Chapter 1

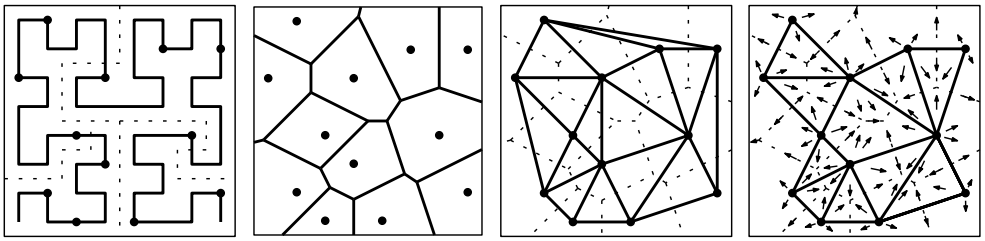
Introduction

Point sets occur in many applications, for instance as positions in space or in space and time, and as building blocks of more complex geometric objects such as curves and surfaces. For example, most digital data generated today includes spatial information, e.g., a location in space [97]. In recent animated movies hundreds of millions of points are used in the rendering process of a complex scene [36].

More generally, points are used in most geometric algorithms. Examples of classical application areas where geometric algorithms are used are computer graphics and imaging, shape reconstruction, computer vision, geographical information systems, mesh generation, robotics, manufacturing and product design, molecular biology, astrophysics [33]. More recently, geometric aspects of machine learning [129] and geometric algorithms on massive data sets [1, 7] have received increasing attention. New applications of geometric algorithms appear in emerging research areas such as sensor and ad-hoc networks [92, 149].

In geometric problems often large numbers of points need to be stored and processed. For analyzing and efficiently handling large point sets, these need to be organized. An efficient storage access depends on the data layout, i.e., on how the data is organized on memory. Organizing point sets in a geometric data structure may reveal spatial relations of the points, in particular proximity relations. Furthermore, point sets need to be organized as input to applications, for further geometric processing, and analysis.

One possibility of organizing a point set is using space-filling curves [126]. For instance for accessing large data bases, a data layout based on space-filling curves can be used to guarantee good geometric locality [88]. Consider for example the schematic layout of a database in Figure 1.1(a). The stored data is shown with their positions in space. The data is stored along a space-filling curve. The space-filling curve is indicated in the figure by the solid line. The dashed line indicates which points are grouped into one page. The advantage of this data layout is that queries on the data are expected to need to access only a few number of pages. Furthermore, this data layout adapts well to update. A property of space-filling curves, which is important for the data layout, is that points which are close on the curve are also close in space. This property makes ordering the points along a space-filling curve applicable as heuristic for



(a) Space-filling curve order (b) Voronoi diagram (c) Delaunay triangulation (d) Flow complex

Figure 1.1: Geometric structures on point sets.

the traveling salesperson problem [122], i.e., the problem of finding the shortest round-trip through a set of points. Further applications of space-filling curves include mesh indexing [111] and processing raster data, e.g., images [147] and terrains [94].

A geometric structure reflecting proximity relations of a point set is the Voronoi diagram. Given a set of points in the plane, its Voronoi diagram is a partioning of the plane into cells such that all points in a cell have the same closest neighbor in the point set (see Figure 1.1(b)). This concept can be illustrated by the post office problem. Given the locations of several post offices, find the closest post office to a given position. The post office problem can be solved efficiently for several queries by computing the Voronoi diagram of the post office locations and determining the cell of the Voronoi diagram in which a query position lies. The concept of a Voronoi diagram has independently emerged in several sciences, for instance in biology, physiology, chemistry, physics, crystallography, metereology, geography, and mathematics [8]. A list of applications from A (anthropology, archeology, astronomy) to Z (zoology) is given in [56].

In Figure 1.1(c) points with adjacent Voronoi cells are connected by an edge. The resulting tessellation of the space is called Delaunay tessellation or Delaunay triangulation. The Delaunay tessellation is an efficient way of representing the combinatorial structure of the Voronoi diagram. Furthermore the Delaunay tessellation has many useful properties.

A structure revealing topological properties of a point set is the flow complex. The flow complex can be described as follows. Each point in the plane flows away from its closest neighbors in the point set. This induces a partition of the plane where a cell of the partition consists of all points that flow into the same point. The partition is shown in Figure 1.1(d) with the direction of flow indicated by arrows. Organizing a point set by computing its flow complex is a first step for further analysis and processing of the point set. The flow complex has been applied in surface reconstruction, shape segmentation, image processing and molecular biology.

In this thesis, we are interested in the computation of the structures described above: orders along space-filling curves, Delaunay tessellations, and flow complexes. The Voronoi diagram can be easily computed from the Delaunay tessellation. We will use space-filling curve orders for constructing Delaunay tessellations and use Delaunay tessellations for constructing flow complexes.

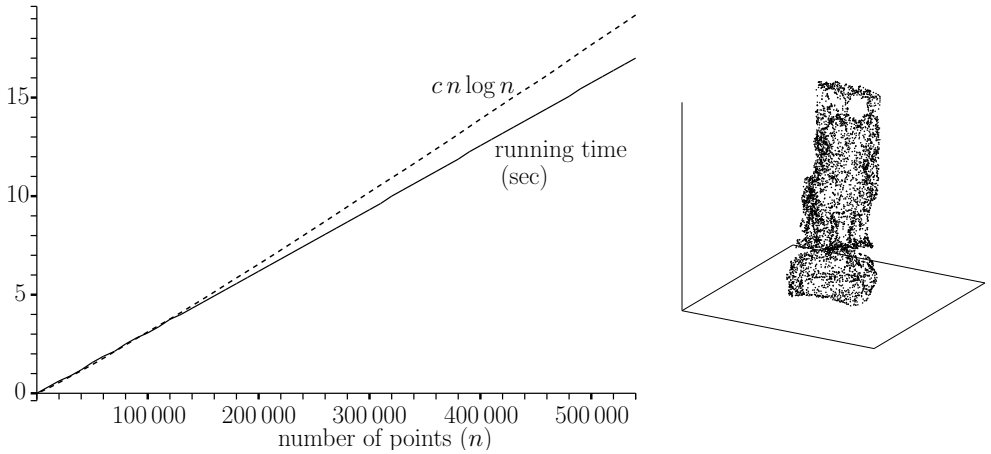


Figure 1.2: Experimental running time of constructing a Delaunay tessellation along space-filling curve orders.

For space-filling curve orders and Delaunay tessellations we will focus on theoretically analyzing the running time of algorithms. Commonly, the running time of an algorithm is measured by a worst-case analysis. That is, an upper bound on the running time depending on the number of points is determined for the worst-possible point distribution. Even if the complexity of the Delaunay tessellation of a point set is linear, as it is in two dimensions, the running time of any algorithm for constructing it using standard operations will be of order at least $n \log n$ in the worst case, where n is the number of points. Figure 1.2 shows the graph of the function $x \log x$ as a dashed line. In comparison, Figure 1.2 also shows the graph of the running time in experiments of a Delaunay tessellation construction algorithm using space-filling curves for surface data (a small sample of the surface data is also shown in the figure). The graph suggests a linear running time. Besides a good performance in practice, theoretical guarantees on its performance are of interest. However, a worst-case analysis is not suitable to explain the linear running time in experiments.

An alternative to a worst-case analysis is an average-case analysis. In an average-case analysis, the expected running time of the algorithm is determined based on the distribution of the input points. For computing space-filling curve orders and Delaunay tessellations (with space-filling curve orders), we will consider both worst-case and average-case analyses but with an emphasis on the average-case.

For the flow complex we will not be interested in the running times but simply in an algorithm for computing it in arbitrary dimension. Previously the flow complex has been studied only in two and three dimensions. For these cases the known algorithms rely on the special structure of the flow complex in these dimensions. We consider the fundamental geometric structure of the flow complex and how it can be used to design an algorithm for computing it. In turn this results in a deeper understanding of the topological structure of the flow complex.

Overview of results and structure of the thesis.

Chapter 2:

- Space-filling curves and space-filling curve orders of point sets are introduced.
- Analysis of fast space-filling curve order computation in particular in the average-case.
- For points on a grid, optimization of discrete space-filling curve orders depending on the measure of locality.

Chapter 3:

- The Voronoi diagram, Delaunay triangulation, and incremental construction are introduced.
- Worst-case analysis of incremental constructions con BRIO in the general framework of configuration spaces.
- Algorithm for constructing Delaunay triangulation using space-filling curves.
- Average-case analysis for uniform points in the plane.

Chapter 4:

- Average-case analysis using a different counting scheme than in Chapter 3.
- Analysis is applied to uniform points in higher dimensions and normally distributed points in the plane.
- Experimental evaluation of the algorithm on various point distributions.
- Analysis of a further algorithm for constructing the Delaunay tessellation based on nearest neighbor search in higher dimensions.

Chapter 5:

- The flow complex is introduced.
- Analysis of properties of the flow complex.
- A general algorithm for computing the flow complex in any dimension.

Chapter 6:

- Flow shapes are introduced.
- Proof of the homotopy equivalence of union of balls and flow shapes.