EFFICIENT DATA STRUCTURES FOR SPH NEIGHBORHOOD SEARCH

Matthias Teschner

University of Freiburg, Computer Science Department, 79110 Freiburg, Germany
e-mail: teschner@informatik.uni-freiburg.de, web page: http://cg.informatik.uni-freiburg.de/

The computation time in SPH fluid simulation is often dominated by enforcing incompressibility and by the search for particle neighbors. On desktop PCs, we currently compute scenarios with up to 100 million SPH particles, corresponding to about 3 billion neighboring particles that have to be found in each simulation step. This abstract discusses spatial data structures for the efficient neighborhood search in SPH. A result is outlined in Fig. 1.

In the context of SPH, there seems to be an agreement that hierarchical spatial data structure, e.g. kd-trees, are less efficient, while uniform grids have certain advantages. Uniform grids can be efficiently built and queried. Grid cells and, even more importantly, adjacent grid cells can directly be accessed without traversing a hierarchy. And there exist an optimal cell size corresponding to the influence radius of a particle if all particles have the same influence radius.

Uniform grids can be implemented in various ways. The fastest implementation would certainly be the explicit representation of the entire simulation domain in a three-dimensional array of grid cells. This, however, is prohibitive for larger scenarios where the simulation domain might be non-uniformly and only sparsely filled with particles. It might be further expensive to adapt the data structure for dynamically changing domains. To address these issues, spatial hashing, index sort and various variants of these two concepts have been proposed [1,2].

Spatial hashing maps grid cells from an effectively infinite simulation domain to a finite list. In order to minimize hash collisions, some variants, e.g. compact hashing, use rather large hash tables which only process handles to a second, compact list. The second list is motivated by the fact that its traversal is significantly faster than the traversal of the sparsely filled hash table. Further improvements are concerned with spatial locality, e.g. z-curves and separate processing of particle references and satellite data.

Index sort is a second promising implementation of uniform grids. For each particle, a key is computed from its grid cell. Then, all particles are sorted with respect to this key. Thus, particles in the same grid cell are adjacent to each other in the sorted list. References to the location of the first particle in the sorted list are stored per grid cell. For sorting, radix sort is commonly employed, but even insertion sort is an interesting alternative as the list does not change much between simulation steps. Z-curves can be employed as well.

Figure 1: The neighbourhood search with compact hashing takes 22s on a 16-core PC in this breaking dam scenario with 107 million fluid particles.

REFERENCES

[1] M. IHMSEN, N. AKINCI, M. BECKER, M. TESCHNER: A Parallel SPH Implementation on Multi-core CPUs. *Computer Graphics Forum*, vol. 30, no. 1, pp. 99-112, 2011.

[2] M. IHMSEN, J. ORTHMANN, B. SOLENTHALER, A. KOLB, M. TESCHNER: SPH Fluids in Computer Graphics. *Eurographics 2014 - State of the Art Reports*, pp. 21-42, 2014.