
2. **Force.** A force can be one of three types:
   (a) **wind** – a constant force with a specified direction and strength (together given as a vector).
   (b) **radial** – an attractive/repulsive force created by a point in space, whose magnitude follows the inverse-square law.
   (c) **gravity** – a force that always points downwards (in the negative y direction) and, if not specified, corresponds to an acceleration of $9.81\,\mathrm{m/s^2}$.
3. **Particle type.** The particles in this homework will be balls with mass. The particle type defines the mass (or mass range) and radius (or radius range) of a particle.
4. **Particle emitter.** The homework without optional extensions only specifies a point emitter, which emits particles from a single point in space uniformly in all directions with a given rate (in units of inverse time). The emission should be implemented as a Poisson process. The type of each emitted particle is chosen uniformly from a given set of particle types. Every emitter also has a limit on the number of particles emitted. When it reaches that number, it stops emitting.

If a line cannot be parsed in one of the above forms, it should be ignored.

Here is an example input file:

```
# Simulation space (the size of the environment we want to simulate)
# space minX minY minZ maxX maxY maxZ
space -1024 -1024 -1024 1024.5 1024 1024

# Forces:
# - wind (x, y, z components in newtons):
# wind x y z
# - radial (x, y, z position, strength in newtons,
#   positive is attractive, negative is repulsive):
# radial x y z strength
# - gravity (acceleration in m/s^2):
# gravity [acceleration]
wind -5.4 7 10
```

```
radial -101 102 -111.1 -11.3
gravity

# Particle types (mass range, radius range):
# type (minMass [maxMass]) (minRadius [maxRadius])
type (0.1) (1.0)
type (0.12) (1.1 3.13)
type (0.1 0.12) (1.1 3.14)
type (0.23) (1.0 3.0) (0.1)

# Particle emitter (location in space, initial velocity range,
#     emission rate, particle type index (starting with 0)):
# emitter point (x y z) (minVx minVy minVz [maxVx maxVy maxVz])
#     (rate) (id [id2 id3 ...]) (maxParticleCount)
emitter point (100 100.1 200) (0 0 0) (100) (0) (10000)
emitter point (100 100.2 200) (-1 -3 -2 3 2 1.3) (103) (2) (11123)
emitter point (100 100.3 200) (-1 -2 -1 1.4 1 1) (112) (0 1 3) (111123)
```

3 Implementation

3.1 Particle dynamics simulation

Implement a particle dynamics solver that takes the simulation definition from the input file as defined in section 2 and simulates the creation and movement of particles. The simulation should run in real time. The input file defines the initial state of the system. The dynamics solver can be anything, including a simple Euler integrator.

Optional: use a higher-order integrator, such as the 4th order Runge-Kutta method.

3.2 Boundary collisions

Implement particle collisions with the simulation space boundaries. The particles must bounce from the boundaries as if they would hit the wall. The bounces should be completely elastic.

3.3 Visualization

Implement a simple visualization of the simulation.

3.4 *Optional: collisions between particles*

Implement collisions between particles. The collisions between particles should also be completely elastic. Take into account momentum interchange between the colliding particles. Optimise the collision detection with an acceleration structure such as a regular grid, octree or k-d tree.

3.5 *Optional: area emitters*

Add area emitters to the simulation. An area emitter chooses the initial position of the particle uniformly from an axis-aligned bounding box. The initial velocity of the particles is 0. The input file may accept the following description of an area emitter:

```
# emitter area (minX minY minZ maxX maxY maxZ) (rate) (id [id2 id3 ...]) (maxParticleCount)
emitter area (-10 -15.5 -20 10 11.1 12.3) (100) (0 1) (10000)
```

3.6 *Optional: drag*

Add drag to the particles. The force of the drag is $\frac{1}{2}\rho c v^2 A$*, where* ρ *is the air density equal to* $1.23\,\mathrm{kg/m^3}$ *v is the velocity of the particle, A is the cross-sectional area of the particle in* m^2*, and c is the unitless drag coefficient. The input file may accept the following description of a particle type:*

```
# type (minMass [maxMass]) (minRadius [maxRadius]) (dragCoefficient)
type (0.1 0.2) (3 4) (1.1)
```

The existing type definitions from the base instructions should still work. That is, if the drag coefficient is not specified, it is assumed to be 0.

4 Outputs

The expected output of this homework is a real-time simulation and visualization of a particle system.

5 Grading

This assignment is worth 10 points:

- 5 points for the dynamics simulation and boundary collisions,
- 3 points for the forces, and
- 2 points for the emitter.