Chapter 1

Introduction

Geometric pattern matching deals with important practical as well as interesting theoretical problems. The basic pattern matching problems read as follows:

Problem 1.1 (δ-measure problem for Π).
Given *two patterns P, Q from a set of* valid patterns Π*, and a* distance measure $\delta : \Pi \times \Pi \to \mathbb{R}$ *on* Π.
Compute *the distance $\delta(P, Q)$.*

Problem 1.2 (Equivalence problem for Π with respect to $\mathcal{T}$).
Given *two patterns P, Q from a set of* valid patterns Π*, and a set of* admissible transformations $\mathcal{T}$ *of* Π.
Decide, *whether there is a transformation $\tau \in \mathcal{T}$ such that $\tau(P) = Q$.*

Problem 1.3 (δ-matching problem for Π with respect to $\mathcal{T}$).
Given *two patterns P, Q from a set of* valid patterns Π*, a* distance measure $\delta : \Pi \times \Pi \to \mathbb{R}$ *on* Π*, and a set of* admissible transformations $\mathcal{T}$ *of* Π.
Find *a transformation $\tau \in \mathcal{T}$ such that $\delta(\tau(P), Q)$ is as small as possible.*

Needless to say that these problems and their variants have numerous applications in a wide variety of scientific disciplines like character recognition, geographical information systems, computer aided design, to name just a few.

In the field of *geometric* pattern matching, the patterns Π are modelled by 'simple' geometric objects like point sets, sets of line segments, polygons, or polyhedral surfaces, in d-dimensional Euclidean space $\mathbb{R}^d$ for some $d > 0$, with the cases $d = 2, 3$ being the most prominent choices here (for obvious reasons). The distance measure that is used, like the Hausdorff distance δ_H (c.f. Definition 1.4 on page 7) or the Fréchet distance δ_F (c.f. Definition 3.10 on page 23), is usually a metric on Π (or on a larger class of sets) and the admissible transformations $\mathcal{T}$ are 'natural' transformations of the underlying space $\mathbb{R}^d$ like translations, Euclidean motions, or affine mappings. This provides a mathematically sound and rigorous foundation for the pattern matching problem — something that many 'classical' approaches like, e.g., neural nets, syntactic matching, or feature based techniques

fail to achieve. Moreover it is possible to apply the powerful machinery and the techniques from computational geometry; as it turns out this is indeed a very good approach — the field has developed interesting connections to a variety of mathematical subjects and produced nice algorithmic as well as mathematical results over the last years. The survey article of Alt and Guibas [15] provides an excellent and extensive overview; the interested reader is referred to this paper and the references therein.

We should note that in order to apply results from geometric pattern matching to problems that arise in 'real world applications', where one has to handle, e.g., pixel images from a digital camera, point clouds from a laser scanner, or volumetric data from a magnetic resonance scanner, it is necessary to convert the input data to the abstract mathematical objects that are used to model the problem, in a 'reasonable' manner. This gives rise to various interesting problems and questions and constitutes an active field of research on its own. However, we will not look into this preprocessing step in this thesis; instead we concentrate on the geometric pattern matching problem from a purely theoretical point of view and try to develop efficient algorithms for various incarnations of it.

Preliminaries The model of computation we adopt in this thesis is the *real RAM* as proposed by [43]. It constitutes a suitable adaption of a classical machine model commonly used in algorithmics (the so-called *random-access machine* — RAM for short — described, e.g., in [5]) to the field of computational geometry. The main difference is that each register of the machine can hold a single *real* number, and that arithmetic operations, comparisons, as well as the application of analytic functions (like, e.g., k-th roots, trigonometric functions, exponentiation, or logarithms) are all available at unit cost.

Some of our algorithms are *randomized*; thus we assume that the machine provides access to a random bit generator. All randomized algorithms in this thesis are of the *Las Vegas* type, i.e., they always return the correct result, but their runtime is a random variable; when we specify the runtime of a randomized algorithm as a function of the input size only, we mean the expectation of that random variable.

Throughout this thesis the $\mathcal{O}_\epsilon$-notation will be employed to emphasize that the constants involved may depend on a parameter $\epsilon > 0$[a]. A statement of the form 'We can compute ... in $\mathcal{O}_\epsilon(T(n,\epsilon))$ time.' actually means 'We can compute ... in $\mathcal{O}_\epsilon(T(n,\epsilon))$ time, *for any* $\epsilon > 0$.'.

Most of the major definitions are contained in the introductory sections of the individual parts. Some less important ones are provided only when needed and are dispersed throughout the text and footnotes. For easier reference the appendix contains an index and a glossary that repeats most of the crucial definitions. The reader is referred to – by now – standard textbooks about computational geometry like [43], [32] for reference to basic definitions, notions and techniques from that field.

[a] To be more precise, we have that $T(n) = \mathcal{O}_\epsilon(f(n,\epsilon))$ iff there is a function $C(\epsilon)$, and for any $\epsilon > 0$ we have that for all n, $T(n) \leq C(\epsilon) \cdot f(n,\epsilon)$ holds.

Overview This thesis is roughly organized as follows: It consists of three parts that (essentially) can be read independently of each other. Each of the parts deals with pattern matching problems for different kinds of geometric objects. More specifically, patterns are modelled with *finite sets of points* in Part I, with *plane polygonal curves* in Part II, and with *polyhedral surfaces* in Part III. Throughout this work P and Q will denote sets of m and n such 'simple' geometric objects.

The introduction to each of the parts gives a short problem description and motivation and provides the necessary definitions. A survey of previous results on the problem under consideration follows, and finally a summary of our own results is given.

Our main contributions are the following (we omit all necessary definitions and refer to the introductory sections of the individual parts):

- In chapter 2 we present an algorithm for the d-dimensional congruence test problem of finite point patterns that runs in $O(n^{\lceil d/3 \rceil} \log n)$ time (c.f. Theorem 2.2 on page 13).

- In chapter 3 we present efficient algorithms to measure the one-sided Hausdorff distance of a d-dimensional m-point set P to a set Q of n geometric objects of constant 'size' each. To be more precise we look at the case where Q is a set of n *semialgebraic* sets in $\mathbb{R}^d$, each of constant *description complexity*. We develop an algorithm to compute $\tilde{\delta}_H(P, Q)$ in $\mathcal{O}_\epsilon(mn^\epsilon \log m + m^{1+\epsilon-\frac{1}{2d-2}} n)$ randomized time (c.f. Theorem 3.8 on page 20).

- In chapter 4 we present exact and approximation algorithms to solve the δ_F-matching problem for polygonal curves with respect to translations. To be more precise, we describe an algorithm that solves the corresponding decision problem in $\mathcal{O}\big((mn)^3(m+n)^2\big)$ time when we consider the Fréchet distance and in $\mathcal{O}\big((mn)^3\big)$ when we look at the weak Fréchet distance (c.f. Theorem 4.23 on page 37 and Theorem 4.24 on page 38). We complement the exact solution with an $\mathcal{O}(\epsilon^{-2}mn)$ time approximation algorithm that yields a Fréchet distance which differs from the optimum value by a factor of $(1+\epsilon)$ only (c.f. Theorem 4.29 on page 40). To this end we describe reference points for the Fréchet distance, and use them to obtain the aforementioned approximation algorithms. We conclude with a negative result that shows that no such reference points for *affine* maps exist (c.f. Theorem 4.31 on page 41).

- In chapter 5 we show that for a certain class of curves — the so called κ-*straight* curves — there is a close relationship between the Fréchet and the Hausdorff distance (c.f. Theorem 5.3 on page 44). The parameter $\kappa \geq 1$ – roughly speaking – measures how much these curves 'resemble' a straight line. This result gives rise to a randomized approximation algorithm that computes an upper bound on the Fréchet distance between two such curves that is off from the exact value by a multiplicative factor of $(\kappa+1)$. The algorithm runs in $\mathcal{O}((m+n)\log^2(m+n)2^{\alpha(m+n)})$ time (c.f. Theorem 5.6 on page 46 and Corollary 5.8 on page 49). We also provide an $\mathcal{O}(n \log^2 n)$ time algorithm to decide for any $\kappa \geq 1$, if a given polygonal curve on n vertices is κ-straight (c.f. Theorem 5.9 on page 49).

- In partIII we develop efficient algorithms for the δ_H-measure problem for simple polyhedral surfaces in $\mathbb{R}^3$. We give an algorithm that computes $\delta_H(P,Q)$ in $\mathcal{O}_\epsilon((mn)^{15/16+\epsilon}(m^{17/16}+n^{17/16}))$ randomized time (c.f. Theorem 6.19 on page 68). For variants of the Hausdorff distance, which we get when using a polyhedral distance function, we obtain an algorithm that solves the decision problem in $\mathcal{O}_\epsilon((m+n)^{2+\epsilon}+mn(\sqrt{m^{1+\epsilon}}+\sqrt{n^{1+\epsilon}}))$ time (c.f. Theorem 6.12 on page 66).

All of these results improve upon earlier approaches to the problems under consideration; some of them even constitute the first non-trivial algorithmic solution. Most of the results presented in this thesis have been published in [24], [16], [17], [1], and [11].

Acknowledgements I would like to thank a lot of people. The following individuals (in alphabetical order) contributed – in one way or the other – to this thesis: Pankaj Kumar Agarwal, Helmut Alt, Peter Braß, Herbert Edelsbrunner, Stefan Felsner, Frank Hoffmann, Rolf Klein, Klaus Kriegel, Günter Rote, Micha Sharir, Carola Wenk. First of all I have to thank Helmut Alt for the obvious things: he was an excellent advisor and very often much more than that. I am very grateful to Peter Braß and Günter Rote for sharing some of their insights with me and for asking the right questions at the right time. Thanks also go to my coauthors from which I learned so much: Pankaj Kumar Agarwal, Helmut Alt, Peter Braß, Alon Efrat, Frank Hoffmann, Rolf Klein, Klaus Kriegel, Günter Rote, Micha Sharir, and Carola Wenk. My colleagues from the work group 'Theoretical Computer Science' at the Freie Universität Berlin deserve credit for providing a constant source of inspiration and motivation, for patiently answering my stupid questions, and for standing my weird personality for so long. Special thanks go to Helmut Alt and Stefan Felsner for reading earlier drafts of this thesis. Last but definitely not least I want to thank my family for their love and their support.