Simulation in Computer Graphics - Exercises

Computer Graphics - Computer Science Department - University of Freiburg

Space Subdivision

In this exercise, we implement a simple particle simulation with inter-particle repulsion forces. To detect the neighborhood of a particle, we employ a space subdivision technique.

Many particle simulation methods need to compute inter-particle forces, i.e. forces that act between pairs of particles. Usually these forces are repulsing, they prevent that particles get too close and overlap. Examples for such particle simulations include SPH methods in fluid dynamics or crowd simulations. In order to compute a force between a pair of particles, we must determine the neighbors of a particle p_i, i.e. the particles p_j, $j = 1 \ldots n$ whose distance to p_i is smaller than a given threshold $d : \{p_j : |p_j - p_i| < d\}$. The most simple way to determine these neighbors was to test all n^2 pairs of particles.

A more efficient way is to partition the world space into hexagonal cells. The cells are collected in a linear array of size $N_x \times N_y \times N_z$, where N_x, N_y and N_z are the number of cells in x-, y-, and z-direction. Each cell contains a list of all particles located in that cell. In a first step, we compute for each particle the coordinates of the cell in which the particle is located. We then add the particle to the cells particle list. To determine the neighborhood of a particle, we determine the cell this particle is located in, and then we can simply traverse the cells particle list. To ensure that all neighboring particles are detected, also the neighboring cells in the so-called 3-cube (i.e. all $3 \times 3 \times 3$ cells around the particle) must be traversed.

- Implement the function `getCell` which returns the coordinates (i, j, k) of the cubical cell in which the particle particle is located. Consider that the cell is a cube with edge length `CELL_SIZE`. Moreover, try to avoid negative cell indices.
- Implement the function `getCellID` which computes the unique cell identification number from a cell coordinate triplet (i, j, k).
- Implement the function `addSingleParticleForce` which computes the inter-particle forces for a pair of particles. The minimum distance d between two particles is the sum of their radii (`particle1.radius + particle2.radius`). The force results from a spring with resting length d and stiffness `PARTICLE_FORCE_CONSTANT`.
- Implement the function `addParticleForces` which determines the neighborhood of the particle and calls `addSingleParticleForce` for each particle in the neighborhood.
- Experiment with different cell sizes `CELL_SIZE`.