Computer Graphics 2 - Exercise Sheet 1

Technische Universität Berlin - Computer Graphics

Handout 20. April 2023, **Due** 9. May 2022

Maximilian Kohlbrenner, Tobias Djuren, Dimitris Bogiokas, Prof. Dr. Marc Alexa

In this assignment you will create a framework and an important data structure for the following programming assignments. First, you need to download a zip archive containing point cloud files from the CG2 ISIS page.

We provide a C++ sample with the basic setup of a `polyscope` viewer in a second zip archive. **Using the given example code is optional.** You are free to create your own framework from scratch, which will be completed throughout the semester. You can use other programming languages than C++, for example Python, JavaScript or Julia. But keep in mind that the data structure needs to perform fast as we will later use it for thousands of queries.

Exercise 1: Spatial Data Structures (5 + 0.5 Points)

In this assignment, point data should be read from a file and a suitable spatial data structure should be constructed. This data structure should support efficient (sub-linear) searches. Data points and query results should be visualized on the screen.

1. Parse the point data from OFF-files and store them. You can find the specification of OFF files under the link `http://paulbourke.net/dataformats/oogl/#OFF`. (1 Point)
2. The loaded data points should be stored in a suitable spatial data structure (your choice). The data structure should be efficient (sub-linear) for the requests. (1 Point) If you implement a k-d tree, perform the median search in (asymptotic) linear time on average. (0.5 Bonus Points)
3. The query functions `collectKNearest(Point p, int knearest)` and `collectInRadius(Point p, float radius)` should be implemented. Write a test routine to demonstrate the **sub-linear** runtime. (2 Points)
4. Implement the visualization of the data points, the spatial data structures as well as the search results. (1 Point)

Examples for the visualization of the data structures and the search results are visible in Figure 1.

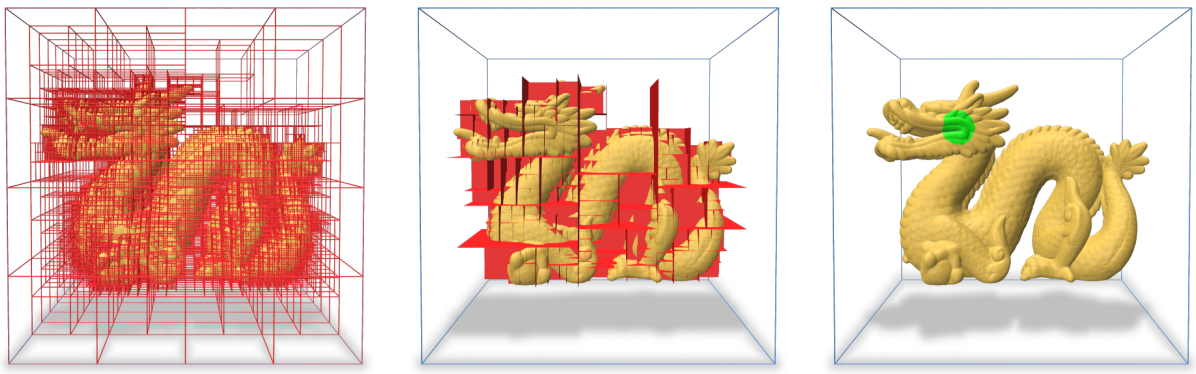

Figure 1: Visualization of an octree (left), k-d tree (mid) and `collectInRadius` (right).

Exercise 2: Theory (3 Points)

1. Describe an algorithm to find the median of n numbers in (asymptotic) linear time on average. Is it possible to guarantee linear time even for the worst case? (1 Point)

   Hint: Median search is needed for example in the k-d tree construction.

2. Given are n points in space, which have a pairwise distance of $\geq \epsilon$. Constructing an Octree starting with a cubic cell of edge length s (this cell contains all n points), what is the maximal depth of this Octree and why? (1 Point)

3. Is there a configuration of data points $\mathbf{p}_i$ and a query point $\mathbf{q}$ such that a k-nearest neighbor query has linear runtime for *all* spatial data structures presented in the lecture? The number of data points $N > 0$ must be arbitrary and all points should have non-zero pairwise distance, *i.e.*, $\|\mathbf{p}_i - \mathbf{p}_j\| > 0$, for $i \neq j$. (1 Point)