Geometric matching of weighted point sets

Geometrisch Vergelijken van Verzamelingen van Gewogen Punten
(met een samenvatting in het Nederlands)

PROEFSCHRIFT

ter verkrijging van de graad van
doctor aan de Universiteit Utrecht
op gezag van Rector Magnificus, Prof. Dr. W.H. Gispen,
ingevolge het besluit van het College voor Promoties
in het openbaar te verdedigen
op woensdag 21 september 2005 des middags te 12:45 uur

door

Panagiotis Giannopoulos

geboren op 1 maart 1973,
te Athene

Promotor: Prof. Dr. Mark H. Overmars
Co-promotor: Dr. Remco C. Veltkamp
Faculteit Wiskunde en Informatica
Universiteit Utrecht

Advanced School for Computing and Imaging

This work was carried out in the ASCI graduate School.
ASCI dissertation series number 115.

This research has been partially supported by the Dutch Technology Foundation STW, project no. UIF 5055, and a Marie Curie Fellowship of the European Community programme "Combinatorics Geometry and Computation", FU Berlin, under contract number HPMT-CT-2001-00282.

ISBN 90-393-4021-8

Preface

This thesis is about geometric pattern matching. In particular, it deals with questions of the following type:

> Given two patterns, determine how similar they are, according to some distance measure between the patterns.

An introduction to the field of geometric pattern matching is given in Chapter 1. The next two chapters deal with transportation distances for weighted point sets. The results in Chapter 2 have been previously published in Giannopoulos and Veltkamp [65] and Typke et al. [129], while a preliminary version of Chapter 3 has been published in Cabello et al. [32].

Chapters 4 and 5 deal with the problem of maximizing the area of overlap of two unions of disks under translations and rigid motions; the results have already appeared in a technical report by de Berg et al. [52] and published in de Berg et al. [50].

Being a PhD student, almost always entails being involved, to some extend, in exciting research problems not directly connected to the main thesis' topic. One such problem I came across, concerns geometric spanner graphs and minimizing their dilation by adding one or more edges (shortcuts). The results of this research are not included in this thesis; the interested reader can find them in Farshi et al. [89].

This research has been partially supported by the Dutch Technology Foundation STW, project "Shape Matching Environment", no. UIF 5055. The two main objectives of this project were the following: first, the development of algorithms for complex shape matching problems, including matching point sets with accuracy weights, collections of curves and regions, articulated objects; second, the development of a state-of-the-art software shape matching environment of currently available algorithms. The results presented here correspond to the first part of the project.

Acknowledgements. First, I wish to thank my promotor Mark Overmars and my supervisor Remco Veltkamp for giving me the chance to work in the GIVE group in Utrecht and for always being very supportive and understanding even when my research interests diverted from theirs. Remco's experience, ideas and guidance have been of major help in the first few years of the PhD's labyrinth.

I am grateful to Helmut Alt at Freie Universität, Berlin, for accepting me as a guest in his group in the fall of 2002 and for giving me the chance to participate in the graduate programme CGC during that time. But, above all, I thank him for letting me come back to Berlin for the most difficult and final year of my research. Doing research with Helmut proved to be serious fun.

I would also like to thank my co-authors: Mark de Berg, Sergio Cabello, Günter Rote, Christian Knauer, René van Oostrum, Rainer Typke, Frans Wiering and, of course, Remco Veltkamp. My special thanks to Joachim Gudmundsson for introducing me to geometric graphs and related problems. Thanks also goes to the members of the reading committee: Prof. Helmut Alt, Prof. Mark de Berg, Prof. Frans C.A. Groen, Prof. Stefan Schirra and Prof. Gert Vegter, for reviewing this thesis and providing useful comments.

There are quite a few friends of mine who deserve my deepest thanks: Andrea, Anne Marie and Fientje, Anke, Ares, Avi, Bill, Betta, Despoina, Giuseppe, Giwrgos, Marleen and Stelios, your help is so much appreciated.

This thesis is dedicated to my family for their unlimited love and support.

P.G.
Berlin, Utrecht,
July 2005

Contents

1 Introduction **9**
- 1.1 Shape . . . 9
- 1.2 Applications . . . 10
- 1.3 Approaches to shape matching . . . 11
  - 1.3.1 Similarity measure based shape matching . . . 12
- 1.4 Similarity measures for geometric patterns . . . 12
  - 1.4.1 Properties of similarity measures . . . 13
- 1.5 Shape matching problems using similarity measures . . . 16
  - 1.5.1 Finite point sets . . . 18
  - 1.5.2 Curves . . . 22
- 1.6 Overview of the thesis . . . 23

2 Weighted point set matching **27**
- 2.1 Introduction . . . 27
- 2.2 Fuzzy sets and the Hausdorff distance . . . 28
- 2.3 The mass transportation problem . . . 30
- 2.4 The Earth Movers Distance . . . 30
  - 2.4.1 Definition . . . 30
  - 2.4.2 Properties and Computation . . . 31
  - 2.4.3 Drawbacks of the EMD . . . 32
- 2.5 The Proportional Transportation Distance . . . 33
  - 2.5.1 Properties . . . 34
- 2.6 Lower and Upper Bounds . . . 35
  - 2.6.1 Upper Bounds on the EMD . . . 35
  - 2.6.2 Upper Bounds on the PTD . . . 36
  - 2.6.3 An example of the Lower and Upper Bounds' use . . . 37
- 2.7 Experimental Results . . . 38
  - 2.7.1 Logos . . . 39
  - 2.7.2 Fish Contours . . . 41
  - 2.7.3 Melodies . . . 46
  - 2.7.4 Polyhedral Models . . . 47
- 2.8 Concluding remarks . . . 48

3 Minimizing the Earth Mover's Distance under Transformations 51

3.1 Introduction 51
    3.1.1 Preliminaries 51
    3.1.2 Relevant work 52
    3.1.3 Results 53
3.2 Approximating the EMD 54
3.3 Lower bounds on the EMD 57
3.4 Approximation algorithms for translations 58
    3.4.1 Equal weight sets 60
    3.4.2 Partial assignment 61
3.5 Approximation algorithms for rigid motions 63
    3.5.1 Rotations 63
    3.5.2 Rigid motions 68
3.6 Concluding remarks 71

4 Maximizing the Area of Overlap of two Unions of Convex Objects under Translations 73

4.1 Introduction 73
    4.1.1 Applications 74
    4.1.2 Relevant work 74
    4.1.3 Results 75
4.2 Preliminaries 76
4.3 Complexity of the function of the area of overlap 77
    4.3.1 Sets of convex homothets 78
    4.3.2 Sets of fat convex objects 80
    4.3.3 A brute force approach to computing the optimum translation 81
    4.3.4 Disjoint unit disks 81
4.4 A lower bound on the area of overlap 84
4.5 Constant factor approximation schemes 89
4.6 An $(1-\epsilon)$-approximation algorithm 93
4.7 Concluding remarks 96

5 Maximizing the area of overlap of two unions of disks under rigid motions 99

5.1 Introduction 99
    5.1.1 Results 99
5.2 Basic properties of the overlap function 100
5.3 The rotational case 102
5.4 A $(1-\epsilon)$-approximation algorithm for rigid motions 106
    5.4.1 An improvement for sets with small diameter 108
5.5 A Monte Carlo algorithm 109
5.6 Sets of intersecting disks with different radii 112

6 Further research 115

Bibliography **117**

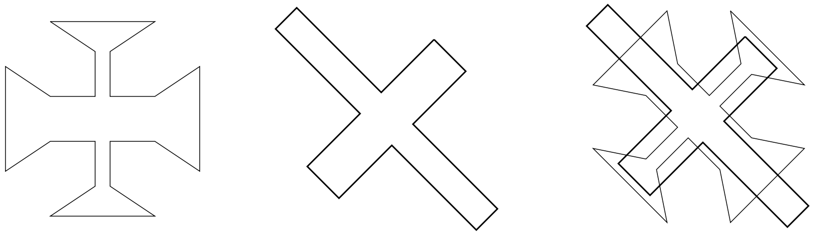
Samenvatting **127**

Curriculum Vitae **129**

Chapter 1

Introduction

Shape matching is a fundamental problem in computer vision, pattern recognition and robotics: given two shapes A, B, one would like to know how much they *resemble* each other. Usually, one of the shapes may undergo certain transformations like translations, rotations or scalings in order to be matched with the other as well as possible. Then the problem becomes that of finding a transformed version of A that is *similar* to B according to some criterion. An example of two shapes and a possible match between them is shown in Figure 1.1.



Pattern A Pattern B Match

Figure 1.1: Two patterns A and B and a possible match of a rotated and translated version of A to B.

1.1 Shape

Shape is a modality that can be found in abundance in our physical world. Indeed, every physical object around us can be attributed a 'shape'. However an exact definition of shape is elusive. Shape can be *formed* by color or intensity patterns from which a geometrical representation can be derived. Alternatively, "shape is limit of solid" [104].

Shape in mainly considered as a geometrical characteristic. Veltkamp [137] notes that, from the human visual perception point of view, shape can be considered as a single visual entity, which is different from a single *geometric* entity. For example, a finite point set is not a single geometric entity, but can well be perceived as a single visual entity, a shape. Apart from geometry, the notion of shape is also concerned with structure (connectivity), attributes (color, texture), semantics (functionality, features) and interaction with time (animation, deformation, morphing).

This thesis deals with geometric shapes, i.e. shapes defined by finite unions of geometric primitives such as points, straight line segments, curve segments, triangles and disks. In the literature, the term pattern is often used instead of shape. Formally, the *shape* of a pattern is the pattern under all transformations in a transformation group. Here, we avoid this formality and the terms pattern and shape are used synonymously.

In many applications, images are typical sources of geometric patterns. Normally, the image is subjected to a feature extraction process, such as corner detection, edge detection and region detection, whose details shall not concern us here. Feature extraction is an active field of research on itself, related to disciplines such as image processing and computer vision; see Jain et al. [84] and Bosdogianni and Petrou [103] for details.

1.2 Applications

The application domain of shape matching is vast and includes a range of different applications such as optical character recognition, medical image registration, molecular docking and content based image retrieval; some examples are given below.

Optical character recognition

Optical character recognition (OCR) is the task of automatically reading handwritten or printed text [49]. First, the text is scanned and then the resulting image is converted into an electronic document of recognized characters and words. The potential of OCR systems is enormous because they enable users to harness the power of computers to access printed documents. OCR is already being used widely in the legal profession, where searches that once required hours or days can now be accomplished in a few seconds. OCR is a highly specialized form of pattern matching. For example, special OCR methods exist for Chinese character recognition [78]. Most OCR algorithms are designed specifically for character recognition and can not be generalized to recognition of more complex shapes.

Medical image registration

Medical image registration is the process of the alignment of two medical images taken from different views or obtained using different input devices [90]. For example, in multi-modality matching, one image might be a CT (computer tomography) scan while the other is an MRI (magnetic resonance imaging) image.

Alignment aims at matching salient points between two or three-dimensional images that correspond to the same physical region of the scene being imaged. For medical purposes, accuracy in registration is critical and thus of major importance.

Molecular docking

Molecular docking is an important task in computational drug design [60]. A fundamental assumption for rational drug design is that drug activity is obtained through the molecular docking of one molecule (the ligand) to the pocket of another, usually larger, molecule (the receptor) which is commonly a protein. In their active, or binding conformations, the molecules exhibit geometric and chemical complementarity, both of which are essential for successful drug activity. Thus, most approaches take into consideration both the chemical and geometric characteristics of the molecules.

Given two interacting molecules of known geometry, the docking problem consists of finding their relative positions during the interaction. Recent methods adopt the approach of matching points (features) of the binding pocket to points (features) of the ligand. Such points can be defined by using energy calculations or the geometry of the receptor and the ligand or both. Once these points have been identified the docking problem reduces to point set matching.

Content based image retrieval

Today, large image databases are common place in multimedia applications. Content based image retrieval (CBIR) is the task of automatically retrieving images by their content instead of external features such as keywords [123]. A recent user survey about cognition aspects of image retrieval shows that users are more interested in geometry based (shape) retrieval than intensity-based (color and texture) retrieval [123]. Retrieving by shape is still considered one of the most difficult aspects of CBIR with all current commercial CBIR systems performing poorly with this respect. Thus there is an increasing demand for efficient and reliable shape matching methods.

1.3 Approaches to shape matching

Shape matching has been approached in a number of ways within the context of computer vision and pattern recognition. These include aspect graphs [67], the generalized Hough transform [22], geometric hashing [141], the alignment method [79], deformable templates [118], wavelet-based transforms [83], neural networks [68], and statistics [120]; for a survey see Hagedoorn and Veltkamp [72].

Global object methods work on an object as a whole, i.e. a complete object area or contour. Examples are Fourier descriptors [88], moments [38] curvature scale space [94] and shape contexts [24]. An important drawback of these methods is that they require complete object segmentation in images which is in itself an ill-posed problem [84]. In general, such methods are not robust against noise and occlusion and not appropriate for partial shape matching. Therefore, global methods are not well suited for measuring the similarity between two patterns that differ much.

Most of the methods mentioned above depend heavily on heuristics that give no theoretical guarantee on the quality of the matching they achieve, but perform well in practice. Some can be very simple and fast but they lack specific properties and thus generality. A different approach is similarity measure based shape matching explained below.

1.3.1 Similarity measure based shape matching

A similarity measure is a function that assigns a non-negative real number to each pair of patterns. Similarity measure based methods have several advantages over most global methods. They are independent of pattern representations as long as the used measure is defined on patterns as sets. Moreover, similarity based methods can be used in partial matching. A disadvantage of these methods is their running time which is often high and tends to increase with the complexity of the similarity measure.

Computational geometry

Computational geometry is the sub-area of algorithmic design that deals with the design and analysis of algorithms for problems involving geometric objects. The standard approach taken in computational geometry is the development of exact, provably correct and efficient solutions to geometric problems. For an introduction to the field, see the classical textbooks by Preparata and Shamos [105] and de Berg et al. [53].

This thesis considers shape matching mostly from the computational geometry point of view. A typical example of this approach is finding a geometric transformation that minimizes a similarity measure between two patterns. Similarity measure based matching has been a topic of extensive research over the last fifteen years; see the surveys by Alt and Guibas [11] and Hagedoorn and Veltkamp [72].

Many problems in geometric pattern matching can be seen as geometric optimization problems in which one seeks to optimize a function of one or more variables, subject to a number of constraints induced by a given collection of geometric objects. By exploiting the geometric as well as combinatorial properties of the problem one opts for the development of efficient algorithms; for a survey on geometric optimization algorithms, see Agarwal and Sharir [4].

1.4 Similarity measures for geometric patterns

A similarity measure is a function that assigns a nonnegative real number to pairs of patterns indicating the degree of their resemblance. The choice of the measure depends on the properties required of it and the particular geometric primitives defining the patterns, both of which depend on the particular shape matching problem for the application at hand.

For example, the Hausdorff distance can be applied on two sets of points, segments, polygonal and general curves. An alternative to the Hausdorff distance for polygonal curves is the L_p metric distance between their turning angle functions. Moreover, for curves, the Fréchet distance is more appropriate than the Hausdorff distance. Also, the similarity of closed curves can be measured by the overlap or the symmetric difference of their enclosed area. A formal definition of the most important similarity measures–including those mentioned above–will be given in Section 1.5.1. Emphasis will be given on similarity measures for point sets.

For many shape matching applications such as shape classification and content based image retrieval, particular properties of similarity measures are desirable. The next section describes such properties and discusses their implications.

1.4.1 Properties of similarity measures

A similarity or distance measure d on a collection of shapes S is a function $d : S \times S \rightarrow \mathbb{R}^+ \cup \{0\}$. The following four properties define a *metric*.

i. *Self-identity*: For all $A \in S, d(A, A) = 0$.

ii. *Positivity*: For all $A \neq B \in S, d(A, B) > 0$.

iii. *Symmetry*: For all $A, B \in S, d(A, B) = d(B, A)$.

iv. *Triangle inequality*: For all $A, B, C \in S$, $d(A, C) \leq d(A, B) + d(B, C)$.

A *pseudo-metric* is a function that satisfies properties (i), (iii) and (iv). In his thesis Hagedoorn [73] presents a theory of similarity measures in which he defines metric and pseudo-metric pattern spaces. Formally, a *metric pattern space* is a structure (X, S, d), where X is a topological space, S is a collection of subsets of X, and d is a metric on S. If d is a pseudo-metric, then (X, S, d) is called a *pseudo-metric pattern space*. The set X is called the base set. A metric defined on X is called the base metric.

In this thesis, X is $\mathbb{R}^k$, thus, the collection of patterns S is a collection of subsets of $\mathbb{R}^k$. A well-known base metric on $\mathbb{R}^k$ is the L_p metric for $p \geq 1$. For every $x, y \in \mathbb{R}^k$

$$L_p(x, y) = \sqrt[p]{\sum_{i=1}^{k} |x_i - y_i|}.$$

The case $p = 2$ is known as the Euclidean metric; unless stated otherwise, L_2 will be our base metric. Many similarity measures on shapes are based on the L_p distance, some of which are presented in section 1.5.1.

Consider a set S with a pseudo-metric d. Grouping together elements of S with zero distance results in a metric on the resulting partition of S.

Shape matching often involves the task of computing the similarity of two patterns independent of transformations. A similarity measure d on a set S is independent of the transformations in a given transformation group G on S if and only if $d(g(A), B) = d(A, B)$ for any two patterns $A, B \in S$ and any transformation $g \in G$; here, $g(A)$ denotes the application of the transformation g on A. Examples of transformation groups on $\mathbb{R}$ are *translations*, *homotheties*, i.e., translations combined with scaling, *rotations* about a fixed point, (Euclidean) *isometries* or *rigid motions*, i.e., translations combined with rotations and reflection, and *similarities*, i.e., isometries combined with scaling.

One property of similarity measures that is important for the above task is transformation invariance for a chosen transformation group.

v. *Transformation invariance*: For a chosen transformation group G on a set S and for all $A, B \in S$, $g \in G$, $d(g(A), g(B)) = d(A, B)$. This also implies that $d(g(A), B) = d(A, g^{-1}(B))$.

Formally, the *shape* of a pattern $A \in S$ for a transformation group G equals the orbit of G passing through A that is defined as $G(A) = \{g(A) | g \in G\}$. The collection of all these orbits forms the *shape space* S/G. Given a metric (pseudo-metric) d on S that is invariant for a transformation group G on S, we can define a new metric (pseudo-metric) $d^G : S/G \times S/G \to \mathbb{R}^+ \cup \{0\}$ defined by

$$d^G(G(A), G(B)) = \inf\{d(g(A), B) | g \in G\}.$$

Thus, d^G is a transformation-independent metric on the space of shapes S/G; see Section 1.5 for examples of such metrics.

For measuring the similarity between complex geometric patterns one often seeks to devise new measures based on other simpler distance measures. A typical example is the L_p-based Hausdorff distance for point sets in $\mathbb{R}^k$; see section 1.5.1. These new measures intend to capture additional ‘shape’ information of patterns and acquire new useful properties. This task is quite challenging and often one has to cancel one property in order to include another one. A natural property to ask from a similarity measure is to ‘respect’ scaling of the base distance, that is, if the base distance is scaled by some positive factor the new distance is scaled by the same factor as well. In Chapter 2 we will see an example of a distance measure on fuzzy sets that does not satisfy this property.

A general property often asked from similarity measures is robustness. Hagedoorn [73] introduced four axiom-properties expressing four different types of robustness: ‘deformation’, ‘blur’, ‘crack’ and ‘noise’ robustness. Basically, they express the following principle: if the ‘difference’ between two patterns is sufficiently limited then their distance should be small. Actually, each of the axioms defines a *continuity* property related to the particular type of ‘difference’.

We have mentioned several properties for similarity measures. A property can be useful or undesirable depending on the application at hand. An example of two shape matching applications with conflicting property interests is given below.

Efficient shape-based retrieval

Two challenging problems in content-based image retrieval are the following: retrieving images by shape, as opposed to color and texture, and efficient searching in large image databases [72, 139]. As we have already discussed in the previous sections, similarity measures for geometric patterns can be used for shape-based (geometry) image comparison. Barros et al. [23] and Vleugels and Veltkamp [139] showed that similarity measures which satisfy the triangle inequality can be used also for efficient comparison of a query image to a large number of images in a database.

This is based on the following simple observation. Consider a shape A_1 that closely matches a query shape A_q: $d(A_1, A_q)$ is relatively small. Let A_r be some reference shape. If the triangle inequality holds, $d(A_r, A_q) \leq d(A_r, A_1) + d(A_1, A_q)$, then we know that $|d(A_r, A_q) - d(A_r, A_1)|$ is small as well. We can measure the distance between a database shape A_1 and a query A_q by comparing their distances from a reference shape A_r. The computation of the latter distances can be done off-line for all A_i. So, at run-time the computation of a single $d(A_r, A_q)$ yields all A_i that resemble A_r as much as A_q resembles A_r, including those that resemble A_q.

In other words, for each image object in the database, its distance to a set of m predetermined reference or *vantage objects* is calculated; the m-vector of these distances specifies a point in the m-dimensional *vantage space*. An efficient nearest-neighbor search on these points can determine the objects that are similar to a given query object. Vleugels and Veltkamp [139] demonstrated the efficiency of this approach through experiments on a database of $72,816$ hieroglyphic polylines.

Partial matching

Partial matching is the task of finding sub-patterns common to two shapes. It is considered one of the most complex tasks in shape matching and is important in real-world applications where occlusion in the input shapes is present. Ideally, a similarity measure should take into account occlusion but give small distance for two shapes with considerable ‘overlap’. The triangle inequality is a ‘non-compatible’ property with the notion of partial matching, as the typical example of the ‘man’ the ‘centaur’ and the ‘horse’ shows [136]. It is natural to require from a similarity measure to give a small distance between the ‘man’ and the ‘centaur’ and the ‘centaur’ and the ‘horse’ since in both cases there are considerable large common sub-patterns. However the distance between the ‘man’ and the ‘horse’ is expected to be large thus cancelling the triangle inequality among the three shapes.

It is quite difficult to construct similarity measures that inherently perform partial matching. Two examples are the partial Hausdorff distance [80] and the Earth Movers Distance [44]. Partial matching can be ‘enforced’ by special problem formulations such as the *largest common pattern* problem given in Section 1.5.

1.5 Shape matching problems using similarity measures

Shape matching problems using similarity measures can be formulated in various ways that can be taxonomized according to a variety of criteria. A rough division comes from the notion of *exact* and *approximate* similarity or pattern matching. In the sequel, we assume a collection of patterns S, a similarity measure d and a transformation group G on S. Exact pattern matching can be formulated as the following *optimization* problem:

- Given $A, B \in S$, compute $\min_{g \in G} d(g(A), B)$ and a transformation $g \in G$ that achieves the minimum.

The minimum measures the similarity between A and B independently of transformations, while, depending on the nature of d, the transformation that achieves the minimum can be easily used to find the sub-pattern of B which is most similar to A.

Note that when G is the *identity* group, this problem reduces to the computation of the distance between A and B. As we will see in Sections 1.5.1 and 1.5.2 most similarity measures can be efficiently computed for patterns in two and three dimensions, thus, the identity group is excluded for the rest of this section.

Exact pattern matching has two main disadvantages [64]. Firstly, it assumes that all input geometric data are given with infinite precision and that two points match if their coordinates are identical. However, this is an inaccurate representation of reality, where the input is typically noisy and any match requires some notion of a tolerance parameter. In point pattern matching, for example, point coordinates are known to some finite precision, and hence each point is actually a ball of small radius. Secondly, most of the algorithms designed to compute exact similarity are complex with high running times. This provides the motivation for *approximations* which come in two forms: approximation algorithms and approximate similarity.

Approximation algorithms

Approximation algorithms for hard geometric optimization problems is a well developed science; see the book by Vazirani [135] and the surveys by Bern and Eppstein [25] and Arora [18]. Typically, the attention is limited to problems for which no polynomial-time exact algorithms are known or likely to exist (NP-hard problems). However, for the reasons explained above, it makes sense to look for efficient approximation algorithms also for shape matching problems for which complex and high degree polynomial running time algorithms are known. Studying approximation algorithms is also interesting for the problem structure that is revealed and the techniques that can be developed.

In this thesis we are interested in constant-factor and $(1+\epsilon)$-approximation algorithms. Assuming maximization problems, such algorithms compute a value that is at least a constant or $(1+\epsilon)$ times the optimum, for any given $\epsilon > 0$. We present such algorithms for particular shape matching problems in Chapters 3, 4 and 5.

Approximate similarity

Approximate similarity is captured by approximation schemes particularly devised for shape matching problems. The most simple approximate similarity formulation is given as a *decision* problem:

- Given $A, B \in S$ and a threshold $\epsilon > 0$, decide whether there exists some $g \in G$ for which $d(g(A), B) \leq \epsilon$.

Using this decision problem we can apply binary search over ϵ to approximate infinitesimally the minimum value of ϵ for which the decision problem is true, i.e., the solution of the corresponding optimization problem we saw in the beginning of this section. In some cases the solution space has a particular structure which allows for parametric search to be used [92, 46, 102, 132], yielding the optimal solution.

Partial matching is a difficult problem to formulate. Typically, one would like to find sub-patterns 'common' to both shapes according to some similarity measure. However, there are two parameters to be optimized here: the size of the sub-patterns and the quality of the matching. These are two conflicting tasks and the notion of a threshold helps in defining our goal. Partial matching can be formulated as the *largest common pattern* (LCP) problem and its natural dual which are defined as follows: Given $A, B \in S$ and a size measure $|.|$ appropriately defined for the patterns in S, then

- given a parameter $\epsilon > 0$ determine $A' \subseteq A, B' \subseteq B$ such that

  $$\min_{g \in G} d(g(A'), B') \leq \epsilon$$

  and $|A'|$ is maximized.

- given a parameter $K > 0$, compute

  $$\min_{A' \in A, B' \in B} \min_{g \in G} d(g(A'), B')$$

  where $|A'| \geq K$.

Note that the acromym LCP stands usually for the largest common point set problem first introduced by Akutsu et al. [8]; see section 1.5.1 for details.

Heffernan and Schirra [75] proposed an approximate decision formulation, the (ϵ, β)-approximate pattern matching. Although it was primarily used for point set matching, it can be generally defined as follows: Given $A, B \in S$ and parameters $\epsilon, \beta > 0$ and denoting $\epsilon^* = \min_{g \in G} d(g(A), B)$,

- If $\epsilon^* \leq \epsilon$, then return $g \in G$ for which $d(g(A), B) \leq (1+\beta)\epsilon$;

- If $\epsilon^* > (1+\beta)\epsilon$, then return NONE.
- If $\epsilon^* \in (\epsilon, (1+\beta)\epsilon]$, return I DONT KNOW.

The above formulation captures the intuition that determining an answer close to the optimal is harder than determining an answer that is further away; if ϵ^* lies in the approximation range $(\epsilon, (1+\beta)\epsilon]$, the algorithm cannot decide. Note that this formulation has two disadvantages over a general $(1+\epsilon)$-approximation algorithm. Firstly, in some problem instances a possibly large amount of computing time will be consumed without an approximation being returned. Secondly, the running times given for point sets [75] increase substantially when one tries to get an answer for an ϵ close to ϵ^*.

Finally, Gavrilov et al. [64] gave another general approximate scheme that captures the notion of partial matching. This is the (α, β)-approximate pattern matching that can be defined as the following *promise* problem: Given $A, B \in S$, parameters α, β and numbers l, r,

- determine $g \in G$ for which there exist $A' \subseteq A, B' \subseteq B$ such that

  $$d(g(A'), B') \leq (1+\beta)r$$

  and $|A| \geq (1-\alpha)l$.

The numbers l and r are the promise parameters: we assume that there exists a subset $A' \subseteq A$ of size l that matches some $B' \subseteq B$ to within distance r, and we wish to approximate this solution.

An obvious drawback of approximations is that they cannot be used in the typical scenario where one seeks in a collection of patterns the most similar one to a given pattern. However, an approximation can still be used for finding sub-patterns of a large pattern B that are similar to a given pattern A.

The notion of approximation algorithms for approximate pattern matching, unavoidably brings some confusion. In the sequel we give some more approximation schemes defined specifically for point set matching and summarize some algorithmic results for point set and curve matching problems.

1.5.1 Finite point sets

In this section we summarize results on point set matching based on the three most commonly used distance measures: the discrete or exact metric, the bottleneck distance and the Hausdorff distance. Our aim is to show the complexity of these problems and name a few techniques used for their solution.

Discrete metric

Given point sets A and B in $\mathbb{R}^k$, the discrete metric d_E between them is

$$d_E(A, B) = \begin{cases} 0 & \text{if } A = B \\ 1 & \text{otherwise} \end{cases}$$

The discrete metric is invariant under general homeomorphisms but has no noise tolerance. We assume that $|A| = |B| = n$. Minimizing the discrete metric under translations can be easily done in optimal $O(n \log n)$ time for any k [13]: it suffices to match those two points with the lexicographically smallest coordinate vectors and then to check whether the rest of the points match as well. For rigid motions, the problem is known as the exact *congruence* problem which can be solved optimally in $O(n \log n)$ time for $k = 2, 3$ [13]: the problem is easily reduced to the classical substring matching problem for $k = 2$ and the isomorphism of labeled, three-connected and planar graphs problem for $k = 3$.

For $k \geq 4$ the best known algorithm runs in $O(n^{\lceil k/3 \rceil} \log n)$ time [31]. This algorithm is much more involved than the ones in the previous low dimensional cases and is based on a dimension reduction technique different than the ones presented by Alt et al. [13], Akutsu [7] and Matousek [7] for the same problem. Braß and Knauer [31] also conjectured that an optimal $O(n \log n)$-algorithm for any fixed k, should be possible. Finally, reflections and scalings can be easily incorporated in all the above algorithms without affecting the running times.

When $|A| = n$ and $B = m$ with $n \neq m$ the interesting problem is to find common sub-patterns to both A and B under the exact metric. Then the largest common pattern problem and its dual reduce to the following two problems respectively:

- Find the largest common congruent subset to both A and B.
- Determine whether A is congruent to some subset of B.

For the first problem, Akutsu et al. [8] gave algorithms of time complexity $O(n^{1.43}m^{1.77} + n^2)$ for $k = 2$, $O(n^{1.89}m^{2.8} + n^3)$ for $k = 3$, $O(n^{2.87}m^4 + n^4)$ for $k = 4$, and $O(n^{k-1}m^k + n^k)$ for $k \geq 5$. These algorithms are mainly based on simple voting schemes and good upper bounds on a combinatorial-geometric quantity related to the distance multiplicity vectors of A and B. Using simple random sampling techniques in addition, the authors gave also output-sensitive randomized algorithms for computing the optimum or a $(1-\epsilon)$ approximation of the optimum; for the latter and for $k = 2$, the algorithm runs in $O((n^{1.43}m^{1.77} + n^2)K^{-2} \log^3 n)$ expected time, where K is the largest common congruent subset, and succeeds with probability $1 - n^{-1}$. In the same paper, algorithms for similarity instead of congruence were given based on the same ideas and with similar running times.

For the second problem, Akutsu et al. [8] gave an $O(\min\{n^{1.43}m^{0.77}, n^{4/3}m\})$ running time algorithm for $k = 2$. For $k = 3$, Braß [29] improved the upper bound on the maximum number of congruent triangles by Akutsu et al., and presented a randomized $O(mn^{7/4} \log n \beta(n))$ algorithm , where $\beta(n)$ is an extremely slow growing function. Finally, for $k \geq 4$, Rezende and Lee [54] presented an $O(nm^k)$-algorithm based on *circular sorting*, an extension of geometrical sorting [70].

Bottleneck distance

We first examine the case where $|A| = |B| = n$. Let $F(A, B)$ be the set of all bijections from A onto B. Then, the bottleneck distance d_B is defined as

$$d_B(A, B) = \min_{f \in F(A,B)} \max_{a \in A} d(f(a), a)$$

where d is the L_p metric. Bottleneck matching may be useful for applications where it is known that there is a bijection between the points of both patterns. However, it is not appropriate for applications where $|A| \neq |B|$ due to noise or occlusion. The bottleneck distance is invariant under rigid motions.

Efrat et al. [57] presented, for $k = 2$, an $O(n^{1.5} \log n)$-algorithm that decides whether $d_B(A, B) \leq \epsilon$, for any given $\epsilon \geq 0$; the algorithm uses Dinitz's algorithm for finding perfect matchings and takes advantage of the properties of the underlying geometric graph. Using this algorithm for the decision problem and special data structures for limiting the search over the values of ϵ, they also gave an $O(n^{1.5} \log n)$-algorithm that computes d_B. For $k = 3$, these two problems can be both solved in $O(n^{11/6+\epsilon})$ time. For any $k \geq 2$ and the L_∞-metric, the corresponding running times are $O(n^{1.5} \log^{k-1} n)$ and $O(n^{1.5} \log^k n)$ [57]. These algorithms are quite complex, thus difficult to implement, and have large constants of proportionality in their running times. The authors also gave a $(1 + \epsilon)$-approximation algorithm for d_B that uses an optimal algorithm for approximate nearest neighbor searching [19] and is relatively easy to implement. This algorithm runs in $O((1 + 1/\epsilon)^k n^{1.5} \log n)$ time for any $k \geq 2$ and $\epsilon > 0$.

The decision problem for the optimal transformation that minimizes d_B can be formulated as follows: Given $\epsilon > 0$ decide whether there exists a transformation of A such that every point in A falls into the ϵ-neighborhood of exactly one a point in B, i.e. a disk of radius ϵ centered at that point. Note that the smallest ϵ for which this is true is the minimum d_B over all transformations. For $k = 2$ and translations, Alt et al. [13] gave an $O(n^6)$-algorithm that uses standard graph theoretical techniques for finding whether a perfect matching exists for all $O(n^4)$ combinatorially distinct translations. This was improved by Efrat et al. [57] to $O(n^5 \log n)$ by using some of the techniques devised for the problems described in the previous paragraph to search for augmenting paths when moving from one combinatorially distinct translation to another one. Using parametric search the latter algorithm can be used to compute the optimum translation in $O(n^5 \log^2 n)$ time. For rigid motions the problem is usually referred to as the approximate congruence decision problem for which only a $O(n^8)$-algorithm exists [13] for $k = 2$. The optimization problem can be solved in $O(n^8 \log n)$ time. However the problem of numerical instability that motivated approximate formulations still remains: these algorithms need to compute the intersection point of curves of degree 6 with circles.

Arkin et al. [17] considered the above decision problem under various assumptions on the ϵ-neighborhoods of the points in B and gave lower and upper bounds on the number of matches and improved running times. For example, for $k = 2$ and translations when the neighborhoods are pairwise disjoint there can be only one possible match and this can be decided with a very simple algorithm in $O(n \log n)$ time. For rigid motions under the same conditions, there are point sets with $\Omega(n)$ matches and no two point sets can have more than $O(n^2)$ matches; the algorithm runs in $O(n^4 \log n)$ time.

A $(1+\epsilon)$-approximation of the minimum bottleneck distance under translations can be achieved by imposing a uniform grid of appropriate size depending on ϵ and k on the space of translations and computing a $(1+\epsilon)$-approximate bottleneck distance for each grid point [57]. This algorithm runs in $O(c(\epsilon,k)n^{1.5} \log n \log \epsilon^{-1})$ time where $c(\epsilon,k)$ is a constant depending on ϵ and k. Heffernan and Schirra [75] gave (ϵ,β)-approximate decision algorithms for $k=2$, that discretize the space of transformations using an appropriate grid, perform some “structural pertubation” of both sets A and B and finally solve a maximum-flow problem on the underlying geometric network. The algorithms run in $O(n^{1.5})$ and $O(n^{2.5})$ for translations and isometries respectively.

Finally, the LCP problem for the bottleneck distance, for $|A| = n \neq m = |B|$, can be solved in $O(m^3 n^4 \sqrt{m+n})$ time [14] for $k=2$ and rigid motions using similar techniques as the ones by Alt et al. [13]. For higher dimensions similar algorithms with high polynomial running times exist while the problem is NP-hard for unbounded dimensions [14].

Hausdorff metric

The Hausdorff metric is the most studied similarity measure in computational geometry. Perhaps the reason is that it is defined on a very general pattern collection: the set of all compact, i.e. closed and bounded, subsets of $\mathbb{R}^k$. There are numerous results even for finite point sets only; we will not try to give all these results here, but rather mention the most important ones. For two point sets A and B in $\mathbb{R}^k$, with $|A| = n, |B| = m$, the Hausdorff metric $H(A,B)$ is defined as

$$H(A,B) = max\{h(A,B), h(B,A)\}$$

where $h(A,B)$ is the directed Hausdorff distance of A to B given by

$$h(A,B) = \max_{a \in A} \min_{b \in B} d(a,b)$$

where d is one of the L_p-metrics. Sets A and B need not have the same number of points. Rather than a one-to-one matching, the Hausdorff distance achieves in general a many-to-many correspondence. The Hausdorff metric is invariant under rigid motions and quite sensitive to noise: its value is determined by the “worst” matching point.

For $k=2$ and any L_p the Hausdorff distance can be optimally computed in $O(n+m)\log(n+m)$ time [9]; for the L_2, this is simply done by using Voronoi diagrams.

Rucklidge [115] showed that the graph of the function of the Hausdorff distance under translations for $k=2$ with the L_1 or L_∞ metric can have $\Omega(n^3)$ local minima. This lower bound can be overcome by examining the depth of the intersection of unions of rectangles without actually computing the whole arrangement [42] of these unions. Using segment trees this can be done in $O(nm \log nm)$ time. Finding the smallest ϵ for which the decision problem is true can be done in $O(nm \log^2 nm)$ time using an optimization technique by Frederickson and Johnson [61]. For the same problem, using a different method based on a randomized optimization technique, Chan [34] removed a $\log nm$ factor. For higher dimensions several algorithms of the same flavor have been given by Chew et al. [40]. The corresponding lower bound for the same problem for $k = 2$ and the L_2 metric is again $\Omega(n^3)$. In this case the minimum Hausdorff distance can be found in $O(nm(n+m) \log nm)$ time by computing all local minima on the upper envelope of a set of Voronoi surfaces [48]. For $k = 3$ and any $\delta > 0$ there is a $O((nm)^2(n+m)^{1+\delta})$-algorithm based on similar ideas. For higher dimensions the problem can be solved in $O(n^{\lceil 3k/2 \rceil + 1 + \delta})$ time [40].

For rigid motions, the Hausdorff distance can be computed in $O(n^2m^2(n+m) \log^2 nm)$ [41] using similar ideas as in the case of translations for the L_1 metric, and parametric search. The corresponding lower bound is $\Omega(n^5)$ [115].

Aichholzer et al. [6] presented simple constant factor approximation algorithms for $k = 2, 3$ that exploit properties of characteristic points of a point set, the so called *reference points*. For example, for $k = 2$ they achieve a factor of $1 + 4/\pi$ in $O((n+m) \log(n+m))$ and $O(nm \log(nm) \log^* nm)$ for translations and rigid motions respectively. Based also on reference points and combining the ideas of Heffernan and Schirra [75], Chew et al. [43] presented (ϵ, β) approximate decision algorithms for the Hausdorff distance under translations and rigid motions that run in $O((m+n) \log nm)$ and $O(nm \log nm)$ respectively. Gavrilov et al. [64] gave also (ϵ, β)-approximate solutions that exploit upper bounds on approximate distance distributions and whose running time depends on the diameter of the point sets. Also, for rigid motions in two dimensions they reduced the problem to a combinatorial pattern matching using algebraic convolutions, yielding an algorithm that runs in $O(n(n + \Delta))$ where Δ is the diameter of set A and assuming $n \geq m$.

The techniques of Gavrilov et al. give also algorithms for the (ϵ, β) -largest common point set problem for $k = 2, 3$. For $k = 2$ the algorithm runs in $O(\Delta mn)$ time, logarithmic factors omitted. Finally, Indyk and Venkatasubramanian [82] introduced a generalized bottleneck distance that includes both the bottleneck and the Hausdorff distance as special cases. This measure relaxes the condition that the mapping must be one-to-one but guarantees that only a few points are mapped to any point. Using again combinatorial pattern matching techniques they gave a randomized algorithm for the (α, β) promise problem in the plane that runs in almost linear time and succeeds with high probability.

1.5.2 Curves

For two objects in the plane, one can base the distance measure on their boundaries. For example, assuming that each boundary is given by a closed curve, one can try to find the translation that minimizes the Hausdorff distance [102] or the Fréchet distance [12] between the two curves. Since these distance measures are based on the boundaries of the objects, they are fairly sensitive to noise. The *area of overlap* (or the *area the symmetric difference*) of two polygons is less sensitive to noise [51, 10] and therefore more appropriate for certain applications.

The function of the area of overlap of two translated simple polygons was first studied by Mount et al. [96]. They showed that this function is a continuous, piecewise polynomial surface of at most degree two with $O((nm)^2)$ pieces and a representation that can be computed in $O((nm)^2)$ time for two polygons with n and m vertices. No efficient algorithm is known that computes the translation that maximizes the area of overlap and does not compute the whole function. De Berg et al. [51], gave an efficient $O((n+m)\log(n+m))$ algorithm for determining the optimal translation for the case of *convex* polygons; they also gave a constant-factor approximation algorithm. Finally, Alt et al. [10] used similar techniques as in [51] and gave a constant-factor approximation of the minimum area of symmetric difference of two convex shapes.

1.6 Overview of the thesis

Point set matching has been studied in depth, as the relevant survey in the previous section shows. However there are hardly any results on more complex cases where a shape is modeled by points that, apart from their coordinates, are characterized by an extra attribute, a weight. As we will see in the next chapter, this model is more appropriate for certain cases where the relative importance of a point needs to be taken into account as a separate parameter and not an extra coordinate; for example, see Section 2.7.3 on matching music scores. This thesis deals with shape matching problems related to weighted-point sets. As one can expect, most of these problems have high complexity, as their point set counterparts do, although a clear comparison cannot be made since the similarity measures used are different. We present several approximation algorithms for weighted-point set matching problems using different similarity measures. The rest of this section summarizes our results.

The notion of weighted point sets is presented in Chapter 2, where the focus lies on weighted point set matching using transportation distances. Here the weight of a point is defined as an amount of mass. An example of such distance is the Earth Mover's Distance (EMD), first introduced by Rubner et al. [113] for color-based image retrieval purposes. The EMD between two weighted point sets measures the minimum amount of work needed to transform one set into another one by weight transportation. One disadvantage of the EMD is that it does not obey the triangle inequality. After giving its definition and analyzing its properties, we introduce a variant, the Proportional Transportation Distance (PTD), which is shown to be a pseudo-metric. We exhibit the potential use of EMD and PTD in shape matching and retrieval—discussing the advantages of one over the other—by testing them on a variety of data sets: company logos, fish contours, music scores and polyhedral models. The experiments show that the EMD and PTD can be effectively used in shape matching and that the use of PTD, together with the vantage objects method discussed in Section 1.4.1, has a clear advantage over the EMD for time-efficient retrieval in large databases. These results have been previously published in Giannopoulos and Veltkamp [65] and Typke et al. [129].

The next three chapters of the thesis take a more algorithmic approach by dealing with the fundamental shape matching problem as this was defined in Section 1.5. We present approximation algorithms for several optimization problems in which the objective function is fairly complex and for which no (practical) algorithms for finding the exact optimum are known yet. A general and simple track is followed in all the chapters. First, we establish lower bounds on the objective function. These direct our search for the optimum which we then approximate by using an appropriate discretization of the variables' space.

Chapter 3 studies the problem of minimizing the EMD of two weighted-point sets in the plane under translations and rigid motions; a preliminary version of this chapter has been published in Cabello et al. [32]. We give the first polynomial-time $(1+\epsilon)$ and $(2+\epsilon)$-approximation algorithms for this problem. In the general case where the sets have unequal total weights our algorithms run in $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$ time for translations and a $(2+\epsilon)$-approximation in $O((n^4m^2/\epsilon^4)\log^2(n/\epsilon))$ time for rigid motions. When the sets have equal total weights, the respective running times decrease to $O((n^2/\epsilon^4)\log^2(n/\epsilon))$ and $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$. We also show how to compute a $(1+\epsilon)$-approximation of the minimum cost assignment under translations and rigid motions in $O((n^{3/2}/\epsilon^{7/2})\log^5 n)$ and $O((n^{7/2}/\epsilon^{9/2})\log^6 n)$ time respectively.

In Chapter 4, the weight of a point is relative to the area or volume inclosed in a simple object around the point. In this way a weighted-point set can define a shape given by the union of simple objects. In particular we deal with unions of either homothets of a single planar convex object or arbitrary planar fat convex objects under certain conditions. A homothet of an object is a translated and/or scaled version of the object, while fatness is a property that ensures that the object is not infinitesimally thin. We measure the similarity of two weighted-point sets by the area of their respective unions; see Figure 1.2 for an example. Our main result is an almost quadratic $(1-\epsilon)$-approximation algorithm for the minimum area of overlap of two such unions under translations; these results

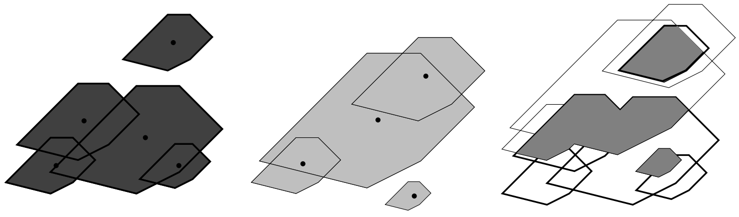


Figure 1.2: Two sets of two-dimensional homothets and the area of overlap of their respective unions for some translation.

have already appeared in a technical report by de Berg et al. [52], and are published in de Berg et al. [50].

Chapter 5 extends the transformation space for the latter problem to rigid motions. We focus on unions of disks and give a $O((n^2m^2/\epsilon^3)\log m)$-time algorithm that computes a $(1-\epsilon)$-approximation of the optimum. Under the condition that the optimum is at least a constant fraction of the area of one of the two unions, we show how to compute such an approximation in $O((m^2/\epsilon^4)\log(m/\epsilon)$ $\log^2 m)$ time; in this case the algorithm is probabilistic and succeeds with high probability. A preliminary version of this chapter has been published in de Berg et al. [50].

Chapter 2

Weighted point set matching

2.1 Introduction

A weighted point set is a set of points each having an attribute or weight. A typical scenario where the use of weighted point sets arises is the following. Let's assume that a two- or three-dimensional shape has been reduced to a finite set of points that describe the shape. Practical shape matching applications impose computational limitations that pose a trade off between the size of the point set and its shape descriptive power. Normally, one would like to decrease the size of the set without loosing too much 'shape' information. A simple way to achieve this is to search for points in the shape whose position is important and associate a relative weight with them. For two-dimensional shapes, for example, the weight could measure the amount of some property such as the curvature of the points along a contour.

This chapter deals with similarity measures for matching geometric patterns given as sets of weighted points. Our goal is to search for distance functions that, apart from the position of the points, incorporate this extra information as well.

In general, the meaning of a point weight and the way it affects matching depends on the particular application. For example, the weight could be a certainty measure of the correctness of the points' coordinates. The interpretation for matching could be that points with low weight values match easier than those with high weight values, since their *position* is uncertain, and they have more freedom to move around. Alternatively, a small weight could mean that the *existence* of the point is uncertain; the smaller its weight the less its influence in the matching.

If the weights are in [0, 1] they could be interpreted as "membership values" used in fuzzy sets. For a set $S \subseteq \mathbb{R}^k$, a fuzzy set U is a function $U : S \rightarrow [0, 1]$; S is the support set of U. The value $U(x)$ is thought of as the membership value of x in S. In other words, the points take their weights from the interval [0, 1], where weight 0 means the point is not contained in the set and weight 1 means the point is completely contained in the set. An ordinary or "crisp" set is a fuzzy set whose membership values are either 0 or 1.

It is tempting to demand of any distance defined for fuzzy sets to be a metric invariant under rigid motion, that respects scaling of the underlying distance. In [30] it is proved that there is no metric that satisfies all these properties. So it makes sense to take a close look at distance properties, and choose those that are relevant for the application in mind. We are mainly interested in a distance measure for weighted point sets that is symmetric, invariant under rigid motion, respects scaling and obeys the triangle inequality. As explained in Section 1.4.1, these properties make it suitable for shape based image database retrieval. Depending also on the application, the property of *positivity* can be dropped as long as this does not result in counter-intuitive identifications of arbitrarily different sets.

In the next two sections we survey related work on distance measures for weighted point sets. We expose advantages and disadvantages of proposed distances in the literature and show that either the properties of interest don't hold, or when they do, they can produce extremely counter-intuitive results. Then in section 2.5 we introduce a pseudo-metric, the proportional transportation distance (PTD), which has the desired properties and, although it does not follow the positivity property, gives zero distance only for weight-scaled versions of the same set. We exhibit its potential use in section 2.7, by testing it on four different test data sets of company logos, fish contours, music scores and polyhedral models. Section 2.6, gives simple upper bounds on the PTD and the Earth Movers Distance (EMD) related to incremental change of the point sets. Finally, partial matching based on transportation distances is discussed in Section 2.8.

2.2 Fuzzy sets and the Hausdorff distance

An often used metric for compact subsets of $\mathbb{R}^k$ is the Hausdorff distance, that has been already defined for finite point sets in Section 1.5.1. The directed Hausdorff distance between two sets A, B both compact subsets of $\mathbb{R}^k$ is given by

$$h(A,B) = \sup_{a\in A} \inf_{b\in B} d(a,b)$$

where d is one of the L_p metrics. Then the Hausdorff metric $H(A,B)$ is defined by

$$H(A,B) = \max\{h(A,B), h(B,A)\}.$$

The rest of the section gives an overview of the extensions of the Hausdorff metric to weighted point sets that have been proposed. All of them use the notion of level sets. Let S be the support set and a non-empty compact subset of $\mathbb{R}^k$. For a fuzzy set w on S, the set $\sigma_t(w) = \{x \in S | w(x) \geq t\}$ is the "internal"

t-level set of w; the set $\xi_t(w) = \{x \in S | w(x) \leq t\}$ is called the "external" t-level set of w.

An interesting question is whether the Hausdorff metric can be extended to fuzzy sets. Chaudhuri and Rozenfeld [37] presented such an extension that is a metric, respects scaling of the sets and is invariant under isomorphisms. We give its definition below.

Let each fuzzy set U have a finite number of distinct membership values and let $T = \{t_1, \ldots, t_m\}$ be the set of all membership values of all such sets. We assume that both $\sigma_t(U), \xi_t(U)$ are compact for every U and $t \in T$. Then the Hausdorff distance $H(A, B)$ between two fuzzy sets A and B is given by

$$H(A, B) = \frac{\sum_{k=1}^{m} t_k H(\sigma_{t_k}(A), \sigma_{t_k}(B))}{\sum_{k=1}^{m} t_k}$$

The serious drawback of this definition is that it is not always valid, since not all the sets have the same maximum membership value. As a consequence the level sets are empty for some sets and membership values, and the Hausdorff distance between the empty and some other set is undefined. This problem can be solved by modifying the sets such that they have the same maximum membership value. This has the undesirable effects that the nature of the sets change, and that sets that initially differ only in their weights cannot be distinguished anymore, e.g. two sets that are equal at all support points except their maxima. In other words, positivity is not satisfied any more and the distance fails to differentiate between fuzzy sets with arbitrarily different maximum membership values. This problem can be tackled by adding a term that is a metric itself, denoting dissimilarity rather than geometrical distance based on weights of the original sets only. On the other hand, under this correction, the metric will no more respect scaling of the underlying distance. Both definitions coincide with the ordinary Hausdorff distance for crisp sets.

Another metric was defined by Boxer[28]. Here the problem of emerging null level-sets is solved by introducing an auxiliary compact set S' where $S \cap S' = \emptyset$, and extending every fuzzy set A defined on S to the fuzzy set A' defined on $S \cup S'$ such that $A(x) = 0$ for all $x \in S'$. Under this definition the resulting distance is a metric, respects scaling but is not motion invariant. Another disadvantage of this metric is that it does not coincide with the ordinary Hausdorff distance in the case of ordinary sets. For a general compact set S, the selection of S' might be difficult and different selections have different effects on the metric. The arbitrary choice of S' makes this metric less appealing than it seems, though in the simple case of S being the set of pixels of an image, S' can naturally be the border of the image.

Last, another proposal came from Fan[59]. As a compact subset of $\mathbb{R}^k$, S is bounded. We arbitrarily choose the diameter d of S; then $d = \sup H(A, B)$ exists for any two fuzzy sets A, B on S. Then the Hausdorff metric can be extended such that $H(\emptyset, \emptyset) = 0$ and $H(A, \emptyset) = d$ for all fuzzy sets A on S. Under this convention we can use the definition of Chaudhuri and Rozenfeld given above. This metric is motion invariant, coincides with the Hausdorff distance for crisp sets but does not respect scaling since d is fixed. An intuitive drawback is that the value of the metric can be heavily biased by the diameter d.

We conclude that no extension of the Hausdorff distance that meets our goals is known. In the next sections we take an alternative approach based on mass transportation distances.

2.3 The mass transportation problem

The mass transportation problem also known as the Monge-Kantorovich mass transference problem was first introduced by Monge [95] back in 1781 in the following way:

> Split two equally large volumes into infinitely small particles and then associate them with each other so that the sum of products of these paths of the particles to a volume is least. Along what paths must the particles be transported and what is the smallest transportation cost?

In 1942 Kantorovich reformulated the problem as a continuous linear optimization problem over a convex set. An extensive work on the history, theory, modifications and applications of general Mass Transportation Problems can be found in a two volume work of Rachev and Rüschendorf [106, 107]. The possible use of the mass transportation problem in defining a similarity measure between two shapes was first considered by Mumford [97], while Rote [110] proposed a metric for polygons based on similar ideas.

In this thesis, we focus on the discrete version of Kantorovich's formulation, whose name Earth Movers Distance (EMD) is attributed to Rubner et al. [113]. The EMD has been successfully used for color-based image retrieval by Cohen [44] and Rubner [112] as well as shape-based image retrieval by Fry [62] and Cohen [44]. We give the definition of the EMD below.

2.4 The Earth Movers Distance

2.4.1 Definition

Let $A = \{a_1, a_2, \ldots, a_m\}$ be a weighted point set such that $a_i = \{(x_i, w_i)\}, i = 1, \ldots, m$, where $x_i \in \mathbb{R}^k$ with $w_i \in \mathbb{R}^+ \cup \{0\}$ being its corresponding weight. Let also $W = \sum_{i=1}^{m} w_i$ be the total weight of set A.

The Earth Movers Distance between two weighted point sets measures the minimum amount of work needed to transform one to the other by moving weight under certain conditions which are discussed later on. Intuitively speaking, a weighted point a_i can be seen as an amount of earth or mass, equal to w_i units, situated at x_i; alternatively it can be taken as an empty hole of w_i units of mass capacity. We can arbitrarily assign the role of the supplier to one set and that of the receiver/demander to the other one, setting, in that way, the direction of weight movement. The EMD then, measures the minimum amount of work needed to fill the holes with earth.

The EMD can be expressed as a linear program. Given two weighted point sets A, B and a ground distance d, we denote as f_{ij} the elementary flow of weight from x_i to y_j, over the elementary distance d_{ij}. If W, U are the total weights of A, B respectively, the set of all feasible flows $\mathcal{F}(A, B)$ between A and B is defined by the following constraints:

1. $f_{ij} \geq 0, i = 1, \ldots, m, j = 1, \ldots, n$
2. $\sum_{j=1}^{n} f_{ij} \leq w_i, i = 1, \ldots, m$
3. $\sum_{i=1}^{m} f_{ij} \leq u_j, j = 1, \ldots, n$
4. $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\{W, U\}$

These constraints simply say that each particular flow is non-negative, a point from the "supplier" set does not give away more weight than it has and a point from the "receiver" set does not receive more weight than its capacity. Finally the total weight moved is the minimum of the total weights of the two sets.

The flow of weight f_{ij} over a distance d_{ij} is penalized by its product with this distance. The sum of all these individual products-costs is the total cost paid in order to transform A to B. The EMD(A, B) is defined as the minimum total cost over $\mathcal{F}(A, B)$, normalized by the weight of the lighter set; a unit of cost or work corresponds to transporting a unit of weight by a unit of ground distance. That is:

$$\mathrm{EMD}(A, B) = \frac{\min_{F \in \mathcal{F}(A,B)} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\min\{W, U\}} \tag{2.1}$$

Stated in a different way, the EMD is the average ground distance that weights travels during an optimal flow [45].

2.4.2 Properties and Computation

The most important properties of the EMD can be summarized below:

1. It is a metric if the ground distance is a metric and is applied on the space of equal total weight sets. A nice proof of this can be found in [112].
2. Its transformation invariance group coincides with that of the ground distance (assuming transformations that do not modify the weight of the points).
3. It respects scaling of the underlying ground distance.
4. It is invariant under uniform weight scaling.

5. It allows for partial matching by definition, thus showing some tolerance for occlusion, since the total weights need not be the same. In the case of two sets with unequal total weights, some of the weight of the heavier distribution remains unused or unmatched depending on the direction of the weight flow.

6. It is continuous, in other words, infinitesimal small changes in position and/or weight of existing points cause only infinitesimal change in its value. Moreover addition of a point, i.e noise (which can be seen as increasing its weight from zero to a positive value) can lead to an arbitrary small change in the EMD's value by making the point's weight arbitrarily small. These two facts simply imply that the EMD is deformation and noise robust according to the corresponding robustness axioms presented in [73].

The EMD can be computed efficiently by solving the corresponding linear programming problem, using for example a streamlined version of the simplex algorithm for the transportation problem [76]. In practice simplex performs well, but in theory it can perform an exponential number of steps before giving a solution. Theoretically better - polynomial time - algorithms for general linear programming, like an interior point algorithm could be used; however it is likely to perform better than the simplex method only for very large problem sizes. Since the transportation problem is a special case of the minimum cost flow problem in networks, a polynomial time algorithm that solves the latter can be used as well [5]; see Section 3.2 for further details. Finally, for the special case of the L_2-based EMD in the plane one can use the algorithm of Atkinson and Vaidya [20] that runs in $O((n+m)^{2.5}\log(n+m)\log N)$ where N is the magnitude of the largest supply or demand at any point.

2.4.3 Drawbacks of the EMD

In this section we identify two drawbacks of the EMD relevant for our purpose. Let $\mathcal{N}$ denote the space of weighted point sets, in which any two sets can have unequal total weights. The EMD has the following drawbacks when applied on $\mathcal{N}$:

1. It does not obey the positivity property. The EMD does not take into account the surplus of weight, if any, between two sets. As a result, there are cases where it does not distinguish between two non-identical sets. For example in Figure 2.1, B' was constructed by adding just the point b'_i of arbitrary weight u to B. However EMD(B, B')=0. We can add new points or weight on existing points on the left of b'_2 thus making B' arbitrary different from B, but its distance to B will not increase.

2. It does not obey the triangle inequality. Figure 2.1 gives a simple counterexample. Moving point c_1 to the right, thus increasing ϵ, can invalidate the triangle inequality between sets A, B and C: The inequality holds for $\epsilon = 1$

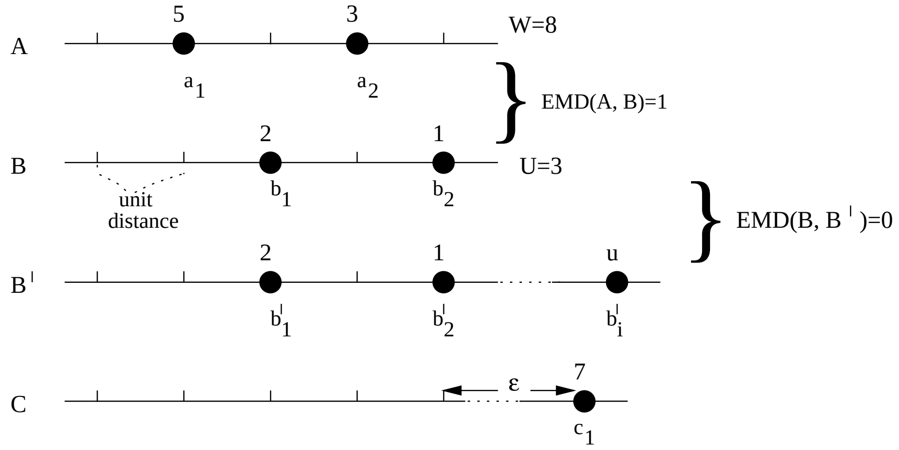


Figure 2.1: The EMD does not distinguish between sets B and B', and does not satisfy the triangle inequality for sets A, B and C.

but not for $\epsilon = 10$. As a result the EMD prevents the triangle inequality from being used in speeding up database retrieval; see Section 1.4.1

Consequently the EMD on $\mathcal{N}$ is not a metric.

2.5 The Proportional Transportation Distance

An interesting question, that naturally arises, is the following: is there a similarity measure based on weight transportation such that the surplus of weight between two point sets is taken into account and the triangle inequality still holds?

In the sequel we present a new distance for weighted point sets in $\mathcal{N}$. Let $A, B \in \mathcal{N}$. When measuring the distance from A to B, rather than taking A as the supplier and B as the demander moving only as much weight as needed, trying to fill the 'holes' with 'earth', we move the total weight of A to the positions of the points in B. What we measure then, is the minimum amount of work needed to transform A to a new set A' that resembles B. In particular, we redistribute A's total weight from the position of its points, to the position of B's points leaving the old percentages of weights in B the same.

We call this distance the Proportional Transportation Distance (PTD); it is defined as follows. Let A, B be two weighted point sets and d a ground distance. The set of all feasible flows $\mathcal{F}(A, B)$ from A to B, is now defined by the following constraints:

1. $f_{ij} \geq 0, i = 1, \ldots, m, j = 1, \ldots, n$
2. $\sum_{j=1}^{n} f_{ij} = w_i, i = 1, \ldots, m$

3. $\sum_{i=1}^{m} f_{ij} = \frac{u_j W}{U}, j = 1, \ldots, n$
4. $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = W$

The PTD(A, B) is given by:

$$\mathrm{PTD}(A, B) = \frac{\min_{F \in \mathcal{F}(A,B)} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{W} \tag{2.2}$$

Constraints 2 and 4 forces all of A's weight to move to the positions of points in B. Constraint 3 ensures that this is done in a way that preserves the old percentages of weight in B. Next, we examine PTD's properties.

2.5.1 Properties

Let us take a closer look at PTD's definition. While measuring the PTD(A, B) for any sets A and B, if we substitute the variables f_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, m$, in its LP formulation, call it LP$_1$ with $f'_{ij} W$ ($W \neq 0$ is the total weight of A) we get the following LP problem:

$$\min_{F \in \mathcal{F}(A,B)} \sum_{i=1}^{m} \sum_{j=1}^{n} f'_{ij} d_{ij}, \tag{2.3}$$

where $\mathcal{F}(A, B)$ is defined by:

- $f'_{ij} \geq 0$
- $\sum_{j=1}^{n} f'_{ij} = w_i / W$
- $\sum_{i=1}^{m} f'_{ij} = u_j / U$
- $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = 1$

It is clear that this new formulation, call it LP$_2$, gives us the distance between the two sets of percentages of weights in A, B. Note that the total weights of the new sets are both equal to one. Since the substitution function $f_{ij} = f'_{ij} W, W \neq 0$ is bijective, LP$_1$ is equivalent to LP$_2$. This means that we are working again on the space of equal total weight sets.

However, it's obvious that more than one LP$_1$ problem can be equivalent to the same LP$_2$ problem i.e. any two weighted point sets of the same cardinality and coincident, can have the same percentages of weight at the same positions although their corresponding individual weights are different.

We can now state the properties of PTD.

1. It obviously has the *identity* property.
2. It obeys the *triangle inequality*. This follows from the equivalence between the two LP formulations stated above.

3. It does not follow the *positivity* property since the distance between positionally coinciding sets with the same percentages of weights at the same positions is 0. However this is the only case in which the distance between two non-identical point sets is zero. The PTD will distinguish two sets B and B' where the one came from the other by adding even only one point; see example in Figure 2.1.

4. It has all the other properties that the EMD for equal total weight sets has.

It follows that the PTD is a pseudo-metric. Of course, by identifying sets with zero distance we can produce a metric on the resulting partition of the set $\mathcal{N}$ of generally unequal total weight sets.

2.6 Lower and Upper Bounds

Computing the EMD or PTD can be time consuming, especially for large point sets. In certain cases, an upper or lower bound can be as good as the exact value of the distance. The quality and the usefulness of any such bound depends of course on how close to the exact value it is and how quickly it can be computed.

Lower bounds on the EMD, both for equal and unequal total weight sets, that can be computed efficiently were given by Cohen [44] and Rubner et al. [114]. The bounds were used to cut the cost of nearest neighbor queries in very large image databases. Specifically, a kth nearest neighbor query returns the k database images which are closest to the query. During the query processing, an exact EMD computation need not be performed if there is a lower bound on the EMD which is greater than the kth smallest distance computed so far. In the next chapter we will see another use of lower bounds in the design of approximation algorithms for the minimum EMD under transformations.

Suppose we have computed the distance between two large weighted point sets and we introduce one new point in one of the sets. We would like to avoid computing the distance from scratch; depending on the application at hand an answer to the following question may be good enough: Will the distance get smaller or larger? If it gets larger, can we give an upper bound on this change? In general, due to limited computational resources available, one may just want a value close to the optimal. Such an upper bound can be achieved, for example, by computing a near-optimal basis for the simplex-based transportation algorithm [116]. In the following sections we establish some simple upper bounds for EMD and PTD and give an example of their use.

2.6.1 Upper Bounds on the EMD

We distinguish between the following two cases.

Adding a point to the set with the largest total weight.

Due to the partial matching that the EMD implicitly performs, the addition of a point to the set with the largest total weight or to any of the two sets when their total weights are equal - will either decrease the distance or leave it the same. If every distance of the new point to each of the demanders to which non-zero weight is transported is larger than all the distances over which the weight is transported, the EMD does not change. This means that the cost of covering the demand cannot get less since the new point is too far away.

If the above condition does not hold, there is a potential of lowering the distance. An upper bound can be easily established by following the greedy approach: Find all the demanders who are closer to the new point than their previous suppliers are. Starting from the closest demander transport as much weight as possible from the new point, canceling the equivalent previous flow. We continue the above procedure until the new point has no more weight available.

Adding a point to the set with the smallest total weight.

In this case the new point will always be taken into account. In particular let A, B be the two weighted point sets and $B' = B \cup \{x\}$, where x is the new point with weight u_x. Let also $W, U, U' = U + u_x$, with $W > U$ be the total weights of A, B, B' correspondingly. Note that if $U' > W$ only the amount of $W - U$ is going to be used from x's weight.

Suppose that $\mathrm{EMD}(A, B) = (\min_{F\in\mathcal{F}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij})/U$ has been computed. A trivial upper bound can be established by using the flow $\{f_{ij}\}$ defining $\mathrm{EMD}(A, B)$ and matching all or part of x's weight in a greedy way: transport weight to the unsatisfied demanders starting from the closest one. Looking only at x, this is the minimum cost way to match all or part of its weight with the sub-distribution of A that is left after its initial matching with B. Let us call this cost WORK_x. This gives us the following:

$$\mathrm{EMD}(A, B') \leq \frac{\min_{F\in\mathcal{F}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij} + \mathrm{WORK}_x}{U + u_x} \tag{2.4}$$

We can check whether this upper bound is less than or equal to $\mathrm{EMD}(A, B)$. This gives the following relation $(\min_{F\in\mathcal{F}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij})/\mathrm{WORK}_x \geq U/u_x$; when it holds we are sure that $\mathrm{EMD}(A, B') \leq \mathrm{EMD}(A, B)$.

2.6.2 Upper Bounds on the PTD

Let again A, B be the two weighted point sets and $B' = B \cup \{x\}$, where x is the new point with weight u_x. Since the triangle inequality holds for the PTD we have the following: $\mathrm{PTD}(A, B') \leq \mathrm{PTD}(A, B) + \mathrm{PTD}(B, B')$. This already gives us an upper bound and $\mathrm{PTD}(B, B')$ can be easily computed in $O(|B|)$ time in the following way: We assign zero-cost flow from each point in B to the same point in B'. The demand of all points common to both sets is

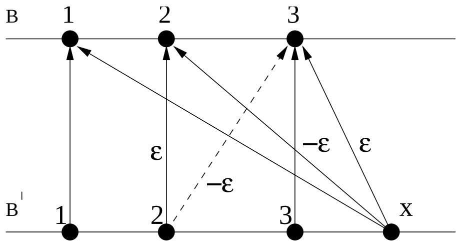


Figure 2.2: Optimal flow $[f_{ij}]$ for ptd(B, b') is shown by thick arrows.

thus covered, since the points in B' have smaller weight than their counterparts in B. Next, we transport all the remaining weight from B to the new point x. It is not difficult to see that this flow is optimal, since, assuming that the triangle inequality holds for the ground distance, the residual network contains no negative cost-directed cycles; for information on optimality conditions for min-cost network flows see [5].

A short proof of the above statement goes as follows: assume a point set B with three points, and the set B' that contains one more point x as shown in Figure 2.2. The flow established by the procedure above has the following non-zero individual flows: $f_{11}, f_{22}, f_{33}, f_{x1}, f_{x2}$ and f_{x3} with direction from B' to B. Suppose that this flow is not optimal; this implies that there is at least one more non-zero individual flow, e.g. f_{23}. We show that the new flow is not optimal. Assume that $f_{23} = \epsilon > 0$. We first cancel the flow f_{23}; this gives an excess of weight, equal to ϵ, to point 2 of B. In return, this gives us the opportunity to cancel an equal amount of flow from f_{x2} and add it to f_{x3}. Thus we have identified a directed-cycle in the residual network with cost: $-\epsilon d(2,3) + 0 - \epsilon d(x,2) + \epsilon d(x,3)$. It is now easy to see that if the triangle inequality holds for d then the cost is negative (d is assumed to be symmetric and positive).

If we take a look at the distribution of B's weights we see that the weight u_j/U of a point $b_j \in B$ becomes $u_j/(U+u_x)$. We can establish an upper bound following the procedure described next. For each $b_j \in B$ assign a new flow f'_{ij} to it, starting from its closest previous supplier–this information is given to us by the previous optimal flow f_{ij}–until $\sum_{i=1}^{m} f'_{ij} = u_j/(U+u_x)$. Note that $\sum_{j=1}^{n} f'_{ij} < w_i/W$ for each $a_i \in A$. Finally all the weight left unmatched is moved to the new point x.

Now, $\mathrm{PTD}(A,B') \geq \sum_{i=1}^{m}\sum_{j=1}^{n} f'_{ij} d_{ij} + \sum_{i=1}^{m} d_{ix}(w_i/W - \sum_{j=1}^{n} f'_{ij})$. Let us call the first term α and the second term β. If $\beta \leq \mathrm{PTD}(B,B')$ then this upper bound is better than the one based on the triangle inequality. Moreover, if $\beta \leq \mathrm{PTD}(A,B) - \alpha$, then we know that $\mathrm{PTD}(A,B') \leq \mathrm{PTD}(A,B)$.

2.6.3 An example of the Lower and Upper Bounds' use

Assume that we have computed a lower and upper bound $\mathrm{lb_D}(A,B'), \mathrm{up_D}(A,B')$ respectively and $\mathrm{D}(A,B)$ is known–D is either the EMD or PTD:

If $\mathrm{lb}_{\mathrm{D}}(A, B') \geq \mathrm{D}(A, B)$ then we conclude that the addition of the new point to B will either increase the distance or leave it the same. On the other hand if $\mathrm{ub}_{\mathrm{D}}(A, B') \leq \mathrm{D}(A, B)$ then the distance will decrease or stay the same.

If neither of the above conditions hold then an upper bound can be used just as an indication of the distance's value. If this indication does not satisfy the user, the bound can be used as an initial feasible solution in an algorithm that solves the transportation problem. For example, one could use the upper bound as a feasible flow in the cycle-canceling algorithm for the min-cost flow problem in networks [5]. This algorithm finds negative cost-directed cycles in the residual network and augments flows on these cycles; in our case, any of these cycles must contain the new point added in B. As already mentioned, the algorithm terminates when the residual network contains no negative cost-directed cycle.

2.7 Experimental Results

We have tested our pseudo-metric on two different image collections, one of company logos and another one of fish contours; the results are described in Sections 2.7.1 and 2.7.2. A preliminary version of these results appeared in Giannopoulos and Veltkamp [65]. Our primary objective is to show that it can be actually used in shape-based object recognition; this together with its properties, namely the triangle inequality, would make it a good candidate for image retrieval applications. However, we did not strive for the best matching and retrieval procedure possible for logos and fish specifically; in the course of the next sections, we discuss also about how our experimental results can be improved.

In both cases the original images are reduced to feature weighted point sets. The weight of a point represents its importance in the image: its edge strength value in the first collection and a curvature measure value in the second. In order to calculate the PTD between two sets, each set is first normalized by its total weight as the LP_2 formulation suggests. The distance computation is based on the EMD publicly available code [111]. This code implements a special version of the simplex method, see Section 2.4.2.

Our calculations are made independent of translation, rotation and scaling by a simple preprocessing procedure that put the original shapes into some standard pose. Solving the optimal transformation problem instead, will give results that are at least as good with a dramatic increase of the computation time. No polynomial time algorithm is known that minimizes the transportation distance under translation; see Chapter 3 for details.

Finally, in Sections 2.7.3 and 2.7.4, we summarize the experiments performed by Typke et al. [129] and Tangelder and Veltkamp [125] where our proposed pseudo-metric was used for measuring melodic similarity and polyhedral model retrieval correspondingly.

All the experiments described in this section were performed on a 2-GHz Pentium 4 system with Windows XP.

2.7.1 Logos

The UMD-Logo-Database [56] contains 105 grey-scale images that are black and white scanned versions of logos. We have selected only 14 logos, good representatives of the whole database, to test the PTD. These can be seen in the first column of Figure 2.3.

The weighted point set for each logo is constructed as follows. First, we identify edges using a Canny edge detector [143] and corners using the SUSAN corner detector [122, 121]. Then an intermediate weighted point set is constructed by selecting only edge points that are also corners, keeping the intensity in the edge image, rather than in the original image, as the point weight. The higher the intensity, the more important the point. In this way, we keep the most dominant points in the logo's shape, reducing the initial number of edge points from some thousands to some hundreds - in most cases less than 500. This makes the computation of the PTD between two sets possible in a reasonable amount of time. For example the PTD between two sets of around 300 points each, takes no more than 5 seconds on our configuration; see previous section.

Finally, the bounding box of the logo in each image is first centered to the origin and then scaled to a unit area rectangle. Taking into account the purpose of our experiments and the type of the logo images in the database, this simple normalization compensates satisfactorily for translational and scaling effects in the original images. Since there are hardly any rotated versions of logos we did not incorporate any rotational normalization.

Note that 'weak' outliers, that can frequently occur in scanned versions of images, may not be detected as edges or points. Since the feature point extraction precedes the normalization step, outliers may be prevented from wrongly determining the bounding box of each logo.

We have computed the distance for every pair of the selected logos. Each row in Figure 2.3 shows the initial logo image, its corresponding weighted point set and best 6 matches. The width of the dots in the weighted point sets is relative to the values of the weight. This can be easily seen in the apple in the second row where points in its interior have lower weight than points on its outline. As an indication of the actual distance values, here are the values for the Apple logo in the first row, from left to right: 0.0, 0.087, 0.146, 0.147, 0.152 and 0.157. Interestingly enough, the worst match for both Apple logos is the Microsoft logo with a distance value of 0.3958 and 0.3956 respectively.

For almost all logos the weights are within a very small range. Discarding the weights, thus setting them all to 1, can produce bad matches only for those sets that have many points with low weight. For example a query with the second Apple logo in the query column, would give as a best match, after itself, the rounded logo with the bird inside (fourth in the query image column); the other Apple logo, first in the column, would comes second. This is due to the fact that more weight in the interior of the query Apple logo has to travel towards the outline of the first Apple logo. In general, the use of the edge strength as the weight value will really pay off if the test database consists of noisy (e.g.

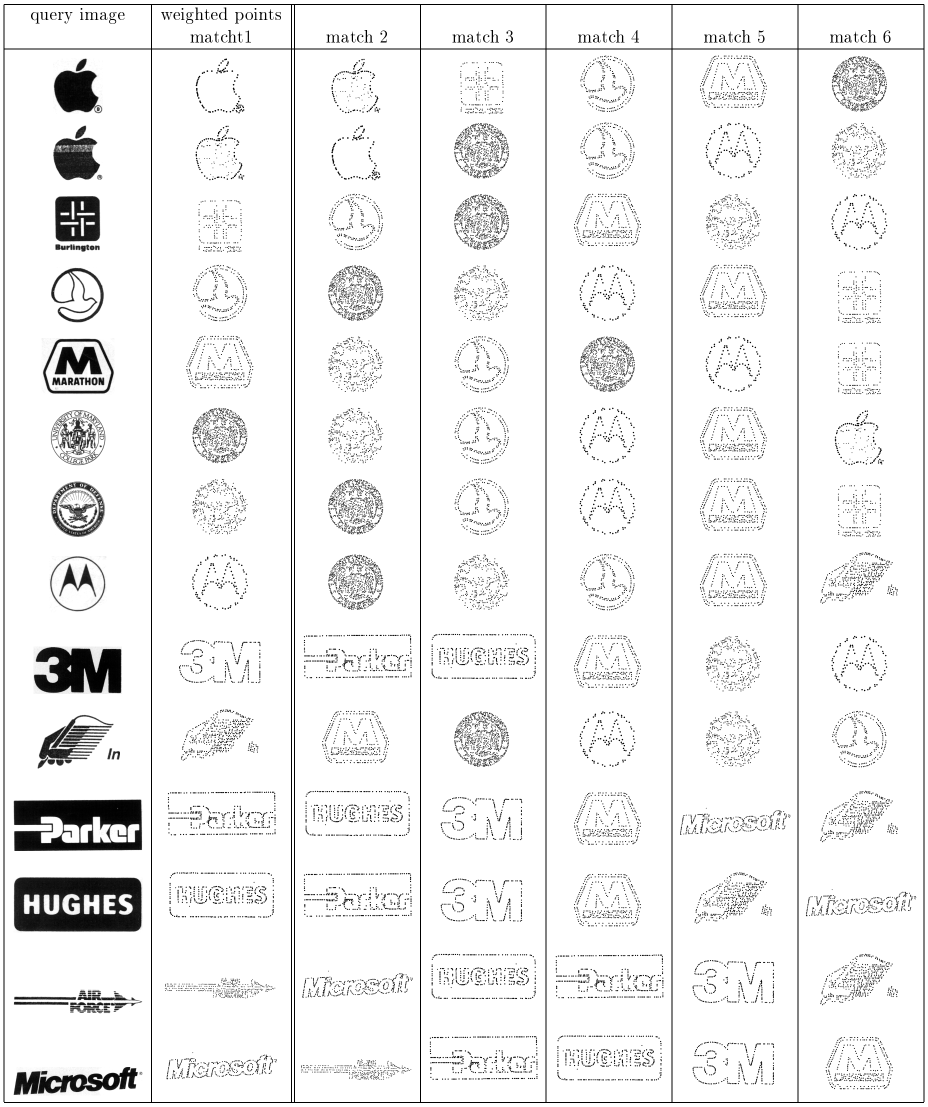


Figure 2.3: Top six matches for the 14 selected logos.

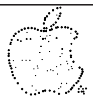
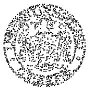
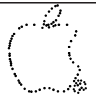
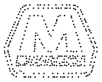
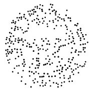
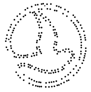
query points match1	match 2	match 3	match 4	match 5	match 6
					

Figure 2.4: Top six matches of the Apple logo using the EMD.

blurred) images, provided that our feature detectors are good enough to avoid most of the noise. Then, the higher the intensity of a point in the edge image, the more certain we are about its participation in the object's shape.

Last, we have compared our PTD results with those of EMD's: for the second Apple logo in the query column, the best six matches using EMD are shown in Figure 2.4. The reason behind this result is the partial matching property of EMD. The weighted point set that corresponds to the Apple logo has almost a hundred points while the one of the University of Maryland (match 2) has almost a thousand and much more total weight. When they are matched, almost every point in the Apple's set will coincide or be very close to a point in the Maryland's set and thus little weight is moved.

2.7.2 Fish Contours

Our second test data set comes from the SQUID database [93] which contains 1100 images of contours of marine animals, mostly fish. Each one of them is thinned to a one pixel width contour which is actually a simple polygon with its defining points ordered, in our case, clockwise.

Preprocessing

In order for our computations to be independent of orientation, position and scaling the following preprocessing steps are applied to the polygons. First, each polygon's center of mass is translated to the origin. Then the polygon is rotated around the origin in such a way that its major principal axis, identified with the PCA method [103], coincides with the x-axis. Next, each polygon is re-translated so that the center of its bounding box lies on the origin. The reason behind this translation lies in our choice of corners as feature points, see next section; since they probably lie on the bounding box, we would like them to play a bigger role in the translational normalization. Finally the polygon is scaled so that its bounding box has unit area.

Corner Extraction

Corners are points with high curvature and therefore significant for a polygon's shape. As already mentioned in the beginning of Section 2.7, we would like to select points with high curvature, and construct a weighted point set with the weights set to the actual curvature value. However, discrete curvature estimation is a difficult problem by itself. Several algorithms have been proposed, each one with its advantages and drawbacks [142].

We used a simple, fast and easy to implement corner detection algorithm by Rosenfeld and Weszka [109, 126]. The algorithm computes the value of a curvature measure for each point and then retains only points of local maxima as corners. A smoothing parameter determines the region of support, the curvature measure value for each point and, as a result, the number of output corners. We set this smoothing parameter to 0.045 for all input polygons. This gives us a small number of corners between 15 and 30 in most cases.

Here we should re-emphasize the fact that this curvature measure was used as a way to obtain weighted point sets and demonstrate the use of the proposed pseudo-metric and is not of the best interest of the application itself; we applied it on one particular collection where all the contours have been generated in the same way and on the same scale. Moreover, while computing the curvature measure, the same smoothing parameter was used for all the contours. Further discussion on the effects of this feature extraction choice on the matching results can be found in the next section.

Distance Computation and Results

We computed the distance for all pairs of all fish polygons. We should mention here that although the PCA method gives the two principal axis, it lacks any information about their direction, resulting thus in a two-way ambiguity for each of them. In our case this means that there are two configurations resulting in the same set of principal axis. This has been observed by other researchers as well [98]. We deal with this problem as follows. Let A and B be two normalized weighted point sets and $-B$ a version of B rotated by 180 degrees around the origin. Then, the distance between A and B equals $\min\{\mathrm{PTD}(A, B), \mathrm{PTD}(A, -B)\}$. Since the sets are small the computation of the distance takes no more than 0.05 seconds on a machine with the same configuration as the one used for the logo collection.

In Figure 2.5 the five best matches for example polygon queries are shown. The results are organized in pairs of rows where the first row shows the normalized polygon and the second the normalized weighted point set. Their orientation is the one of the two that gave the minimum distance according to the previous paragraph. The distance values, from left to right, for the shark query are as follows: 0.0, 0.082, 0.091, 0.102, 0.109. In contrast with the point sets in Figure 2.3, the weights here are spread over the whole range of values in (0, 1); this is shown by dots of variable width. Thus, discarding the weights, influences the results: more similar shapes can come after less similar ones.

The corner extraction method used here imposes a trade off between the number of output corner points and the relevance of the curvature measure value to the actual curvature value. Adjusting the smoothing parameter to give out more corners will result in a weight distribution quite different from the actual curvature distribution. Less corners, however, enclose less information

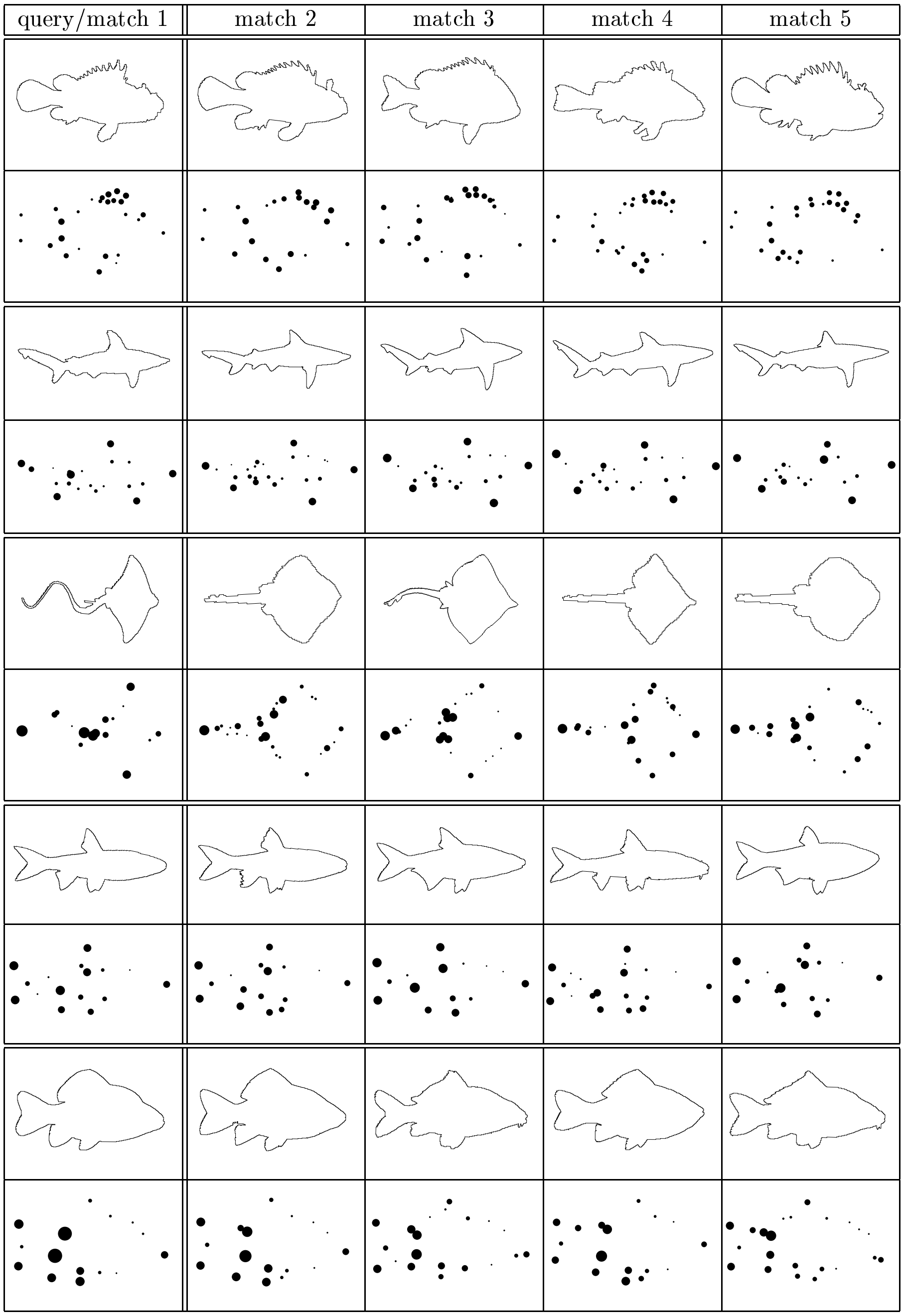


Figure 2.5: Top five matches for selected queries of fish polygons.

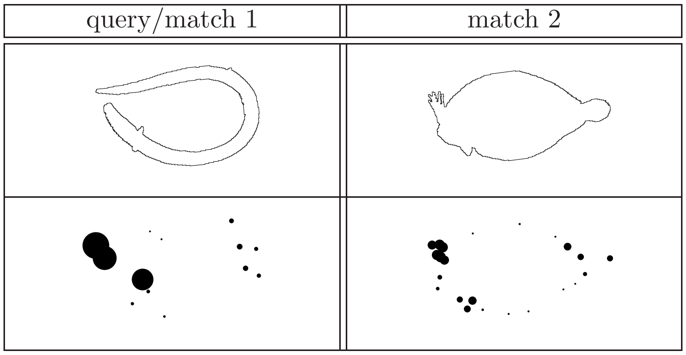


Figure 2.6: A counter-intuitive match.

on the polygon's shape. In general, a bad corner detector and curvature measure can give point sets with points in close positions and similar weight distributions which correspond to dissimilar shapes.

Figure 2.6 shows a counter-intuitive example. Here, the first column corresponds to the query and best match, and the second corresponds to the second best match. Note, for example, that both ends of the head and tail of the query eal fish are matched to the spikes on the head of the second fish, through their corresponding weighted points; these points carry, for both sets, the largest percentage of total weight. The points corresponding to the tail of the second fish are matched with points on the 'belly' of the eal fish, and there is almost no information, i.e points, on the remaining part of the latter.

The problem here lies on the representation of a contour with two dimensional weighted points that are close, as far as the ground distance is concerned, but far regarding their distance along the contour. However, an approach that takes into account distance information along the contour, as the one used in SQUID [94] does, does not always give better results. Figure 2.8 shows a bad match taken from SQUID; starting from left to right in each row, the top six matches are given, where the query and best match are found in the top left corner. Our results for the same query are given in Figure 2.7 and look much better. Apart from the identity match, the next three top matches fail to show up in SQUID's first eighteen results.

We have examined many other query results through a simple demo that can be also found in our web-site [66]. Although shape similarity is a highly subjective matter, generally argued similar shapes are given as a result for most queries. Overall the results are satisfactory and indicate that our pseudo-metric can be effectively used for shape retrieval.

Retrieval

We have experimented with two ways of retrieval, which can be found in our web-site as well: the full distance matrix approach and the vantage (reference)

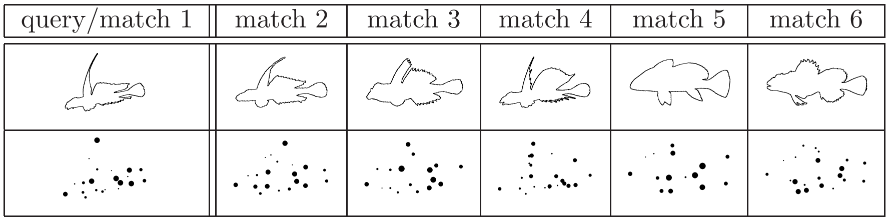
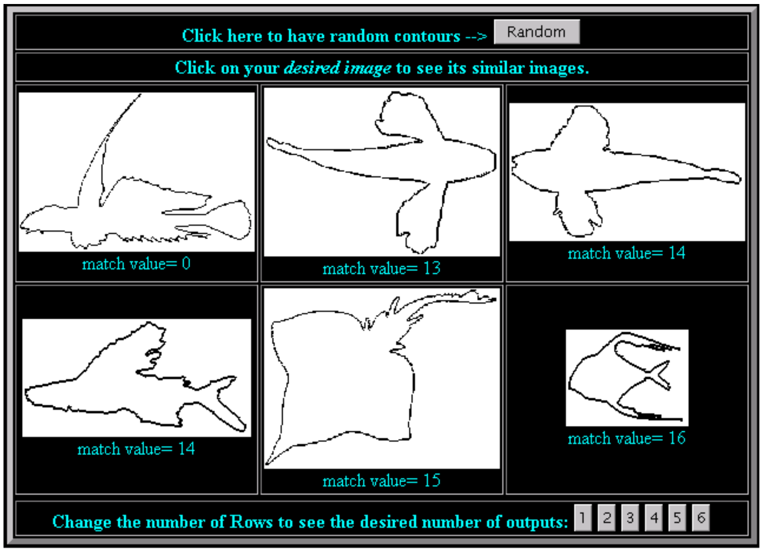


Figure 2.7: An example where PTD-based matching performs better than SQUID; compare to figure 2.8.



Click here to have random contours -->

Click on your *desired image* to see its similar images.

Change the number of Rows to see the desired number of outputs:

Figure 2.8: A counter-intuitive result in SQUID [93].

shapes method, described in Section 1.4. In the first all the distances are precomputed. This approach is of course fast, and gives all best matches that the user is interested in without false positives. A query with a new image can be dealt with by using linear search, where the distances from the new image to all the images in the database are computed. In our case, taking into account the average size of a weighted point set corresponding to a fish contour, this method will give a query time of about two to three minutes; clearly, the linear search method is not at all suitable for larger collections. An alternative to the latter is the vantage objects method which, taking advantage of the triangle inequality, is both scalable and fast. Given a query image and a positive constant ϵ, all images in the database that are within distance ϵ from the query are returned, together with some false positives. In our case, using for example ten vantage objects, the query time is not more than half a second, on the average.

2.7.3 Melodies

The representation of music as a weighted point set in a two-dimensional space has prevailed ever since the 13th century. Music scores are two-dimensional sets of notes (points) with time and pitch as coordinates; varying characteristics are associated with the notes by, for example, using different symbols for different note durations. Following this observation, we used the EMD and PTD to measure melodic similarity directly by comparing weighted point sets [129] instead of first transforming the music into one-dimensional abstract representations as most previous methods did. We tested our method on the RISM A/II[1] collection that contains about half a million musical incipits[2].

In general, each melody is transformed into a signature given as a weighted point set as follows; see Typke et al. [129] for a more detailed description. For each note one point is created; rests are encoded implicitly as the time spans that are not covered by points. In the database, the durations of notes and their positions within measures are specified using divisions of a quarter note, in a way similar to the MIDI format. With every melody, the number of divisions per quarter note is stored. The time coordinate of a note is the sum of lengths of measures preceding the note plus the note's position within its measure, divided by the number of divisions for a quarter note. In order to skip leading rests, all notes are shifted so that the first note starts at time 0. The pitch coordinate of a note is determined by its pitch in the Walter Hewlett's (1992) Base-40 notation. Finally, a note's weight simply reflects its duration. Although the experiments were performed with weights that depend only on note durations, additional weight components, like stress weight and note number weight could lead to additional desirable effects.

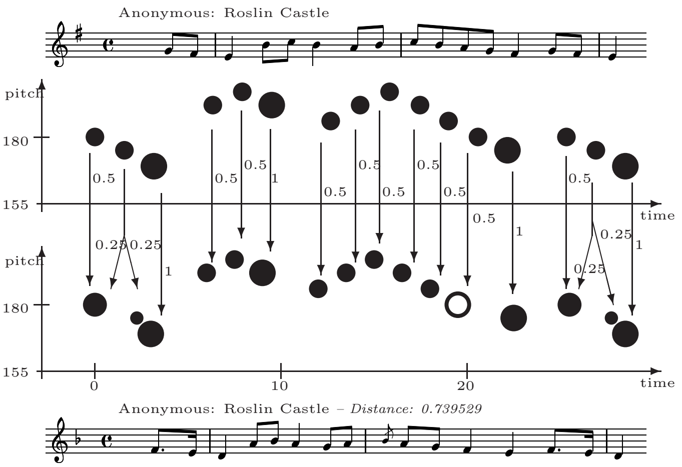


Figure 2.9: An example of two melodies, their signatures and the weight flow between them with the EMD.

---

[1] Répertoire International des Sources Musicales, 1995-2002.

[2] An incipit is the beginning of a piece, typically about 20 notes long.

Before applying the EMD or PTD to a pair of sets, several adjustments are made to the coordinates and weights of the points. For example, one of the two sets is transposed in the pitch axis so that both sets have equal weighted average pitch. Figure 2.9 gives an example of two melodies, their signatures after the necessary adjustments and the weight flow between them when the EMD is used.

Extensive experiments showed that this method has significant advantages over previous methods. For example, compared to the "Frankfurt Experience" and the "Harvard Experience" of sorting RISM incipits described by Howard [77], we were able to identify about twice the percentage of melodies by anonymous composers and group together 76% instead of 46% of the known occurrences of a tune called "Roslin Castle" [129]. The experiments also showed that the partial matching provided by the EMD does not always make musical sense; the authors note that in order to be able to find motifs and themes within complete pieces, it might be necessary to 'segment' the pieces into small chunks and then use the EMD or PTD to compare the chunks individually. For a discussion about EMD, PTD and partial matching, see Section 2.8. Finally, by exploiting PTD's triangle inequality and using vantage objects, an interactive online search engine was build that searches all melodies in the test database [128]. Since this database is large, the computation of the distance of a query melody to every record is prohibitive. Indeed, our approach reduced the query running time approximately 70 minutes to 9 seconds, without altering the result considerably.

2.7.4 Polyhedral Models

Recent improvements in laser scanning technology, 3D visualization and modeling, necessitate the development of tools supporting the automatic search for 3D objects in archives. The World Wide Web provides access to thousands of such objects mostly in virtual reality modeling language (VRML) format which represents 3D models as meshes. Tangelder and Veltkamp [125] described a new geometric approach to 3D shape comparison and retrieval for arbitrary objects represented by a polyhedral mesh. In contrast with existing approaches, their approach uses the overall relative spatial location by representing the 3D shape as a weighted point set, without taking the connectivity relations into account. Below, we describe briefly the way these weighted point sets are constructed.

To compare two objects independently of orientation, position and scaling, PCA is first applied to bring the objects in a standard pose. Each object is enclosed by a 3D grid and a salient point is created for each non-empty grid cell. The following three methods to obtain in each grid cell a salient point and a weight are described and compared: (1) choosing the vertex in the cell with the highest Gaussian curvature, and choosing as weight a measure for that curvature, (2) choosing the area-weighted mean of the vertices in the cell, and choosing as weight a measure denoting the normal variation of the facets in the cell and (3) choosing the center of mass of all vertices in the cell, and the weight equal to one.

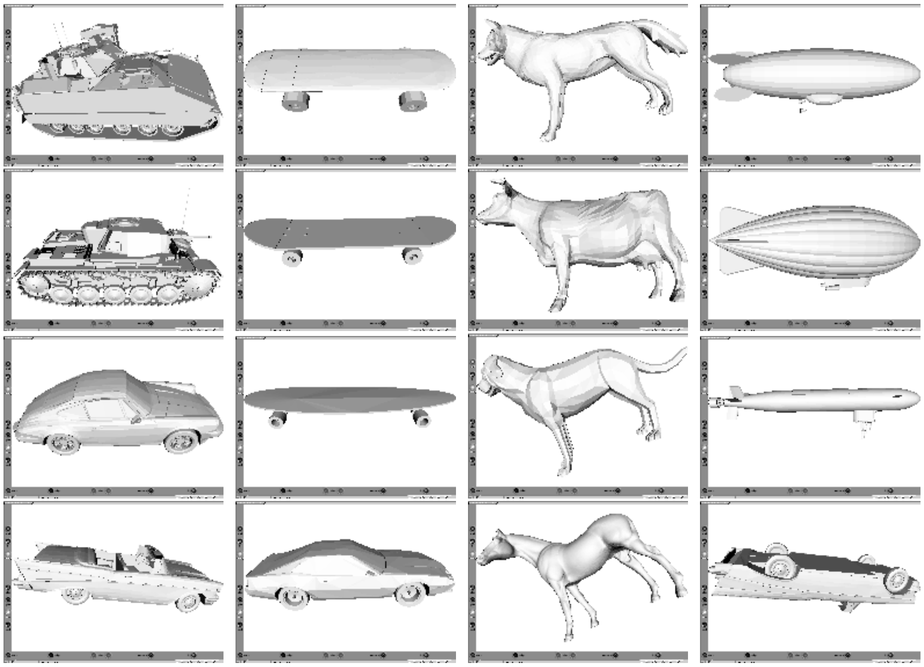


Figure 2.10: Query results using the Gaussian curvature method for the Princeton database. Each column illustrates a query. The top row shows the query objects, the second row the nearest neighbor, the third and fourth row show the second and third nearest neighbor, respectively.

Using the PTD for comparing two weighted point sets, experiments were performed on two different test databases: a database from Princeton consisting of 133 models classified into functional categories such as mugs, cars and boats, and the Utrecht database consisting of 512 models including 376 models of air planes classified into shape categories [124]. The results of the experiments on the Princeton database showed the method to have better performance than the best of the shape distribution based methods given by Osada et al. [101] on the same database. The improvement goes up to as much as 14% when the Gaussian curvature method is used. Example query results of the latter method are shown in Figure 2.10. In the experiments, computing the PTD takes on average about 2 seconds. Using the triangle inequality with the vantage objects method, reduces the query time to the database of 512 models from 17 minutes to 15 seconds on the average when 8 vantage objects are used.

2.8 Concluding remarks

This chapter is about weighted point set matching using similarity measures. A pseudometric, the PTD, with additional properties that make it suitable

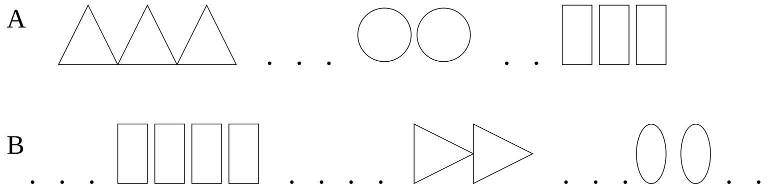


Figure 2.11: An example of partial matching problem that calls for the LCP solution.

for shape matching and retrieval was presented. This distance is based on mass transportation. It's effectiveness has been established via extensive experiments on a range of different test data sets. We end this chapter with a few words on partial matching.

As mentioned in section 2.4.2, the EMD performs partial matching when applied on two sets with unequal total weights, by matching only the smallest of the two weights. Obviously this is not true for the EMD on equal total weight sets or the PTD in general. Searching for a pattern A in a pattern B requires transformations to be taken into account as well. That is, the transformation that minimizes the distance has to be found; this is a quite difficult problem that is dealt with in the next chapter. An algorithm that minimizes the EMD under a given group of transformations was given in [45]. Unfortunately, this algorithm can be trapped in local minima and increases the computation time considerably. However, the optimal transformation does not always produce meaningful results. Consider, for example, the two polygonal patterns in figure 2.11; the dots in the image represent sub-patterns with arbitrary shapes.

We assume that both patterns will be reduced to weighted point sets that outline their corresponding shapes. Here, we would like to be able to identify similar sub-patterns of both sets by finding a suitable transformation and flow that establish the matching i.e the correspondence among individual points. Even if the two sets have unequal total weights, minimization of the EMD under rigid motions, for instance, will not necessarily match the triangles of A to their rotated versions in B. In general, the outcome will depend also on the the weights of individual points and the shape of the missing components, i.e. the ones represented with dots.

The above issues call for the more general partial pattern matching formulation of the Largest Common Pattern problem that was presented in Section 1.5. It would be interesting to try to devise polynomial time LCP algorithms for the EMD or PTD, prove hardness results if this is not possible and look for heuristics that work well in practice.

Chapter 3

Minimizing the Earth Mover's Distance under Transformations

3.1 Introduction

This chapter deals with the problem of minimizing the Earth Mover's Distance under transformations. As already mentioned in the previous chapter, this task is important for measuring similarity independently of transformations and subsequently for finding a given pattern into a larger one. We are interested in transformations that change only the position of the points, not their weights; in particular we focus on translations and rigid motions. We present polynomial-time $(1+\epsilon)$ and $(2+\epsilon)$-approximation algorithms for the minimum Euclidean EMD of two weighted point sets in the plane under translations and rigid motions.

We start with some notation and definitions that will be used throughout the chapter. Then we give a summary of previous work and our results. The algorithms are presented in the rest of the chapter.

3.1.1 Preliminaries

Let $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$ be two planar weighted point sets with $m \leq n$. In a slightly different notation to the one used in the previous chapter, a weighted point $a_i \in A$ is defined as $a_i = \{(x_{a_i}, y_{a_i}), w_i\}, i = 1, \ldots, m$, where $(x_{a_i}, y_{a_i}) \in \mathbb{R}^2$ are the point's coordinates and $w_i \in \mathbb{R}^+ \cup \{0\}$ is its weight. A weighted point $b_j \in B$ is defined similarly as $b_j = \{(x_{b_j}, y_{b_j}), u_j\}, j = 1, \ldots, n$. Let $W = \sum_{i=1}^{m} w_i$ and $U = \sum_{j=1}^{n} u_j$ be the total weight, or simply weight, of A and B respectively.

We consider B to be fixed, while A can be translated and/or rotated relative to B. We assume some initial positions for both sets, denoted simply by A and

B. We denote by R_θ a rotation about the origin by some angle $\theta \in [0, 2\pi)$ and by $T_{\vec{t}}$ a translation by some $\vec{t} \in \mathbb{R}^2$. Let $\mathcal{I}$ be the set of all possible rigid motions in the plane. Any rigid motion $I \in \mathcal{I}$ can be uniquely defined as a translation followed by a rotation, that is, $I = I_{\vec{t},\theta} = R_\theta \circ T_{\vec{t}}$, for some $\theta \in [0, 2\pi)$ and $\vec{t} \in \mathbb{R}^2$. In general, transformed versions of A are denoted by $A(\vec{t}, \theta) = \{a_1(\vec{t}, \theta), \dots, a_m(\vec{t}, \theta)\}$ for some $I_{\vec{t},\theta} \in \mathcal{I}$. For simplicity, rotated only versions of A are denoted by $A(\theta) = \{a_1(\theta), \dots, a_m(\theta)\}$. Similarly, translated only versions of A are denoted by $A(\vec{t}) = \{a_1(\vec{t}), \dots, a_m(\vec{t})\}$.

The EMD between $A(\vec{t}, \theta)$ and B, as $\vec{t}, \theta$ vary, is a function EMD : $\mathcal{I} \to \mathbb{R}^+ \cup \{0\}$ defined as

$$\mathrm{EMD}(\vec{t}, \theta) = \min_{F \in \mathcal{F}(A,B)} \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} d_{ij}(\vec{t}, \theta)}{\min\{W, U\}},$$

where $d_{ij}(\vec{t}, \theta)$ is the distance of $a_i(\vec{t}, \theta)$ to b_j, and $F = \{f_{ij}\} \in \mathcal{F}(A, B)$ with $\mathcal{F}(A, B)$ being the set of all feasible flows between A and B defined by the usual constraints:

1. $f_{ij} \geq 0, i = 1, \dots, m, j = 1, \dots, n$
2. $\sum_{j=1}^n f_{ij} \leq w_i, i = 1, \dots, m$
3. $\sum_{i=1}^m f_{ij} \leq u_j, j = 1, \dots, n$
4. $\sum_{i=1}^m \sum_{j=1}^n f_{ij} = \min\{W, U\}$

In case that $\vec{t}$ or θ or both are constant, we simply write EMD(θ), EMD($\vec{t}$) and EMD respectively. Unless stated otherwise, we deal with the Euclidean or L_2-based EMD where d_{ij} is given by the L_2-norm. We study the following problem:

Given two weighted point sets A, B compute a rigid motion $I_{\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}}$ that minimizes EMD$(\vec{t}, \theta)$.

Note that the objective function is not linear in $\vec{t}$ and θ but it is still linear in F. Thus, the minimum EMD occurs at some vertex of the convex polytope $\mathcal{F}(A, B)$. This suggests the following straightforward algorithm: for every vertex $F = \{f_{ij}\}$ of $\mathcal{F}(A, B)$ compute the transformation that minimizes $\sum_{i=1}^m \sum_{j=1}^n f_{ij} d_{ij}(\vec{t}, \theta)$. Assuming that the latter problem can be solved optimally, the algorithm is guaranteed to find a transformation that minimizes the EMD. However, the number of the vertices of $\mathcal{F}(A, B)$ can be exponential in m and n. Moreover, as we shall see in the next section, the problem of finding the optimum transformation for a given flow is not trivial.

3.1.2 Relevant work

The problem of minimizing the EMD under transformations was first studied by Cohen and Guibas [45]. They gave simple algorithms for finding the minimum EMD under translations for the following two special cases: one-dimensional, equal weight sets with L_1, and k-dimensional equal weight sets with L_2^2. The latter case is quite restrictive since, in general, the sets need not have the same weight, and the use of square Euclidean distance is statistically less robust than Euclidean distance [86]. Currently, no (practical) algorithm that computes an optimal translation and/or rotation is known for the Euclidean EMD.

Based on an idea similar to the Iterative Closest Point Iteration [26], Cohen and Guibas presented also a Flow-Transformation iteration which alternates between finding the optimum flow for a given transformation, and the optimum transformation for a given flow. They showed that this iterative procedure converges, but not necessarily to a global optimum. The task of finding the optimum flow for a given transformation is just the transportation problem we have seen in the previous chapter; see also Section 3.2. We discuss the second task below.

Finding the optimal transformation for a given flow. Recall that a given flow $F = \{f_{ij}\}$ between two sets A and B induces in general a many-to-many correspondence between the points of A and the points of B. The problem of finding the optimal transformation for a given flow is actually the problem of finding a transformation of the points in set A that minimizes the sum of weighted distances to the corresponding points in the other set. This problem can be solved efficiently for the case of squared Euclidean distance and translations, rigid motions and similarities [130], linear and affine transformations [134].

For the Euclidean distance and translations, the problem reduces to the Euclidean Minisum problem or the Fermat-Weber problem [35], where one wants to find a point that minimizes the sum of weighted distances to a set of given points. In 1937 Weiszfeld [140] proposed an iterative procedure to solve this problem. Unfortunately, this procedure is not guaranteed to find a location that achieves the global minimum. Currently, no exact solution to the Fermat-Weber problem is known even in the real RAM model of computation [21, 27]. However, Bose et al. [27] gave an $O(n \log n)$ time deterministic and $O(n)$ time randomized $(1+\epsilon)$-approximation algorithm in any fixed dimension.

We conclude that even if the exponential time algorithm for finding the minimum Euclidean EMD under translations is used, we only know how to achieve a $(1+\epsilon)$-approximation of the optimum. However, as we will show, such an approximation can be also achieved in polynomial time.

3.1.3 Results

In this chapter, we give polynomial-time $(1+\epsilon)$ and $(2+\epsilon)$-approximation algorithms for the minimum Euclidean EMD under translations and rigid motions; the algorithms for translations are given in Section 3.4, while the algorithms for rigid motions can be found in Section 3.5. In the general case where the sets have unequal total weights we compute a $(1+\epsilon)$-approximation in $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$ time for translations and a $(2+\epsilon)$-approximation in $O((n^4m^2/\epsilon^4)\log^2(n/\epsilon))$ time for rigid motions. When the sets have equal total weights, the respective running times decrease to $O((n^2/\epsilon^4)\log^2(n/\epsilon))$ and $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$.

The case where $w_i = u_j = 1, i = 1,\dots,m, j = 1,\dots,n$, deserves special attention: the integer solutions property of the minimum cost flow problem and the fact that $0 \le f_{ij} \le 1$ imply that there is a minimum cost flow from A to B that results in a *partial assignment* between A and B, that is, a perfect matching between A and a subset of B; when $n = m$ the problem is simply referred to as the *assignment* problem. We show how to compute a $(1+\epsilon)$-approximation of the minimum cost assignment under translations and rigid motions in $O((n^{3/2}/\epsilon^{7/2})\log^5 n)$ and $O((n^{7/2}/\epsilon^{9/2})\log^6 n)$ time respectively. Finally, we give probabilistic $(1+\epsilon)$-approximations of the minimum cost partial assignment under translations in $O((n^3/\epsilon^4)\log^2(n/\epsilon)\log n)$ time and under rigid motions in $O((n^4m/\epsilon^5)\log^2(n/\epsilon)\log n\log m)$ time; both algorithms succeed with high probability.

In Section 3.3, we give two simple lower bounds on the EMD that are vital to our approximation algorithms. These algorithms need to compute the EMD for a given transformation. Computing the EMD exactly is expensive, and unnecessary since we opt for approximations for our original problem. We begin by showing how to get a $(1+\epsilon)$-approximation of the EMD in almost quadratic time.

3.2 Approximating the EMD

The transportation problem is a special version of the minimum cost network flow problem, the uncapacitated version of which can be formulated as follows [5]: Let $G(V,E)$ be a directed graph (network) defined by a set V of vertices and a set E of directed edges. Each edge $(v_i,v_j) \in E$ connecting $v_i \in V$ to $v_j \in V$ has an associated cost d_{ij} that denotes the cost per unit flow on that edge. Each edge (v_i,v_j) has also an associated lower bound l_{ij} that denotes the minimum amount of flow on the edge. Each vertex v_i has a weight $w_i \in \mathbb{R}$ representing its supply/demand. If $w_i > 0$, v_i is a supply vertex; if $w_i < 0$, v_i is a demand vertex with a demand of $-w_i$; and if $w_i = 0$, v_i is a transhipment vertex. Without loss of generality we assume that the total demand is at least as much as the total supply, i.e., $\sum_{i=1}^{|V|} w_i \le 0$. Let f_{ij} be the variable representing the amount of flow from v_i to v_j on the edge (v_i,v_j). The objective is to

$$\text{minimize} \sum_{i,j:(v_i,v_j)\in E} f_{ij}d_{ij}$$

subject to

$$\sum_{j:(v_i,v_j)\in E} f_{ij} - \sum_{j:(v_j,v_i)\in E} f_{ji} = w_i, \forall i : w_i \ge 0,$$

$$\sum_{j:(v_i,v_j)\in E} f_{ij} - \sum_{j:(v_j,v_i)\in E} f_{ji} \geq w_i, \forall i : w_i < 0,$$

$$l_{ij} \leq f_{ij}, \forall i,j : (v_i,v_j) \in E.$$

Clearly, the Euclidean EMD formulation fits into the above model where G is a directed complete bipartite graph with $V = A \cup B$ and $E = \{(a_i,b_j) : a_i \in A, b_j \in B\}$, there are no transhipment vertices, the edges are directed from supply vertices to demand vertices, d_{ij} is the Euclidean distance of a_i to b_j, and $l_{ij} = 0$.

Currently, the fastest strongly polynomial-time algorithm for the minimum cost flow problem on $G(V,E)$ is due to Orlin [99], and runs in $O((|E| \log |V|)(|E| + |V| \log |V|))$ time. Several weakly polynomial-time algorithms exist, that assume integer edge costs and/or integer supplies and demands [5]. For example, the algorithm by Goldberg and Tarjan [69] assumes integer edge costs and runs in $O(|V||E| \log(|V|^2/|E|) \log(|V|C))$ time, where C is the largest edge cost.

For the general transportation problem, Gabow and Tarjan [63] presented an algorithm that assumes integer edge costs and integer supplies and demands; the algorithm achieves a time bound of $O((\min\{\sqrt{W}, |V|\}|E| + W \log W) \log(|V|C))$ where W is the sum of supplies and demands. For the transportation problem in the plane, Atkinson and Vaidya [20] presented an algorithm that assumes integer supplies and demands and runs in $O(|V|^{2.5} \log(|V|) \log W)$ time, where W is the largest supply or demand; an improved version of this algorithm, by Agarwal et al. [3], runs in $|V|^{2+\delta} \log(|V|) \log W)$ time, for any fixed $\delta > 0$.

Consider the complete bipartite graph $G(V,E)$ with $V = A \cup B$ and $E = \{(a_i,b_j) : a_i \in A, b_j \in B\}$. Using the algorithm of Callahan and Kosaraju [33], we can construct, in $O((n+m)\log(n+m) + ((n+m)/\epsilon^2)\log(1/\epsilon))$ time, a linear size $(1+\epsilon)$-spanner G_s, i.e., a graph $G_s(V,E_s)$ with $|E_s| = O((n+m)/\epsilon)$ such that the shortest path between any two points in G_s is at most $(1+\epsilon)$ times the Euclidean distance of the points. As we will see below, running the algorithm of Orlin on G_s produces an approximate value EMD_s such that $\text{EMD} \leq \text{EMD}_s \leq (1+\epsilon)\text{EMD}$. For convenience, this simple procedure is referred to as ApxEMD(A,B,ϵ) and given in Figure 3.1.

ApxEMD(A,B,ϵ):

1. Let $G(V,E)$ be a complete bipartite graph with $V = A \cup B$ and $E = \{(a_i,b_j) : a_i \in A, b_j \in B\}$.
2. Construct a $(1+\epsilon)$-spanner $G_s(V,E_s)$ using the algorithm of Callahan and Kosaraju [33], such that $|E_s| = O(n/\epsilon)$.
3. Find a minimum cost flow on G_s using the algorithm by Orlin [99], and report the cost.

Figure 3.1: Algorithm ApxEMD(A,B,ϵ).

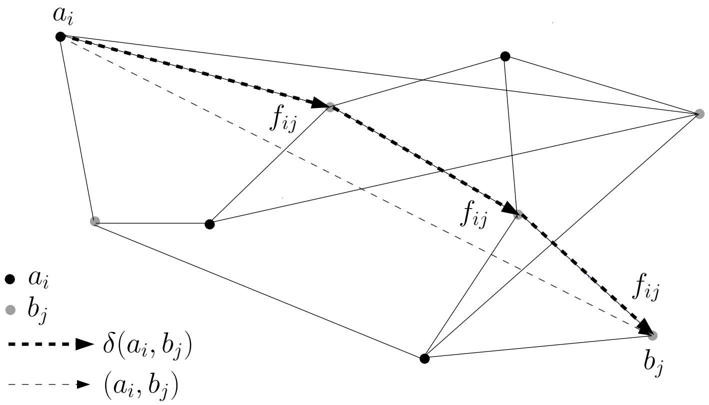


Figure 3.2: Two point sets $A = \{a_i\}, B = \{b_j\}$, a spanner G_s on $A \cup B$, and an optimal flow f_{ij} sent over $\delta(a_i, b_j)$ in G_s.

Lemma 3.1 *For any given* $\epsilon > 0$, $\textsc{ApxEMD}(A, B, \epsilon)$ *computes a value* EMD_s *such that* $\mathrm{EMD} \leq \mathrm{EMD}_s \leq (1+\epsilon)\mathrm{EMD}$ *in* $O((n^2/\epsilon^2)\log^2(n/\epsilon))$ *time.*

Proof: First, we convert G_s into a directed graph as follows: each edge $(a_i, b_j) \in E_s$ is substituted by two directed edges (a_i, b_j) and (b_j, a_i) both with cost d_{ij}. For any pair of vertices a_i, b_j, any shortest path from a_i to b_j in G_s is now directed; let $\delta(a_i, b_j)$ be such a path and $d(a_i, b_j)$ be its length. Note that since G_s is not necessarily bipartite, $\delta(a_i, b_j)$ might contain one or more other vertices of A and/or B.

Let $\{f_{ij}\}$ be a minimum cost flow on G. In G_s, we choose to send an amount of f_{ij} from a_i to b_j over $\delta(a_i, b_j)$; see Figure 3.2 for an illustration. Consider a vertex $v \in V$ that is an intermediate node in $\delta(a_i, b_j)$. Then, f_{ij} enters and leaves v without affecting its total surplus or deficit, that is, the incoming flow minus the outcoming flow. Since $\{f_{ij}\}$ is a feasible flow on G, the flow induced by the above assignment is a feasible flow on G_s. Since $d(a_i, b_j) \leq (1+\epsilon)d_{ij}$ we have

$$\mathrm{EMD}_s \leq \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} d(a_i, b_j)}{\min\{W, U\}} \leq \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}(1+\epsilon)d_{ij}}{\min\{W, U\}} = (1+\epsilon)\mathrm{EMD}.$$

Moreover, any minimum cost flow on G_s can be decomposed into flows along paths from supply vertices to demand vertices in G_s and thereby defines some feasible flow on G. Hence, since $d_{ij} \leq d(a_i, b_j)$, we have that $\mathrm{EMD} \leq \mathrm{EMD}_s$. Constructing G_s takes $O((n+m)\log(n+m) + ((n+m)/\epsilon^2)\log(1/\epsilon)) = O(n \log n + (n/\epsilon^2)\log(1/\epsilon))$ time. Since $|E_s| = O((n+m)/\epsilon) = O(n/\epsilon)$, computing a minimum cost flow on G_s takes $O(((n/\epsilon)\log(n+m))((n/\epsilon) + (n+m)\log(n+m))) = O(((n/\epsilon)\log n)((n/\epsilon) + n \log n))$ time. In total the algorithm takes $O(n \log n + (n/\epsilon^2)\log(1/\epsilon) + ((n/\epsilon)\log n)((n/\epsilon) + n\log n)) = O((n^2/\epsilon^2)\log^2(n/\epsilon))$ time. $\square$

For the assignment or minimum cost Euclidean bipartite matching problem in the plane, Varadarajan and Agarwal [133] presented an algorithm that finds a matching with cost at most $(1+\epsilon)$ times the cost of an optimal matching in $O((n/\epsilon)^{3/2}\log^5 n)$ time; we refer to this algorithm as $\textsc{ApxMATCH}(A,B,\epsilon)$.

Theorem 3.1 *[133, Theorem 3.1] Let A and B be two sets of points in the plane with $|A|=|B|=n$. For any given $\epsilon>0$, a perfect matching between A and B with cost at most $(1+\epsilon)$ times that of an optimal perfect matching can be computed in $O((n/\epsilon)^{3/2}\log^5 n)$ time.*

3.3 Lower bounds on the EMD

We give two lower bounds on the EMD, that depend on the distance between two points that belong to — or can be easily computed from — $A\cup B$. As such, these lower bounds direct our search for the optimal transformation; see next section.

The following simple lower bound comes directly from the definition of the EMD.

Observation 3.1 *Given two weighted point sets A and B,* $\mathrm{EMD}\geq \min_{i,j} d_{ij}$.

Proof: Let $\{f_{ij}\}$ be an optimal flow between A and B. We have

$$\mathrm{EMD}=\frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}d_{ij}}{\min\{W,U\}}\geq\frac{\min_{ij}d_{ij}\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}}{\min\{W,U\}}=\min_{ij}d_{ij},$$

since $\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}=\min\{W,U\}$. ⊔⊓

The next lower bound is due to Rubner et al. [114], and applies to sets with equal weights. A practical use of it was already given in Section 2.6. The lower bound is based on the notion of the *center of mass* of a weighted point set.

The center of mass $C(A)$ of a planar weighted point set $A=\{(x_{a_i},y_{a_i}),w_i\}$, $i=1,\ldots,m$, is defined as

$$C(A)=\frac{\sum_{i=1}^{m} w_i\cdot(x_{a_i},y_{a_i})}{\sum_{i=1}^{m} w_i}.$$

Theorem 3.2 *[114] Let A and B be two weighted point sets with equal weights. Then* $\mathrm{EMD}\geq d(C(A),C(B))$.

As Klein and Veltkamp [87] noted, this lower bound implies that the center of mass is a reference point [6] for equal weight sets, resulting in a trivial 2-approximation algorithm for the minimum EMD under translations; see next section for details.

3.4 Approximation algorithms for translations

We denote by $\vec{t}_{i\to j}$ the translation which matches a_i and b_j; we call such a translation a *point-to-point* translation. Observation 3.1 implies that the point-to-point translation that is closest to $\vec{t}_{\rm opt}$ gives a 2-approximation of $\mathrm{EMD}(\vec{t}_{\rm opt})$. Hence, we have the following:

Lemma 3.2 *Given two weighted point sets A and B, it holds that* $\mathrm{EMD}(\vec{t}_{\rm opt}) \leq \min_{i,j} \mathrm{EMD}(\vec{t}_{i\to j}) \leq 2\mathrm{EMD}(\vec{t}_{\rm opt})$.

Proof: Clearly, $\mathrm{EMD}(\vec{t}_{\rm opt}) \leq \min_{i,j} \mathrm{EMD}(\vec{t}_{i\to j})$. Next, consider the optimal position $A(\vec{t}_{\rm opt})$ of A and an optimal flow $\{f_{ij}\}$ between $A(\vec{t}_{\rm opt})$ and B. Consider also the distance $d_{ij}(\vec{t}_{\rm opt})$ for every pair of points $a_i(\vec{t}_{\rm opt})$, b_j and let $d_{i_0j_0}(\vec{t}_{\rm opt})$ be the smallest of all these distances. Assume that we translate $A(\vec{t}_{\rm opt})$ to the position $A(\vec{t}_{i_0\to j_0})$. Then $d_{i_0j_0}(\vec{t}_{i_0\to j_0})$ equals to 0 and, since $d_{i_0j_0}(\vec{t}_{\rm opt}) \leq d_{ij}(\vec{t}_{\rm opt})$, we have that

$$d_{ij}(\vec{t}_{i_0\to j_0}) \leq d_{ij}(\vec{t}_{\rm opt}) + d_{i_0j_0}(\vec{t}_{\rm opt}) \leq 2d_{ij}(\vec{t}_{\rm opt}),$$

for every $i = 1,\ldots,m$ and $j = 1,\ldots,n$. Hence, we have

$$\begin{aligned}
\min_{i,j} \mathrm{EMD}(\vec{t}_{i\to j}) &\leq \mathrm{EMD}(\vec{t}_{i_0\to j_0}) \\
&\leq \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} d_{ij}(\vec{t}_{i_0\to j_0})}{\min\{W,U\}} \\
&\leq \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} 2d_{ij}(\vec{t}_{\rm opt})}{\min\{W,U\}} \\
&= 2\mathrm{EMD}(\vec{t}_{\rm opt}).
\end{aligned}$$

$\square$

According to Observation 3.1, the point-to-point translation which is closest to $\vec{t}_{\rm opt}$ can be at most $\mathrm{EMD}(\vec{t}_{\rm opt})$ away from $\vec{t}_{\rm opt}$. This bound is crucial for the $(1+\epsilon)$-approximation algorithm given in Figure 3.3. Using a uniform square grid of suitable size we compute the EMD for a limited number of grid translations within a small neighborhood – of size $\mathrm{EMD}(\vec{t}_{\rm opt})$ – of every point-to-point translation. Note that we do not know $\mathrm{EMD}(\vec{t}_{\rm opt})$ but we can compute $\min_{i,j} \mathrm{EMD}(\vec{t}_{i\to j})$ which, according to Lemma 3.2, approximates $\mathrm{EMD}(\vec{t}_{\rm opt})$ well-enough. In order to save time, rather than computing EMD exactly, we will approximate it using the procedure ApxEMD.

Theorem 3.3 *Let $A = \{a_1,\ldots,a_m\}$ and $B = \{b_1,\ldots,b_n\}$ be two weighted point sets in the plane with $m \leq n$. For any given $\epsilon > 0$,* Translation(A,B,ϵ) *computes a translation $\vec{t}_{\rm apx}$ such that $\mathrm{EMD}(\vec{t}_{\rm apx}) \leq (1+\epsilon)\mathrm{EMD}(\vec{t}_{\rm opt})$ in $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$ time.*

```
TRANSLATION$(A, B, \epsilon)$:
  1. Let $\alpha = \min_{i,j}$ APXEMD$(A(\vec{t}_{i\to j}), B, 1)$ and let $G$ be a uniform square grid
     of spacing $c\epsilon\alpha$, where $c = 1/\sqrt{72}$.
  2. For each pair of points $a_i \in A$ and $b_j \in B$ do:
      (a) Place a disk $D$ of radius $\alpha$ around $\vec{t}_{i\to j}$.
      (b) For every grid point $\vec{t}_{\mathrm{g}}$ of any cell of $G$ that intersects $D$ compute a
          value $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$ = APXEMD$(A(\vec{t}_{\mathrm{g}}), B, \epsilon/3)$.
  3. Report the grid point $\vec{t}_{\mathrm{apx}}$ that minimizes $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$.
```

Figure 3.3: Algorithm TRANSLATION(A, B, ϵ).

Proof: According to Lemma 3.2

$$\mathrm{EMD}(\vec{t}_{\mathrm{opt}}) \leq \min_{i,j} \mathrm{EMD}(\vec{t}_{i\to j}) \leq 2\mathrm{EMD}(\vec{t}_{\mathrm{opt}}).$$

From Lemma 3.1 we have that

$$\mathrm{EMD}(\vec{t}_{i\to j}) \leq \textsc{ApxEMD}(A(\vec{t}_{i\to j}), B, 1) \leq 2\mathrm{EMD}(\vec{t}_{i\to j}).$$

Hence, since $\alpha = \min_{i,j} \textsc{ApxEMD}(A(\vec{t}_{i\to j}), B, 1)$ we have that

$$\mathrm{EMD}(\vec{t}_{\mathrm{opt}}) \leq \alpha \leq 4\mathrm{EMD}(\vec{t}_{\mathrm{opt}}).$$

Also, according to Observation 3.1, there is a point-to-point translation $\vec{t}_{i\to j}$ for which $|\vec{t}_{i\to j} - \vec{t}_{\mathrm{opt}}| \leq \mathrm{EMD}(\vec{t}_{\mathrm{opt}}) \leq \alpha$. Algorithm TRANSLATION will, at some stage, consider the α-neighborhood of such a translation, and thus, compute a value $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$ for some grid point $\vec{t}_{\mathrm{g}}$ for which

$$|\vec{t}_{\mathrm{g}} - \vec{t}_{\mathrm{opt}}| \leq \sqrt{2(\epsilon\alpha/\sqrt{72})^2}/2 \leq (\epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}),$$

and thus $d_{ij}(\vec{t}_{\mathrm{g}}) \leq d_{ij}(\vec{t}_{\mathrm{opt}}) + (\epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{opt}})$; see Figure 3.4. Assuming that $\{f_{ij}\}$ is the optimal flow at $\vec{t}_{\mathrm{opt}}$, and similarly to the proof of Lemma 3.2, we have

$$\begin{aligned}
\mathrm{EMD}(\vec{t}_{\mathrm{g}}) &\leq \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} d_{ij}(\vec{t}_{\mathrm{g}})}{\min\{W, U\}} \\
&\leq \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} (d_{ij}(\vec{t}_{\mathrm{opt}}) + (\epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}))}{\min\{W, U\}} \\
&= (1 + \epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}).
\end{aligned}$$

From Lemma 3.1 we have that

$$\mathrm{EMD}(\vec{t}_{\mathrm{g}}) \leq \widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}}) \leq (1 + \epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{g}}).$$

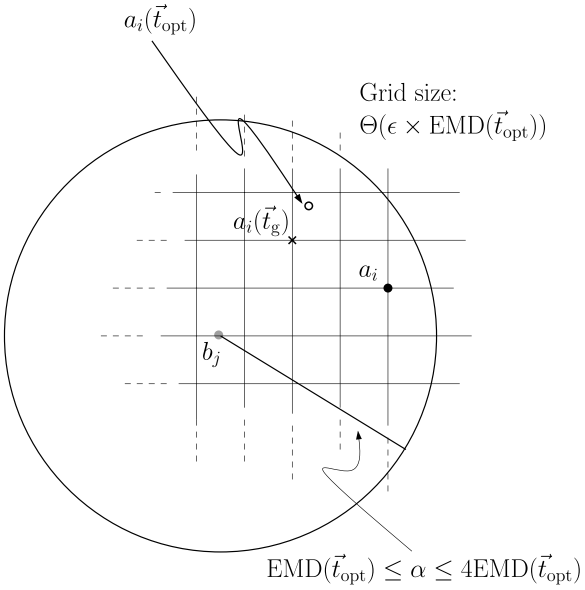

$a_i(\vec{t}_{\text{opt}})$
Grid size:
$\Theta(\epsilon \times \text{EMD}(\vec{t}_{\text{opt}}))$
$a_i(\vec{t}_{\text{g}})$
a_i
b_j
$\text{EMD}(\vec{t}_{\text{opt}}) \leq \alpha \leq 4\text{EMD}(\vec{t}_{\text{opt}})$

Figure 3.4: A pair of points a_i, b_j for which $d_{ij}(\vec{t}_{\text{opt}}) \leq \text{EMD}(\vec{t}_{\text{opt}})$, and a grid translation $\vec{t}_{\text{g}}$ of a_i for which $|\vec{t}_{\text{g}} - \vec{t}_{\text{opt}}| \leq (\epsilon/3)\text{EMD}(\vec{t}_{\text{opt}})$.

Hence, the algorithm reports a translation $\vec{t}_{\text{apx}}$ such that

$$\begin{aligned}
\text{EMD}(\vec{t}_{\text{apx}}) &\leq \widetilde{\text{EMD}}(\vec{t}_{\text{apx}}) \\
&\leq \widetilde{\text{EMD}}(\vec{t}_{\text{g}}) \\
&\leq (1+\epsilon/3)\text{EMD}(\vec{t}_{\text{g}}) \\
&\leq (1+\epsilon/3)(1+\epsilon/3)\text{EMD}(\vec{t}_{\text{opt}}) \\
&\leq (1+\epsilon)\text{EMD}(\vec{t}_{\text{opt}}),
\end{aligned}$$

for every $\epsilon \leq 3$. There are nm point-to-point translations, around each of which procedure ApxEMD is run for $O(\alpha^2/(\alpha^2\epsilon^2)) = O(1/\epsilon^2)$ grid points. Hence, the algorithm runs in $O((nm/\epsilon^2)(n^2/\epsilon^2)\log^2(n/\epsilon)) = O((n^3m/\epsilon^4)\log^2(n/\epsilon))$ time. □

3.4.1 Equal weight sets

In this section we consider the case of sets with equal total weights. Let $\vec{t}_{C(A)\rightarrow C(B)}$ be the translation that matches the centers of mass $C(A)$ and $C(B)$. Theorem 3.2 suggests the following trivial 2-approximation algorithm: compute $\text{EMD}(\vec{t}_{C(A)\rightarrow C(B)})$; we omit the proof since it is straightforward and very similar to the one of Lemma 3.2.

Also, according to Theorem 3.2, $\vec{t}_{\mathrm{opt}}$ is at most $\mathrm{EMD}(\vec{t}_{\mathrm{opt}})$ far away from $\vec{t}_{C(A)\to C(B)}$. Hence, we need to search for $\vec{t}_{\mathrm{opt}}$ only within a small neighborhood of $\vec{t}_{C(A)\to C(B)}$. We modify algorithm TRANSLATION as follows: First we compute $C(A)$ and $C(B)$. Then, we run $\mathrm{ApxEMD}(A(\vec{t}_{C(A)\to C(B)}), B, 1)$ and set α to the value returned. Next, we use the same grid size as in TRANSLATION, and run $\mathrm{ApxEMD}(A(\vec{t}_{\mathrm{g}}), B, \epsilon/3)$ for all the grid points $\vec{t}_{\mathrm{g}}$ which are at most α away from $\vec{t}_{C(A)\to C(B)}$. The minimum over all these approximations gives the desired approximation bound; this follows easily from arguments very similar to the ones used in the proof of Theorem 3.3. Note that the total number of grid points to be tested is $O(1/\epsilon^2)$. Hence, we have managed to save an nm term from the time bound of Theorem 3.3.

Theorem 3.4 *Let* $A = \{a_1, \ldots, a_m\}$ *and* $B = \{b_1, \ldots, b_n\}$ *be two weighted point sets in the plane with equal total weights and* $m \leq n$. *For any given* $\epsilon > 0$, *a translation* $\vec{t}_{\mathrm{apx}}$ *such that* $\mathrm{EMD}(\vec{t}_{\mathrm{apx}}) \leq (1+\epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}})$ *can be computed in* $O((n^2/\epsilon^4)\log^2(n/\epsilon))$ *time.*

For the assignment problem under translations, we can use the above algorithm for equal weight sets, running APXMATCH instead of APXEMD. This reduces the running time further.

Theorem 3.5 *For any given* $\epsilon > 0$, *a* $(1+\epsilon)$*-approximation of the minimum cost assignment under translations can be computed in* $O((n^{3/2}/\epsilon^{7/2})\log^5 n)$ *time.*

Note that the latter algorithm can be also applied to equal weight sets with bounded integer point weights by replacing each point by as many points as its weight.

3.4.2 Partial assignment

In Section 3.3, Observation 3.1, we saw that, in general, there is at least one pair of points a_i, b_j whose distance is at most EMD. Next, we prove that, for the partial assignment case, there is a linear number of pairs of points whose distance is at most 2EMD.

Lemma 3.3 *Given two weighted point sets* $A = \{a_1, \ldots, a_m\}$, $B = \{b_1, \ldots, b_n\}$ *with* $m \leq n$ *and* $w_i = u_j = 1, i = 1, \ldots, m, j = 1, \ldots, n$, *there are at least* $m/2$ *distances* d_{ij} *with* $d_{ij} \leq 2\mathrm{EMD}$.

Proof: Consider an optimal flow $\{f_{ij}\}$ that results in a partial assignment between A and B. Then there are exactly m flow variables f_{ij} with $f_{ij} = 1$ and $m(n-1)$ variables with zero flow. Since $\min\{W, U\} = m$, we have that $\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}d_{ij} = m\mathrm{EMD}$, where exactly m terms d_{ij} appear in the sum. Since $d_{ij} \geq 0$, it follows that there are at most k out of m distances d_{ij} with $d_{ij} \geq (m/k)\mathrm{EMD}$. Equivalently, there are at least $m-k$ distances d_{ij} with $d_{ij} \leq (m/k)\mathrm{EMD}$. We choose $k = m/2$, and the lemma follows. $\sqcup\!\!\!\!\sqcap$

Note that algorithm Translation tests all possible nm pairs of points a_i, b_j in order to find at least one for which $d_{ij}(\vec{t}_{\text{opt}}) \leq \text{EMD}(\vec{t}_{\text{opt}})$. Based on the above lemma, we can easily prove that testing a linear number of pairs suffices in order to find one for which $d_{ij}(\vec{t}_{\text{opt}}) \leq 2\text{EMD}(\vec{t}_{\text{opt}})$ with high probability. Algorithm RandomTranslation is given in Figure 3.5; it is a straightforward probabilistic version of algorithm Translation.

```
RandomTranslation($A, B, \epsilon$):
  1. Repeat $(2/\log e) n \log n$ times:
      (a) Choose a random pair $(a_i, b_j) \in A \times B$.
      (b) Let $\alpha_{ij}$ =ApxEMD($A(\vec{t}_{i \to j}), B, 1$).
     Let $\alpha = 2 \min_{i,j} \alpha_{ij}$ and $G$ be a uniform square grid of spacing $c\epsilon\alpha$,
     where $c = 1/\sqrt{288}$.
  2. Repeat $(2/\log e) n \log n$ times:
      (a) Choose a random pair $(a_i, b_j) \in A \times B$.
      (b) Place a disk $D$ of radius $\alpha$ around $\vec{t}_{i \to j}$.
      (c) For every grid point $\vec{t}_{\text{g}}$ of any cell of $G$ that intersects $D$
          compute a value $\widetilde{\text{EMD}}(\vec{t}_{\text{g}})$ = ApxEMD($A(\vec{t}_{\text{g}}), B, \epsilon/3$).
  3. Report the grid point $\vec{t}_{\text{apx}}$ that minimizes $\widetilde{\text{EMD}}(\vec{t}_{\text{g}})$.
```

Figure 3.5: Algorithm RandomTranslation(A, B, ϵ).

Theorem 3.6 *Let* $A = \{a_1, \ldots, a_m\}$ *and* $B = \{b_1, \ldots, b_n\}$ *be two weighted point sets with* $m \leq n$ *and* $w_i = u_j = 1, i = 1, \ldots, m, j = 1, \ldots, n$. *For any given* $\epsilon > 0$, RandomTranslation(A, B, ϵ) *computes a translation* $\vec{t}_{\text{apx}}$ *such that* $\text{EMD}(\vec{t}_{\text{apx}}) \leq (1+\epsilon)\text{EMD}(\vec{t}_{\text{opt}})$ *in* $O((n^3/\epsilon^4)\log^2(n/\epsilon)\log n)$ *time. The algorithm succeeds with probability at least* $1 - 2n^{-1}$.

Proof: According to Lemma 3.3, there are at least $m/2$ distances $d_{ij}(\vec{t}_{\text{opt}})$ with $d_{ij}(\vec{t}_{\text{opt}}) \leq 2\text{EMD}(\vec{t}_{\text{opt}})$. Since there are in total nm possible distances $d_{ij}(\vec{t}_{\text{opt}})$, we have that

$$\mathbf{Pr}[d_{ij}(\vec{t}_{\text{opt}}) > 2\text{EMD}(\vec{t}_{\text{opt}})] \leq 1 - m/(2nm) = 1 - 1/(2n)$$

for a random pair a_i, b_j. Thus, the probability that K random draws of a pair a_i, b_j will all fail to give a pair for which $d_{ij}(\vec{t}_{\text{opt}}) \leq 2\text{EMD}(\vec{t}_{\text{opt}})$ is at most $(1 - 1/2n)^K$. By choosing $K = (2/\log e) n \log n$ the latter probability is at most $e^{-(\log n)/\log e} = n^{-1}$.

The rest of the proof is almost identical to the proof of Theorem 3.3. That is, if a pair a_i, b_j for which $d_{ij}(\vec{t}_{\text{opt}}) \leq 2\text{EMD}(\vec{t}_{\text{opt}})$ is tested, then the algorithm will compute in step 1 a value α such that

$$2\text{EMD}(\vec{t}_{\text{opt}}) \leq \alpha < 8\text{EMD}(\vec{t}_{\text{opt}}).$$

Similarly, step 2 will report a translation $\vec{t}_{\text{apx}}$ such that

$$\text{EMD}(\vec{t}_{\text{opt}}) \leq \text{EMD}(\vec{t}_{\text{apx}}) \leq (1+\epsilon)\text{EMD}(\vec{t}_{\text{opt}})$$

in $O(((n \log n)/\epsilon^2)(n/\epsilon)^2 \log^2(n/\epsilon)) = O((n^3/\epsilon^4)\log^2(n/\epsilon)\log n)$ time. The algorithm fails to report such a translation if and only if any of its two random steps fail. That is, the algorithm fails with probability at most $2n^{-1}$. $\square$

3.5 Approximation algorithms for rigid motions

We first give $(2+\epsilon)$ and $(1+\epsilon)$-approximation algorithms for rotations for the general and partial assignment case respectively. Then, we combine these algorithms with the $(1+\epsilon)$-approximation algorithms for translations to get approximation algorithms for rigid motions.

3.5.1 Rotations

Let $\angle a_i o b_j$ be the angle between the segments $\overline{oa_i}$ and $\overline{ob_j}$ such that $0 \leq \angle a_i o b_j \leq \pi$. Also, let $\theta_{i \to j}$ be the rotation by $\angle a_i o b_j$ that aligns the origin o and points a_i and b_j such that both a_i and b_j are on the same side of o. Note that this is the rotation that minimizes $d_{ij}(\theta)$; we call such a rotation an *alignment rotation.*

We begin with a simple lemma that we will need later on.

Lemma 3.4 *Let a_i and b_j be two points in the plane with $\angle a_i o b_j = \phi$. If a_i is rotated by an angle $\theta \leq \phi$, then $d_{ij}(\theta) < 2d_{ij}$.*

Proof: First, note that we are only interested in the rotation of a_i that increases its distance to b_j; see Figure 3.6 for an illustration. Since $\theta \leq \phi$ we have that $d_{ij}(\theta) \leq d_{ij}(\phi)$, hence it suffices to bound $d_{ij}(\phi)$.

First, we assume that none of a_i and b_j coincides with the origin. Then, without loss of generality, we assume that $x_{b_j} > 0$ and $y_{b_j} = 0$. We also assume that $y_{a_i} \neq 0$, since, otherwise, if $x_{a_i} > 0$ then $\phi = 0$ and $d_{ij}(\phi) = d_{ij}$, or if $x_{a_i} < 0$ then $\phi = \pi$ and $d_{ij}(\phi) < d_{ij}$. Let $r_i = \sqrt{x_{a_i}^2 + y_{a_i}^2}$ be the rotation radius of a_i. We have

$$d_{ij} = \sqrt{r_i^2 + x_{b_j}^2 - 2x_{b_j} r_i \cos(\phi)}$$

and

$$d_{ij}(\phi) = \sqrt{r_i^2 + x_{b_j}^2 - 2x_{b_j} r_i \cos(2\phi)}.$$

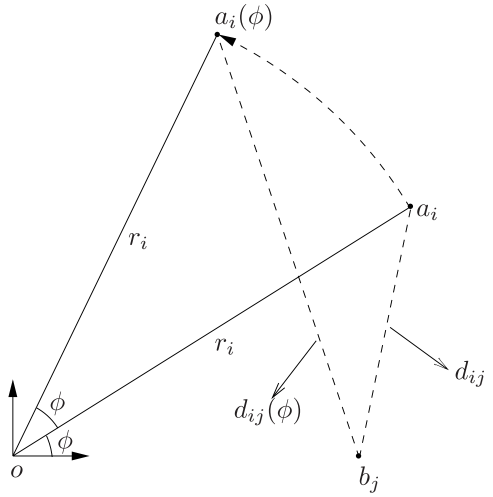


Figure 3.6: If $\angle a_i o b_j = \phi$ and a_i is rotated about o by ϕ then $d_{ij}(\phi) < 2d_{ij}$.

Then

$$\begin{aligned} 4d_{ij}^2 - d_{ij}^2(\phi) &= 3r_i^2 + 3x_{b_j}^2 + 2x_{b_j} r_i (2\cos^2(\phi) - 4\cos(\phi) - 1) \\ &> 3(r_i - x_{b_j})^2 \\ &\geq 0, \end{aligned}$$

where in the equality we used that $\cos(2x) = 2\cos^2(x) - 1$, and in the first inequality we used that $2\cos^2(\phi) - 4\cos(\phi) - 1 > -3$. Hence, $d_{ij}(\theta) \leq d_{ij}(\phi) < 2d_{ij}$. $\square$

Consider the angle $\angle a_i(\theta_{\text{opt}}) o b_j$ for every pair of points $a_i(\theta_{\text{opt}})$ and b_j and let $\angle a_{i_0}(\theta_{\text{opt}}) o b_{j_0}$ be the smallest of all these angles. Then $\theta_{i_0 \to j_0}$ is the alignment rotation that is *closest* to θ_{opt}. Similarly to Lemma 3.2, and using Lemma 3.4, we can now prove that this alignment rotation gives a 2-approximation of $\text{EMD}(\theta_{\text{opt}})$. Hence, we have the following:

Lemma 3.5 *Given two weighted point sets A and B, it holds that* $\text{EMD}(\theta_{\text{opt}}) \leq \min_{i,j} \text{EMD}(\theta_{i \to j}) \leq 2\text{EMD}(\theta_{\text{opt}})$.

Proof: Clearly, $\text{EMD}(\theta_{\text{opt}}) \leq \min_{i,j} \text{EMD}(\theta_{i \to j})$. Consider an optimal position $A(\theta_{\text{opt}})$ of A and an optimal flow $\{f_{ij}\}$ between $A(\theta_{\text{opt}})$ and B. We assume that θ_{opt} is not an alignment rotation, otherwise the lemma holds trivially. Next, consider the angle $\angle a_i(\theta_{\text{opt}}) o b_j$ for every pair of points $a_i(\theta_{\text{opt}})$ and b_j, and let $\angle a_{i_0}(\theta_{\text{opt}}) o b_{j_0}$ be the smallest of all these angles. Assume that we rotate $A(\theta_{\text{opt}})$ by $\angle a_{i_0}(\theta_{\text{opt}}) o b_{j_0}$ to the position $A(\theta_{i_0 \to j_0})$; this is the alignment rotation that is closest to θ_{opt}. Then, $\angle a_{i_0}(\theta_{i_0 \to j_0}) o b_{j_0}$ equals to 0 and

$$\angle a_i(\theta_{i_0 \to j_0}) o b_j \leq \angle a_i(\theta_{\text{opt}}) o b_j + \angle a_{i_0}(\theta_{\text{opt}}) o b_{j_0} \leq 2\angle a_i(\theta_{\text{opt}}) o b_j,$$

for every $i = 1, \ldots, m$ and $j = 1, \ldots, n$. According to Lemma 3.4 we have that $d_{ij}(\theta_{i_0 \to j_0}) < 2d_{ij}(\theta_{\text{opt}})$. Concluding,

$$\begin{aligned} \min_{i,j} \text{EMD}(\theta_{i \to j}) &\leq \text{EMD}(\theta_{i_0 \to j_0}) \\ &\leq \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}(\theta_{i_0 \to j_0})}{\min\{W, U\}} \\ &\leq \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} 2d_{ij}(\theta_{\text{opt}})}{\min\{W, U\}} \\ &= 2\text{EMD}(\theta_{\text{opt}}). \end{aligned}$$

□

By approximating $\min_{i,j} \text{EMD}(\theta_{i \to j})$ with $\min_{i,j} \text{ApxEMD}(A(\theta_{i \to j}), B, \epsilon/2)$ we can get a $(2 + \epsilon)$-approximation of $\text{EMD}(\theta_{\text{opt}})$. We call this algorithm Rotation(A, B, ϵ). Apart from the cost value, Rotation returns the corresponding rotation $\theta_{i \to j}$ as well. Hence we have the following:

Lemma 3.6 *For any given* $\epsilon > 0$, *a rotation* θ_{apx} *such that* $\text{EMD}(\theta_{\text{apx}}) \leq (2 + \epsilon)\text{EMD}(\theta_{\text{opt}})$ *can be computed in* $O((n^3 m/\epsilon^2) \log^2(n/\epsilon))$ *time.*

Partial assignment. Let us now consider rotations for the special case where all the weights are one, for which we will provide a $(1 + \epsilon)$-approximation. Let $a_1 b_{j_1}, \ldots, a_m b_{j_m}$ be a matching corresponding to an optimal integer flow at an optimal rotation θ_{opt} for the problem. Observe that $d_{ij_i}(\theta_{\text{opt}}) \leq m\text{EMD}(\theta_{\text{opt}})$ because $m\text{EMD}(\theta_{\text{opt}}) = \sum_i d_{ij_i}(\theta_{\text{opt}})$. This means that in order to find an optimal rotation we only need to consider the rotations

$$\{\theta \in [0, 2\pi) : d_{ij}(\theta) \leq m\text{EMD}(\theta_{\text{opt}})\},$$

for all i, j. Of course, since we do not know the value $\text{EMD}(\theta_{\text{opt}})$, we will instead consider the rotations $R_{ij}(\alpha) = \{\theta \in [0, 2\pi) : d_{ij}(\theta) \leq m\alpha\}$, for some value α such that $\text{EMD}(\theta_{\text{opt}}) \leq \alpha \leq 3\text{EMD}(\theta_{\text{opt}})$. Inside each R_{ij} we will consider sample rotations Θ_{ij} according to the following. We divide $R_{ij}(\alpha)$ into two parts, $R^{<}_{ij}(\alpha) = \{\theta \in [0, 2\pi) : d_{ij}(\theta) \leq \alpha\}$ and $R^{>}_{ij}(\alpha) = \{\theta \in [0, 2\pi) : \alpha \leq d_{ij}(\theta) \leq m\alpha\}$. To handle $R^{<}_{ij}(\alpha)$, we consider the set of distances

$$D^{<}_{ij}(\alpha) = \{k \cdot \epsilon \tfrac{\alpha}{18} \in [0, \alpha] \mid k \in \mathbb{N}\},$$

which consists of $O(1/\epsilon)$ values. To handle $R^{<}_{ij}(\alpha)$, we consider the set of distances

$$D^{>}_{ij}(\alpha) = \{\alpha(1 + \epsilon/6)^k \in [\alpha, m\alpha] \mid k \in \mathbb{N}\},$$

which contains $O(\log_{1+\epsilon} \frac{m\alpha}{\alpha}) = O(\log_{1+\epsilon} m) = O(\epsilon^{-1} \log m)$ values. Figure 3.7 gives an illustration of these two sets of distances. Let $D_{ij}(\alpha) = \{0, \alpha, m\alpha\} \cup D^{<}_{ij}(\alpha) \cup D^{>}_{ij}(\alpha)$, and consider the set of angles $\Theta_{ij} = \{\theta \in [0, 2\pi) \mid d_{ij}(\theta) \in$

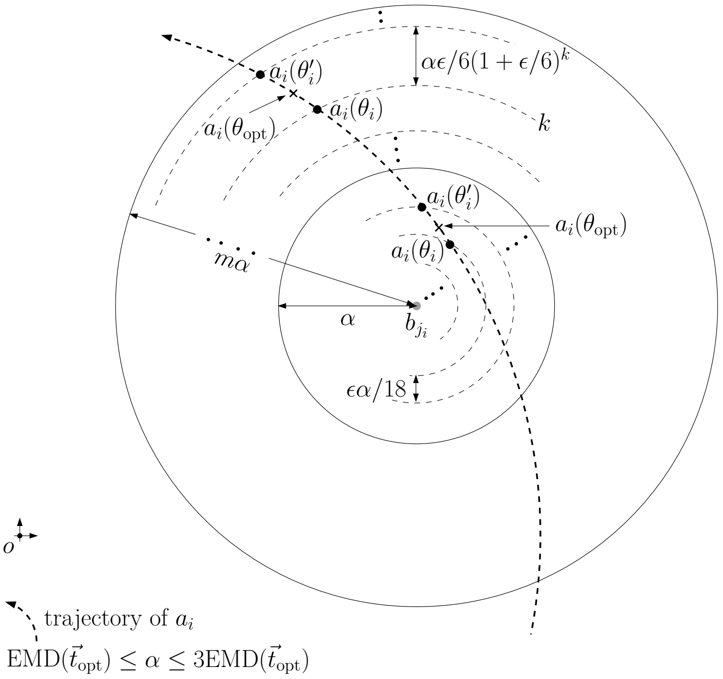

$\alpha\epsilon/6(1+\epsilon/6)^k$
$a_i(\theta_i')$
$a_i(\theta_i)$
$a_i(\theta_{\text{opt}})$
k
$a_i(\theta_i')$
$a_i(\theta_{\text{opt}})$
$a_i(\theta_i)$
$m\alpha$
α
b_{j_i}
$\epsilon\alpha/18$
o
trajectory of a_i
$\text{EMD}(\vec{t}_{\text{opt}}) \leq \alpha \leq 3\text{EMD}(\vec{t}_{\text{opt}})$

Figure 3.7: A pair of points a_i, b_{j_i} for which $d_{ij_i}(\theta_{\text{opt}}) \leq m\text{EMD}(\theta_{\text{opt}})$, and two examples of possible positions of $a_i(\theta_{\text{opt}}), a_i(\theta_i)$ and $a_i(\theta_i')$, depending on $d_{ij_i}(\theta_{\text{opt}})$.

$D_{ij}(\alpha)\}$. This finishes the description of Θ_{ij}, and therefore Θ_{ij} contains $O(\epsilon^{-1} \log m)$ angles.

The claim is that the best rotation among $\bigcup_{ij} \Theta_{ij}$ provides a $(1+\epsilon)$ - approximation for $\text{EMD}(\theta_{\text{opt}})$. The main idea is that the angles from Θ_{ij_i} that are in $R_{ij}^{<}(\alpha)$ take care for the case when $d_{ij_i}(\theta_{\text{opt}})$ is at most $\alpha \geq \text{EMD}(\theta_{\text{opt}})$ by controlling the absolute error this pair produces in the approximation, while the angles from Θ_{ij_i} that are in $R_{ij}^{>}(\alpha)$ take care for the case when $d_{ij_i}(\theta_{\text{opt}})$ is between α and $m\text{EMD}(\theta_{\text{opt}}) \leq m\alpha$ by controlling the relative error that the pair $a_i b_{j_i}$ produces.

A detailed description of the algorithm, referred to as PartRotation, is given in Figure 3.8. The algorithm shown runs ApxEMD for the general case where $m < n$; when $m = n$, ApxMATCH can be used instead.

Theorem 3.7 *Let* $A = \{a_1, \ldots, a_m\}$ *and* $B = \{b_1, \ldots, b_n\}$ *be two point sets with* $m \leq n$ *and* $w_i = u_j = 1, i = 1, \ldots, m, j = 1, \ldots, n$. *For any given* $\epsilon \in (0,1)$, PartRotation(A, B, ϵ) *computes a rotation* θ_{apx} *such that* $\text{EMD}(\theta_{\text{apx}}) \leq (1+\epsilon)\text{EMD}(\theta_{\text{opt}})$ *in* $O((n^3 m/\epsilon^3)\log^2(n/\epsilon)\log m)$ *time. When* $m = n$*, the same approximation can be computed in* $O((n^{7/2}/\epsilon^{5/2})\log^6 n)$ *time.*

Proof: First note that $\text{EMD}(\theta_{\text{opt}}) \leq \alpha \leq 3\text{EMD}(\theta_{\text{opt}})$. Let $a_1 b_{j_1}, \ldots, a_m b_{j_m}$

```
PartRotation($A, B, \epsilon$):
  1. Let $\alpha = \min_{i,j} \text{ApxEMD}(A(\theta_{i\to j}), B, 1)$.
  2. For each pair of points $a_i \in A$ and $b_j \in B$ do:
     (a) Compute $D_{ij}(\alpha) = \{0, \alpha, m\alpha\} \cup D^{<}_{ij}(\alpha) \cup D^{>}_{ij}(\alpha))$.
     (b) Let $\Theta_{ij} = \{\theta \in [0, 2\pi) \mid d_{ij}(\theta) \in D_{ij}(\alpha)\}$
     (c) For each sample rotation $\theta \in \Theta_{ij}$
         compute a value $\widetilde{\text{EMD}}(\theta) = \text{ApxEMD}(A(\theta), B, \epsilon/3)$.
  3. Report the sample rotation $\theta_{\text{apx}}$ that minimizes $\widetilde{\text{EMD}}(\theta)$.
```

Figure 3.8: Algorithm PartRotation(A, B, ϵ).

be a matching corresponding to an optimal integer flow at an optimal rotation θ_{opt}, and consider the sample rotation $\theta_{\text{g}} \in \bigcup_i \Theta_{ij_i}$ that is closest to θ_{opt}. Our objective is to show that $\widetilde{\text{EMD}}(\theta_{\text{g}}) \leq (1+\epsilon)\text{EMD}(\theta_{\text{opt}})$. Observe that if $\theta_{\text{opt}} \in \bigcup \Theta_{ij}$ then the approximation holds trivially.

Consider one pair $a_i b_{j_i}$, and let $\theta_i, \theta'_i \in \Theta_{ij_i}$ be the two closest angles such that θ_{opt} lies between them. We may assume that $d_{ij_i}(\theta_i) \leq d_{ij_i}(\theta'_i)$. Then it holds that $d_{ij_i}(\theta_i) \leq d_{ij_i}(\theta_{\text{opt}}) \leq d_{ij_i}(\theta'_i)$ and $d_{ij_i}(\theta_i) \leq d_{ij_i}(\theta_{\text{g}}) \leq d_{ij_i}(\theta'_i)$. If $d_{ij_i}(\theta_{\text{opt}}) < \alpha$, then $\theta_{\text{opt}} \in R^{<}_{ij_i}(\alpha)$, and also $\theta_i, \theta'_i \in R^{<}_{ij_i}(\alpha)$. Because θ_i, θ'_i are contiguous in Θ_{ij_i}, we have $d_{ij_i}(\theta'_i) - d_{ij_i}(\theta_i) \leq \epsilon\alpha/18$, and therefore

$$d_{ij_i}(\theta_{\text{g}}) - d_{ij_i}(\theta_{\text{opt}}) \leq \epsilon\alpha/18 \leq \epsilon\text{EMD}(\theta_{\text{opt}})/6.$$

If $d_{ij_i}(\theta_{\text{opt}}) > \alpha$, then $\theta_{\text{opt}} \in R^{>}_{ij_i}(\alpha)$, and also $\theta_i, \theta'_i \in R^{>}_{ij_i}(\alpha)$. Because θ_i, θ'_i are contiguous in Θ_{ij_i}, we have $d_{ij_i}(\theta'_i) \leq (1+\epsilon/6)d_{ij_i}(\theta_i)$, and therefore it holds that $d_{ij_i}(\theta_{\text{g}}) \leq (1+\epsilon/6)d_{ij_i}(\theta_{\text{opt}})$.

Then we have

$$\begin{aligned}
\text{EMD}(\theta_{\text{g}}) &\leq \frac{\sum_{i=1}^{m} d_{ij_i}(\theta_{\text{g}})}{m} \\
&\leq \frac{\sum_{\{i:d_{ij_i}(\theta_{\text{opt}})<\alpha\}} d_{ij_i}(\theta_{\text{g}}) + \sum_{\{i:d_{ij_i}(\theta_{\text{opt}})>\alpha\}} d_{ij_i}(\theta_{\text{g}})}{m} \\
&\leq \frac{\sum_{\{i:d_{ij_i}(\theta_{\text{opt}})<\alpha\}} \epsilon\text{EMD}(\theta_{\text{opt}})/6}{m} \\
&\quad + \frac{\sum_{\{i:d_{ij_i}(\theta_{\text{opt}})>\alpha\}} (1+\epsilon/6) d_{ij_i}(\theta_{\text{opt}})}{m} \\
&\leq \frac{\sum_{i=1}^{m} \epsilon\text{EMD}(\theta_{\text{opt}})/6}{m} + (1+\epsilon/6)\frac{\sum_{i=1}^{m} d_{ij_i}(\theta_{\text{opt}})}{m} \\
&= \epsilon\text{EMD}(\theta_{\text{opt}})/6 + (1+\epsilon/6)\text{EMD}(\theta_{\text{opt}}) \\
&= (1+\epsilon/3)\text{EMD}(\theta_{\text{opt}}),
\end{aligned}$$

and we conclude

$$\begin{aligned}\mathrm{EMD}(\theta_{\mathrm{apx}}) &\leq \widetilde{\mathrm{EMD}}(\theta_{\mathrm{apx}})\\ &\leq \widetilde{\mathrm{EMD}}(\theta_{\mathrm{g}})\\ &\leq (1+\epsilon/3)\mathrm{EMD}(\theta_{\mathrm{g}})\\ &\leq (1+\epsilon/3)(1+\epsilon/3)\mathrm{EMD}(\theta_{\mathrm{opt}})\\ &\leq (1+\epsilon)\mathrm{EMD}(\theta_{\mathrm{opt}}).\end{aligned}$$

Regarding the running time, observe that for each pair of points a_i, b_j we run ApxEMD for $O(\epsilon^{-1}\log m)$ sample rotations. Hence PartRotation runs in $O(nm/\epsilon \log m (n^2/\epsilon^2)\log^2(n/\epsilon)) = O((n^3 m/\epsilon^3)\log^2(n/\epsilon)\log m)$ time. When $n = m$ we can use ApxMATCH instead of ApxEMD, reducing the running time to $O(n^2/\epsilon \log n (n/\epsilon)^{3/2}\log^5 n) = O((n^{7/2}/\epsilon^{5/2})\log^6 n)$. $\square$

3.5.2 Rigid motions

We can combine algorithm Rotation with the 2-approximation algorithm for translations in Lemma 3.6 to get a $(4+\epsilon)$-approximation of the minimum EMD under rigid motions in the following way: for each point-to-point translation $\vec{t}_{i\to j}$, compute a $(2+\epsilon/2)$-approximation of the optimum EMD between $A(\vec{t}_{i\to j})$ and B under rotations about b_j. The minimum over all these approximations gives a $2(2+\epsilon/2)$-approximation of $\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$; see, for example, the first step of algorithm RigidMotion shown in Figure 3.9 where a 6-approximation of $\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ is computed.

Lemma 3.7 *For any given $\epsilon > 0$, a $(4+\epsilon)$-approximation of the minimum EMD under rigid motions can be computed in $O((n^4 m^2/\epsilon^2)\log^2(n/\epsilon))$ time.*

Proof: According to Observation 3.1, there exist two points a_{i_0}, b_{j_0} whose distance at an optimal position of A is at most the minimum EMD under rigid motions. The above algorithm will use, at some stage, b_{j_0} as the center of rotation by translating B appropriately. Of course for this 'new' position of B there is an optimal rigid motion of A, $I_{\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}}$ for which $d_{i_0,j_0}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \leq \mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ as well.

If A is translated by $\vec{t}_{i_0\to j_0}$ instead of $\vec{t}_{\mathrm{opt}}$, and then rotated by θ_{opt} we have $d_{ij}(\vec{t}_{i_0\to j_0}, \theta_{\mathrm{opt}}) \leq d_{ij}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) + |\vec{t}_{\mathrm{opt}} - \vec{t}_{i_0\to j_0}|$, for every $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Since $|\vec{t}_{\mathrm{opt}} - \vec{t}_{i_0\to j_0}| = d_{i_0,j_0}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \leq \mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ we have that $d_{ij}(\vec{t}_{i_0\to j_0}, \theta_{\mathrm{opt}}) \leq d_{ij}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) + \mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$. Similarly to the proof of Lemma 3.6, we see that

$$\mathrm{EMD}(\vec{t}_{i_0\to j_0}, \theta_{\mathrm{opt}}) \leq 2\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}).$$

If $\theta^{ij}_{\mathrm{opt}}$ is the optimal rotation of $A(\vec{t}_{i\to j})$ about b_j then

$$\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \leq \mathrm{EMD}(\vec{t}_{i_0\to j_0}, \theta^{i_0 j_0}_{\mathrm{opt}}) \leq \mathrm{EMD}(\vec{t}_{i_0\to j_0}, \theta_{\mathrm{opt}}).$$

Thus,

$$\begin{aligned}\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}) &\le \min_{ij}\mathrm{EMD}(\vec{t}_{i\to j},\theta^{ij}_{\mathrm{opt}})\\ &\le \mathrm{EMD}(\vec{t}_{i_0\to j_0},\theta^{i_0j_0}_{\mathrm{opt}})\\ &\le 2\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}).\end{aligned}$$

From Lemma 3.6 we also have that

$$\mathrm{EMD}(\vec{t}_{i\to j},\theta^{ij}_{\mathrm{opt}}) \le \textsc{Rotation}(A(\vec{t}_{i\to j}),B,\epsilon/2) \le (2+\epsilon/2)\mathrm{EMD}(\vec{t}_{i\to j},\theta^{ij}_{\mathrm{opt}}).$$

Putting it all together we get

$$\begin{aligned}\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}) &\le \min_{ij}\mathrm{EMD}(\vec{t}_{i\to j},\theta^{ij}_{\mathrm{opt}})\\ &\le \min_{ij}\textsc{Rotation}(A(\vec{t}_{i\to j}),B,\epsilon/2)\\ &\le (2+\epsilon/2)\min_{ij}\mathrm{EMD}(\vec{t}_{i\to j},\theta^{ij}_{\mathrm{opt}})\\ &\le 2(2+\epsilon/2)\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})\\ &= (4+\epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}).\end{aligned}$$

Since Rotation is run nm times, the running time of the algorithm is $O(nm(n^3 m/\epsilon^2)\log^2(n/\epsilon)) = O((n^4m^2/\epsilon^2)\log^2(n/\epsilon))$. ⊔⊓

The $(2+\epsilon)$-approximation algorithm for rigid motions is based on similar ideas. According to Observation 3.1, there exist two points a_i, b_j whose distance at $I_{\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}}$ is at most $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$. We place a grid of suitable size around each $\vec{t}_{i\to j}$. For each grid point $\vec{t}_{\mathrm{g}}$ that is at most $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$ away from $\vec{t}_{i\to j}$ we compute a $(2+\epsilon)$-approximation of the optimum EMD between $A(\vec{t}_{\mathrm{g}})$ and B under rotations about b_j. The minimum over all these approximations is within a factor of $(2+\epsilon)$ of $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$. Since we do not know $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$, we first compute a 6-approximation of it as shown above. Algorithm RigidMotion(A,B,ϵ) is shown in Figure 3.9; for the partial assignment problem, a $(1+\epsilon)$-approximation can be achieved by running PartRotation instead of Rotation.

Theorem 3.8 *Let* $A=\{a_1,\ldots,a_m\}$ *and* $B=\{b_1,\ldots,b_n\}$ *be two weighted point sets in the plane with* $m\le n$. *For any given* $\epsilon>0$, RigidMotion(A,B,ϵ) *computes a rigid motion* $I_{\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}}$ *such that* $\mathrm{EMD}(\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}) \le (2+\epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$ *in* $O((n^4m^2/\epsilon^4)\log^2(n/\epsilon))$ *time. A* $(1+\epsilon)$*-approximation of the minimum cost partial assignment under rigid motions can be computed in* $O((n^4m^2/\epsilon^5)\log^2(n/\epsilon))\log m$ *time.*

Proof: The proof is very similar to the proof of Lemma 3.7. First note that according to that lemma, $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}) \le \alpha \le 6\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$. Consider again a pair of points a_{i_0}, b_{j_0} such that $\vec{t}_{\mathrm{opt}}$ is at most $\mathrm{EMD}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$ away

```
RigidMotion($A, B, \epsilon$):
  1. For each pair of points $a_i \in A$ and $b_j \in B$ do:
     (a) Set the center of rotation, i.e. the origin, to be $b_j$ by translating $B$ appropriately.
     (b) Run Rotation($A(\vec{t}_{i \to j}), B, 1$) and let $\alpha_{ij}$ be the cost value returned.
     Let $\alpha = \min_{ij} \alpha_{ij}$.
  2. Let $G$ be a uniform grid of spacing $c\alpha\epsilon$, where $c = 1/\sqrt{288}$. For each pair of points $a_i \in A$ and $b_j \in B$ do:
     (a) Set the center of rotation, i.e. the origin, to be $b_j$ by translating $B$ appropriately.
     (b) Place a disk $D$ of radius $\alpha$ around $\vec{t}_{i \to j}$.
     (c) For every grid point $\vec{t}_{\mathrm{g}}$ of any cell of $G$ that intersects $D$ run Rotation($A(\vec{t}_{\mathrm{g}}), B, \epsilon/3$). Let $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$ and $\theta^{\mathrm{g}}_{\mathrm{apx}}$ be the cost value and angle returned respectively.
  3. Report the grid point $\vec{t}_{\mathrm{apx}}$ that minimizes $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$, and the corresponding angle $\theta_{\mathrm{apx}}$.
```

Figure 3.9: Algorithm RigidMotion(A, B, ϵ).

from $\vec{t}_{i_0 \to j_0}$. Since, at some stage, the algorithm will consider b_{j_0} as the center of rotation, we have that $\vec{t}_{\mathrm{opt}} \in D$, where D is a disk of radius α around $\vec{t}_{i_0 \to j_0}$. For the grid translation $\vec{t}_{\mathrm{g}}$ that is closest to $\vec{t}_{\mathrm{opt}}$ we have $|\vec{t}_{\mathrm{g}} - \vec{t}_{\mathrm{opt}}| \leq (1/4)\epsilon \mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$. Similarly to the proof of Theorem 3.3 we have that

$$\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) \leq (1 + \epsilon/4)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}).$$

If $\theta^{\mathrm{g}}_{\mathrm{opt}}$ is the optimal rotation of $A(\vec{t}_{\mathrm{g}})$ about b_{j_0} then of course $\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}) \leq \mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}})$. Note that Rotation($A(\vec{t}_{\mathrm{g}}), B, \epsilon/3$) returns a cost $\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$ for which

$$\widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}}) \leq (2 + \epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}),$$

and an angle $\theta^{\mathrm{g}}_{\mathrm{apx}}$ for which $\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) \leq \widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}})$. Hence, in total we have

$$\begin{aligned}
\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) &\leq \mathrm{EMD}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) \\
&\leq \widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{apx}}) \\
&\leq \widetilde{\mathrm{EMD}}(\vec{t}_{\mathrm{g}}) \\
&\leq (2 + \epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}) \\
&\leq (2 + \epsilon/3)\mathrm{EMD}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) \\
&\leq (2 + \epsilon/3)(1 + \epsilon/4)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \\
&\leq (2 + \epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}),
\end{aligned}$$

where the last inequality holds for any $\epsilon \le 2$.

Since Rotation runs for $O(nm/\epsilon^2)$ grid translations in total, the algorithm runs in $O((nm/\epsilon^2)(n^3m/\epsilon^2)\log^2(n/\epsilon)) = O((n^4m^2/\epsilon^4)\log^2(n/\epsilon))$ time. Note that for the partial assignment problem a $(1+\epsilon)$-approximation can be achieved by running PartRotation instead of Rotation; the running time increases to $O((nm/\epsilon^2)(n^3m/\epsilon^3)\log(m/\epsilon)\log^2(n/\epsilon)) = O((n^4m^2/\epsilon^5)\log(m/\epsilon)\log^2(n/\epsilon))$. ◻

As in the case of translations, for equal weight sets we need to search for the optimal translation only around $\vec{t}_{C(A)\to C(B)}$. We set the center of rotation to be $C(B)$. Computing the 6-approximation of $\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ can be done simply by running Rotation$(A(\vec{t}_{C(A)\to C(B)}), B, 1)$. Similarly, we need to run Rotation$(A(\vec{t}_{\mathrm{g}}), B, \epsilon/3)$ only for grid points $\vec{t}_{\mathrm{g}}$ that are close to $\vec{t}_{C(A)\to C(B)}$. For the assignment problem, instead of using Rotation, we can use the version of PartRotation that runs ApxMATCH to achieve a $(1+\epsilon)$-approximation.

Theorem 3.9 *If A and B have equal total weights, then, for any given $\epsilon > 0$, a rigid motion $I_{\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}}$ such that* $\mathrm{EMD}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) \le (2+\epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ *can be computed in $O((n^3m/\epsilon^4)\log^2(n/\epsilon))$ time. For the minimum cost assignment problem under rigid motions a $(1+\epsilon)$-approximation can be computed in $O((n^{7/2}/\epsilon^{9/2})\log^6 n)$ time.*

Finally, for the partial assignment problem under rigid motions, we can use the same arguments as in the translational case to convert algorithm Rigid-Motion – that will now use PartRotation – into a randomized one where its two first steps are executed only for a random selection of $\Theta(n\log n)$ pairs of points. We refrain from giving an exact description of the algorithm and a proof, since doing so will only produce unnecessary replication of ideas that have been already introduced. We conclude with the following:

Theorem 3.10 *Let $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$ be two weighted point sets with $m \le n$ and $w_i = u_j = 1, i = 1, ..., m, j = 1, ..., n$. For any given $\epsilon > 0$, a rigid motion $I_{\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}}$ such that* $\mathrm{EMD}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) \le (1+\epsilon)\mathrm{EMD}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ *can be computed in $O((n^4m/\epsilon^5)\log^2(n/\epsilon)\log n\log m)$ time. The algorithm succeeds with probability at least $1 - 2n^{-1}$.*

3.6 Concluding remarks

This chapter presented polynomial-time $(1+\epsilon)$ and $(2+\epsilon)$-approximation algorithms for the minimum Euclidean EMD under translations and rigid motions.

Note that algorithm ApxEMD in Section 3.2 can be trivially generalized in higher dimensions: for two d-dimensional point sets A and B with $|A| = m, |B| = n$ and $m \le n$, a $(1+\epsilon)$-spanner G_s with $O(\epsilon^{-d+1})$ edges can be computed in $O(n\log n + (n/\epsilon^d)\log(1/\epsilon))$ time [33]. As before, we run Orlin's algorithm on G_s and ApxEMD runs in $O((n^2/\epsilon^{2(d-1)})\log^2(n/\epsilon))$ time. Unfortunately we do not know whether the approximation algorithm of Varadarajan and Agarwal for the minimum cost bipartite matching in the plane carries on in higher dimensions. Also, note that the lower bounds in Section 3.3 and Lemma 3.3 hold for any dimension. Hence, for the general EMD in d-dimensional Euclidean space, a $(1+\epsilon)$-approximation of the minimum under translations can be computed in $O((n^3 m/\epsilon^{3d-2})\log^2(n/\epsilon))$ time. Algorithm RandomTranslation generalizes in a similar way.

We intend to work on extending the $(1+\epsilon)$-approximation for rotations to the general case of arbitrary weights; Another interesting and non-trivial task is to give lower and upper bounds of the complexity of the function $\mathrm{EMD}(\vec{t},\theta)$, i.e., the total number of its local optima.

Chapter 4

Maximizing the Area of Overlap of two Unions of Convex Objects under Translations

4.1 Introduction

In the last two chapters we have dealt with similarity measures for weighted-point sets based on weight (mass) transportation. Alternatively, we can think of the weight of a point as a property counted in area or volume: in the simplest case we can assign to each point a ball centered at it with radius relative to its weight. Then, a weighted-point set can define a shape given by a union of balls and a possible measure of the similarity of two such sets is the area of overlap of their respective unions.

In this chapter, we deal with matching shapes that are 'expressed' as unions of convex objects using the area of overlap as a similarity measure. In particular, consider two sets A and B consisting of either homothets of a single planar compact convex object or arbitrary planar compact fat convex objects under the following conditions: (i) the ratio of the areas of any two objects in $A \cup B$ is bounded (ii) any object in A intersects only a constant number of other objects in A, and the same holds for B. Fatness ensures that the objects are not infinitesimally thin; see Section 4.2 for a formal definition of fatness.

Each set is treated as a union of objects, that is, area that is covered by more than one object is counted only once. An example of two such sets A and B, consisting of disk homothets, and the area of overlap of their respective unions for some translation of A is given in Figure 4.1. The objective is to find a translation of A that maximizes the area of its overlap with B. We will see that this is a computationally hard problem, and, as an alternative, we will show

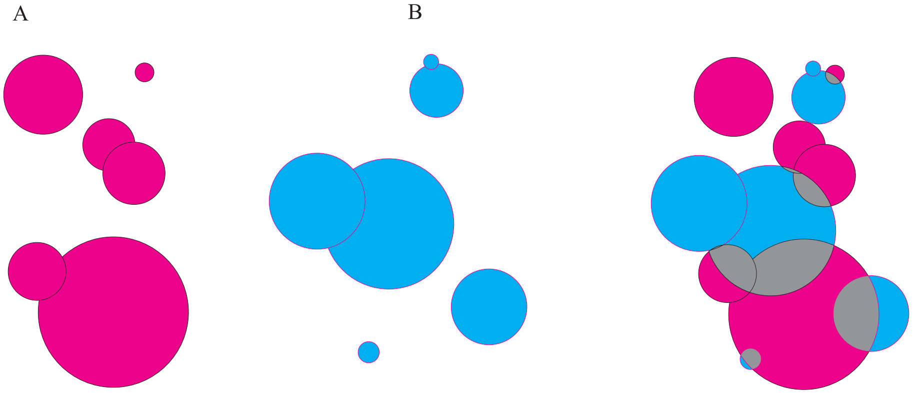


Figure 4.1: Two unions of disks A and B and their overlap for some translation of A.

how to compute a translation for which the area of overlap of is at least $(1-\epsilon)$ times that of an optimal translation.

4.1.1 Applications

As a first application consider the case of polygons with curved boundaries that was put forward as an open question by Mount et al. [96]. On one hand, we tackle a restricted version of this problem since clearly not every such polygon can be represented as a union of either convex homothets or fat convex objects. On the other hand, our setting is more general, since such unions can form shapes that can have non-simple, non-connected components.

Any two- or three-dimensional shape can be efficiently approximated by a finite union of disks or balls—see, for example, the works by O'Rourke and Badler [100] and Amenta and Kolluri [16]. Ranjan and Fournier [108] also used the union of disks or spheres representation to interpolate between two shapes. The assumptions (i) and (ii) above will often be satisfied when disks or balls are used to approximate objects, although the constant in assumption (i) may become large when the approximated objects have fine details. Moreover, both assumptions make perfect sense in molecular modeling with the hard sphere model [74]. Under this model the radii range of the spheres is fairly restricted and no center of a sphere can be inside another sphere; a simple packing argument shows that the latter implies assumption (ii). A related problem with applications in protein shape matching was examined by Agarwal et al. [1], who gave algorithms for minimizing the Hausdorff distance between two unions of disks or balls under translations.

4.1.2 Relevant work

As already mentioned in Section 1.5.2, Mount et al [96] examined the function of the area of overlap of two translated simple polygons, de Berg et al. [51]

gave an efficient algorithm for computing the maximum area of overlap of two convex polygons under translations and Alt et al. [10] gave a constant-factor approximation algorithm of the minimum area of symmetric difference of two convex shapes.

Independently of our work, Cheong et al. [39] gave an almost linear, probabilistic approximation algorithm that computes the maximum area of overlap of two unions of convex objects under translations up to an absolute error with high probability. When the maximum overlap is at least a constant fraction of the area of one of the two sets, the absolute error is in fact a relative error. As they point point out, this can be good enough for certain shape matching applications, since if two shapes are quite dissimilar we usually do not care about how bad the match exactly is. We will present their method in the next chapter where it is actually used in an approximation algorithm for rigid motions.

4.1.3 Results

Throughout this chapter, we focus on the case of sets of convex homothets; special attention is given to the sub-case of sets of disjoint unit disks. Once our main ideas are presented, we state the results for the case of sets of fat convex objects and give detailed proofs only when these differ substantially from the ones given in previous cases.

First, we show in Section 4.3 that the maximum number of combinatorially distinct translations of A with respect to B can be as high as $\Theta(n^2m)$. Moreover, the function describing the area of overlap is quite complex, even for combinatorially equivalent placements. Therefore, we focus on approximation algorithms.

We present a deterministic $(1-\epsilon)$-approximation algorithm for the maximum area of overlap under translations of two unions of either convex homothets or fat convex objects. This algorithm is the main result of this chapter. We assume that our objects have boundaries of constant description complexity and that the maximum number of intersections per object pair is constant. For example this is true for objects which can be defined as semialgebraic sets of constant description complexity[1]. Consequently, we can both compute the overlap of any two objects and decide whether a point is covered by an object in constant time. Under this model of computation, our algorithm runs in $O((nm/\epsilon^2)\log(m/\epsilon))$ time; this is worse than the algorithm of Cheong et al., but our algorithm is deterministic and our error is always relative, even when the optimum is small.

In Section 4.3.4 we study in detail the function of the area of overlap for sets of disjoint unit disks and give an intuitive geometric interpretation of the critical translations that correspond to the function's local/global optima and saddle points. In Section 4.4 we give a lower bound on the maximum area of overlap, expressed in the number of pairs of objects that contribute to that

---

[1] A subset of $\mathbb{R}^d$ is called a *real semialgebraic set* if it is obtained as a finite Boolean combination of sets of the form $\{f = 0\}$ or $\{f > 0\}$ for d-variate polynomials f. A semialgebraic set has *constant description complexity* if it can be described in terms of a constant number of polynomials, with a constant bound on the degrees of the corresponding polynomials.

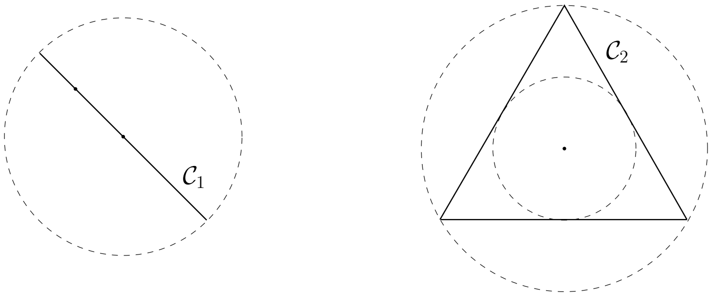


Figure 4.2: Object $\mathcal{C}_1$ is a degenerate convex object (straight line segment) and non-fat. Object C_2 is 4-fat. The dashed circles indicate the boundaries of the smallest enclosing and largest enclosed disks; the black dots represent their centers.

area. This is a vital ingredient of the proof of the approximation bound of the algorithm presented in Section 4.6. As a side-result of our analysis of the lower bound on the maximum area of overlap, constant-factor approximation schemes are presented as well; see Section 4.5. Note that, although we focus on the two-dimensional case, our results can be generalized to 3D for the case of sets of spheres in a straightforward way.

We start with the problem statement, some definitions and the notation that will be used throughout the chapter.

4.2 Preliminaries

Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$, $(n \leq m)$, be two sets of either homothets of *the same* planar compact convex object C or arbitrary planar compact fat convex objects. To avoid confusion we denote by $\mathcal{C}$ an arbitrary planar compact convex object; its area and interior are denoted by $V(\mathcal{C})$ and $\mathrm{Int}(\mathcal{C})$ respectively.

In the case of homothets, each $A_i \in A$ can be written as $\alpha_i \, C(\vec{a_i})$ for some translational vector $\vec{a_i}$ and scaling factor α_i; similarly, each $B_j \in B$ can be written as $\beta_j \, C(\vec{b_j})$.

In the case of fat convex objects, each $\mathcal{C}$ in $A \cup B$ is α-fat, for some fixed $\alpha > 1$; that is, for each $\mathcal{C} \in A \cup B$ the ratio of the area of its smallest enclosing disk $D'_{\mathcal{C}}$ to the area of its largest enclosed disk $D_{\mathcal{C}}$ is at most α; see Figure 4.2 for some examples. For more details concerning fat objects see van der Stappen [131] and Vleugels [138].

Throughout the paper the following assumptions hold:
(i) Let $\mathcal{C}_s$ and $\mathcal{C}_l$ be the object in $A \cup B$ with the smallest, resp. largest area. We assume that $V(\mathcal{C}_l)/V(\mathcal{C}_s) = \lambda$ for some constant $\lambda > 0$; we call λ the *scaling ratio*. Without loss of generality we can also assume that $V(C) = V(\mathcal{C}_s) = 1$ and $V(\mathcal{C}_l) = \lambda$.

(ii) We define the *depth* of a point $p \in \mathbb{R}^2$ with respect to a set of objects as the number of objects in the set that contain it. We assume that the depth of any point $p \in \mathbb{R}^2$ with respect to A is bounded by some constant; the same holds for the depth of any point $p \in \mathbb{R}^2$ with respect to B. We call the minimal such constant the *overall depth* and denote it by β. One case that will receive special attention is that of disjoint unit disks, where $\beta = 1$ and $\lambda = \pi$, i.e., all disks have radii equal to 1.

The two sets A and B lie in a two dimensional coordinate space that we call the *work space*. We consider set B to be fixed, while A can be translated relative to B. Let $\mathcal{T}$ be the infinite set of all possible translations of A, i.e. the set $\mathbb{R}^2$. We call $\mathcal{T}$ the *configuration space*. Translated versions of A are denoted with $A(\vec{t}) = \{A_i(\vec{t}), \ldots, A_n(\vec{t})\}$ for some $\vec{t} \in \mathcal{T}$. The *area of overlap* of $A(\vec{t})$ and B, as $\vec{t}$ varies, is a function $\mathcal{V} : \mathcal{T} \to \mathbb{R}$ with $\mathcal{V}(\vec{t}) = V((\bigcup A(\vec{t})) \cap (\bigcup B))$. We investigate the following problem:

For two sets A, B, defined as above, compute the optimal translation $\vec{t}_{\text{opt}}$ that maximizes $\mathcal{V}(\vec{t})$.

The Minkowski sum of two planar sets A and B, denoted by $A \oplus B$, is the set $\{p_1 + p_2 : p1 \in A, p_2 \in B\}$. Similarly, the Minkowski difference $A \ominus B$ is the set $\{p_1 - p_2 : p1 \in A, p_2 \in B\}$.

We define $T_{ij} = B_j \ominus A_i$. Note that $V(A_i(\vec{t}) \cap B_j) > 0$ if and only if $\vec{t} \in \text{Int}(T_{ij})$. Let $T(A, B) = \{T_{ij} : A_i \in A \text{ and } B_j \in B\}$. Then $\mathcal{V}(\vec{t}) > 0$ if and only if $\vec{t} \in \text{Int}(T(A, B))$. The boundaries of the Minkowski differences $T_{ij} \in T(A, B)$ induce a subdivision (arrangement of cells) of the configuration space [2]. It is easy to see that this arrangement can be non-simple and non-connected. Each cell in the arrangement is a set of combinatorially equivalent placements of A relative to B, i.e. the set of all overlapping pairs $(A_i(\vec{t}), B_j)$ is the same for all $\vec{t}$ in the cell.

We denote by $\mathcal{R}(A_i, B_j)$ the convex region whose boundary is exactly the convex contour traced by the boundary of A_i when A_i is translated around B_j such that their boundaries just touch. Of course $B_j \subset \mathcal{R}(A_i, B_j)$ and if $A_i(\vec{t}) \cap B_j \neq \emptyset$ then $A_i(\vec{t}) \subset \mathcal{R}(A_i, B_j)$.

In the next section we study the combinatorial complexity of the configuration space.

4.3 Complexity of the function of the area of overlap

We look first at the case of sets of convex homothets and then at the case of fat convex objects. Next we discuss a brute force method for finding the optimal

---

[2] Abusing the terminology slightly, we will sometimes use the term ‘configuration space’ when we are actually referring to the decomposition of the configuration space induced by the Minkowski differences.

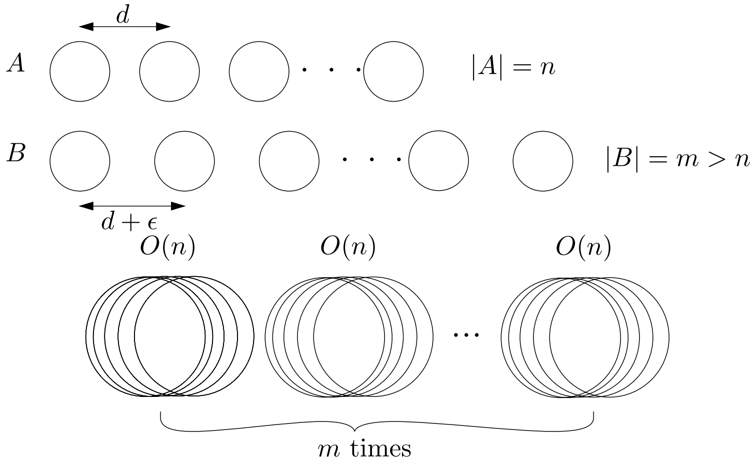


Figure 4.3: Two sets A and B and part of their configuration space $T(A, B)$ with highest $\Theta(n^2m)$ complexity.

translation taking the case of sets of disjoint unit disks as an example. For the latter case, we also study the function of the area of overlap and give a geometric characterization of its critical points.

4.3.1 Sets of convex homothets

We first look at the case of two sets of disjoint unit disks, which will also provide us with a lower bound example on the complexity of the configuration space.

Theorem 4.1 *Let A and B be the sets of n resp. m disjoint unit disks in the plane, with $m \geq n$. The complexity of the arrangement induced by $T(A, B)$, can be as high as $\Theta(n^2m)$.*

Proof: Each set $B_j \ominus A_i$ is a disk of radius 2. Since the disks in both sets are closed and disjoint, no disk of one set can intersect more than five disks of the other set at any position. This is due to the 'kissing' or Hadwiger number of unit open disks, which is six [127]. This implies that a point in the configuration space cannot be covered by more than five of the m disks $B_j \ominus A_i$, for any fixed i, $1 \leq j \leq m$. In total, considering all values of i, no point in the total arrangement can be covered by more than $5n$ disks $B_j \ominus A_i$. A similar argument for the disks in B gives that no point can be covered by more than $5m$ disks $B_j \ominus A_i$. Thus, the maximum depth of any point in the arrangement is $5 \min\{m, n\} = 5n$. Since the complexity of the arrangement of n pseudodisks with maximum depth k is $O(nk)$ [119], the complexity of the configuration space is $O(n^2m)$.

To see that this bound is tight consider the sets A and B shown in Figure 4.3. Let d be the distance between the centers of any two consecutive disks in A and $d' = d + \epsilon$, $\epsilon \neq 0$, the distance between the centers of any two consecutive disks in B. These two sets give an arrangement of which the part of highest complexity is shown in Figure 4.3. In each of the $\Omega(m)$ ‘bunches’ of $\Theta(n)$ disks, the complexity is $\Theta(n^2)$, since each disk intersects all the others. In total the complexity is $\Theta(n^2 m)$. ◻

Next, we generalize to the case of sets of non-disjoint, non-unit homothets of a fixed convex object.

Theorem 4.2 *Let A and B be two sets of n resp. m $(m \geq n)$ homothets of a convex object C in $\mathbb{R}^2$, with scaling ratio λ and overall depth β. The complexity of the arrangement induced by $T(A,B)$, can be as high as $\Theta(n^2 m)$.*

Note that for any planar convex object $\mathcal{C}$, we have $\mathcal{C} \ominus \mathcal{C} \oplus \mathcal{C} = \mathcal{R}(\mathcal{C},\mathcal{C})$. We need the following lemma in order to prove the theorem:

Lemma 4.1 *For a compact convex set $\mathcal{C}$ in $\mathbb{R}^2$, $9V(\mathcal{C}) \leq V(\mathcal{C}\ominus\mathcal{C}\oplus\mathcal{C}) \leq 13V(\mathcal{C})$; the left hand-side equality holds for centrally symmetric objects while the right hand-side equality holds for triangles.*

Proof: We need the following properties of the Minkowski sum $\mathcal{C}_1 \oplus \mathcal{C}_2$ and the mixed area $V(\mathcal{C}_1,\mathcal{C}_2)$ of two planar compact convex sets $\mathcal{C}_1,\mathcal{C}_2$; see Schneider [117]:

- $V(\mathcal{C}_1 \oplus \mathcal{C}_2) = V(\mathcal{C}_1) + 2V(\mathcal{C}_1,\mathcal{C}_2) + V(\mathcal{C}_2)$,
- $V(\mathcal{C} \ominus \mathcal{C}) \leq 6V(\mathcal{C})$, with the equality holding for triangles,
- $V(4V(\mathcal{C}) \leq \mathcal{C} \ominus \mathcal{C})$, with the equality holding for centrally symmetric objects,
- $V(\mathcal{C},\mathcal{C}) = V(\mathcal{C})$,
- $V(\mathcal{C}_1,\mathcal{C}_2) = V(\mathcal{C}_2,\mathcal{C}_1) \geq 0$ and
- $V(\mathcal{C}_1 \ominus \mathcal{C}_1,\mathcal{C}_2) = V(\mathcal{C}_1,\mathcal{C}_2) + V(-\mathcal{C}_1,\mathcal{C}_2)$.

Combining the above, we have:

$$\begin{aligned}
V(\mathcal{C} \ominus \mathcal{C} \oplus \mathcal{C}) &= V(\mathcal{C} \ominus \mathcal{C}) + 2V(\mathcal{C} \ominus \mathcal{C},\mathcal{C}) + V(\mathcal{C}) \\
&= 2V(\mathcal{C}) + 2V(\mathcal{C},-\mathcal{C}) + V(-\mathcal{C}) + 2V(\mathcal{C},\mathcal{C}) + 2V(-\mathcal{C},\mathcal{C}) \\
&= 5V(\mathcal{C}) + 4V(\mathcal{C},-\mathcal{C}).
\end{aligned}$$

Moreover, $V(\mathcal{C} \ominus \mathcal{C}) = 2V(\mathcal{C}) + 2V(\mathcal{C},-\mathcal{C})$ and $4V(\mathcal{C}) \leq V(\mathcal{C} \ominus \mathcal{C}) \leq 6V(\mathcal{C})$. So, $V(\mathcal{C}) \leq V(\mathcal{C},-\mathcal{C}) \leq 2V(\mathcal{C})$ and $9V(\mathcal{C}) \leq V(\mathcal{C} \ominus \mathcal{C} \oplus \mathcal{C}) \leq 13V(\mathcal{C})$. ◻

We can now prove Theorem 4.2:

Proof: We first give an upper bound on the number of B_j's that can intersect an A_i, for any fixed i. Since any A_i, B_j can be fully contained in $\mathcal{C}_l$, $\mathcal{R}(B_j, A_i)$ can be fully contained in $\mathcal{R}(\mathcal{C}_l, \mathcal{C}_l)$ and, by Lemma 4.1, we have

$$V(\mathcal{R}(B_j, A_i)) \leq V(\mathcal{R}(\mathcal{C}_l, \mathcal{C}_l)) \leq 13\lambda.$$

Since $V(B_j) \geq 1$, for any j, and the depth of the arrangement of all such B_j's is at most β, a simple volume argument gives an upper bound of $13\lambda\beta$. Following a similar argument as in the proof of Theorem 4.1, the maximum depth in configuration space is $13\lambda\beta \min\{m, n\} = O(n)$.

Unfortunately, we cannot use the same result from Sharir [119] as we did in Theorem 4.1 since the set of the Minkowski sums $T(A, B)$ is not necessarily a set of pseudodisks.

We continue by bounding the total number of intersections in configuration space from above. We denote with $T_s = \mathcal{C}_s \ominus \mathcal{C}_s$ and $T_l = \mathcal{C}_l \ominus \mathcal{C}_l$ the smallest and largest Minkowski differences respectively with $4 \leq V(T_s)$ and $V(T_l) \leq 6\lambda$, see the proof of Lemma 4.1; any $T_{ij} \in T(A, B)$ can be fully contained in T_l. Thus, for any fixed i', j', $\mathcal{R}(T_{ij}, T_{i'j'})$ can be fully contained in $\mathcal{R}(T_l, T_l)$. Based again on Lemma 4.1, the bounded maximum depth in configuration space and a volume argument, $T_{i'j'}$ cannot intersect more than $3/2 \cdot 13^2\lambda^2\beta \min\{m, n\} = O(n)$ other T_{ij}'s. Since the Minkowski difference of two objects of $O(1)$ complexity has $O(1)$ complexity, and we have $O(nm)$ such differences, the total bound follows. It follows from Theorem 4.1 that this bound is tight. ❏

4.3.2 Sets of fat convex objects

Let D be the smallest of all disks $D_{\mathcal{C}}$, $\mathcal{C} \in A \cup B$. Similarly, let D' be the largest of all disks $D'_{\mathcal{C}}$, $\mathcal{C} \in A \cup B$. Any $\mathcal{C} \in A \cup B$ can contain D and be contained in D'. Thus, since all the objects $\mathcal{C} \in A \cup B$ are α-fat, $V(D) \geq 1/\alpha$ and $V(D') \leq \alpha\lambda$.

Theorem 4.3 *Let A and B be two sets of n resp. m $(m \geq n)$ α-fat objects in $\mathbb{R}^2$, with scaling ratio λ and overall depth β. The complexity of the arrangement induced by $T(A, B)$, can be as high as $\Theta(n^2 m)$.*

The proof is similar to the proof of Theorem 4.2.

Proof: First, any $\mathcal{R}(B_j, A_i)$ can be contained in $\mathcal{R}(D', D')$. Since the overall depth is at most β, based on Lemma 4.1, a simple volume argument gives an upper bound of $9\lambda\alpha\beta$ on the number of B_j's that can intersect an A_i. Following a similar argument as in the proof of Theorem 4.1, the maximum depth in configuration space is $9\lambda\alpha\beta \min\{m, n\} = O(n)$.

Next, note that $V(T_{ij}) \geq 2$ for any T_{ij}. Any $T_{ij} \in T(A, B)$ can be contained in $D' \ominus D'$. $D' \ominus D'$ is point symmetric with $V(D' \ominus D') = 4V(D')$. For any fixed i', j', $\mathcal{R}(T_{ij}, T_{i'j'})$ can be fully contained in $\mathcal{R}(D' \ominus D', D' \ominus D')$. Since $V(\mathcal{R}(D' \ominus D', D' \ominus D')) \leq 36\lambda\alpha$, based on the bounded maximum depth of the configuration space and a volume argument, $T_{i'j'}$ cannot be intersected by more than $162\lambda^2\alpha^2\beta \min\{m, n\} = O(n)$ T_{ij}'s and the total bound follows. Since a unit disk is α-fat, this bound is tight according to Theorem 4.1. $\square$

4.3.3 A brute force approach to computing the optimum translation

Theorems 4.2 and 4.3, imply that $\mathcal{V}$ is a piecewise 2-variate function of complexity $\Theta(n^2m)$ with each piece being a sum of $\Theta(n)$ terms v_{ij}, with each term giving the area of overlap of a single pair of objects (A_i, B_j) as a function of the translational vector $\vec{t}(t_x, t_y)$. This means that the system of the first order necessary conditions $\partial\mathcal{V}/\partial t_x = \partial\mathcal{V}/\partial t_y = 0$, the solutions of which give all the local maxima and saddle points of $\mathcal{V}$, can give rise to $\Theta(n^2m)$ different systems of two equations each where each equation is a sum of $\Theta(n)$ terms. As the case of sets of disjoint unit disks shows, see Section 4.3.4, such a system can be quite complex and hard to solve exactly.

One could examine cases of sets of a particular convex object and, based on geometric properties of it, try to find the exact optimum efficiently. For example, in the case of disjoint unit squares, $\mathcal{V}$ is linear in a translation in a fixed horizontal or vertical direction; the maximum is obtained at an alignment of two horizontal edges and two vertical edges. Section 4.3.4 presents a geometric interpretation of the system of the first order necessary conditions for the case of sets of disjoint unit disks. However, we were unable to use this to find the optimum or at least bound the number of solutions.

4.3.4 Disjoint unit disks

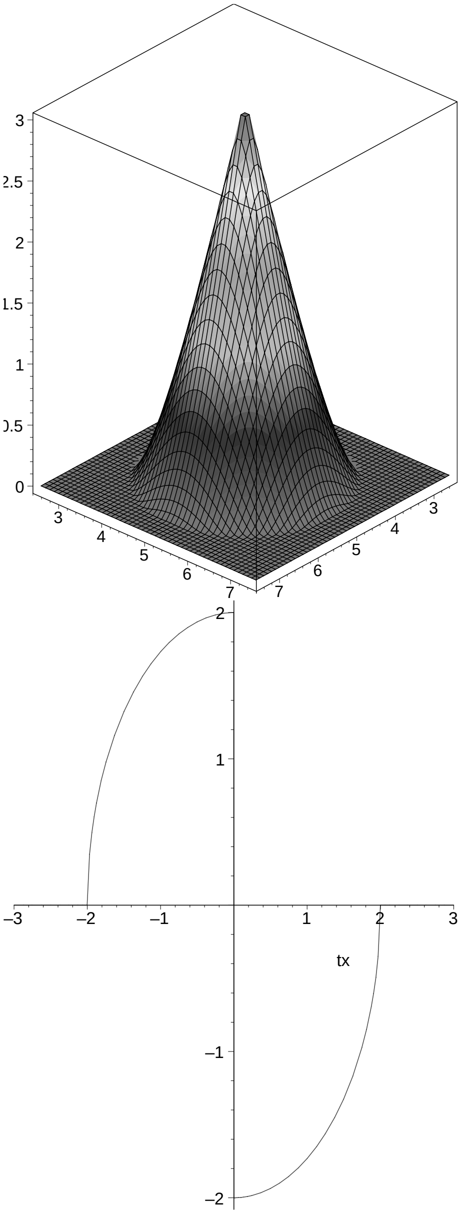
Let A_i and B_j be two disks of unit radius each and (x_i, y_i) and (x_j, y_j) be the coordinates of their centers. The area of their overlap v_{ij} as a function of $\vec{t}(t_x, t_y)$ has the following form:

$$\begin{aligned} v_{ij}(t_x, t_y) \quad &= \quad 2\arccos\left(1/2\sqrt{(x_i - x_j + t_x)^2 + (y_i - y_j + t_y)^2}\right) \\ &\quad - 1/2\left(\sqrt{(x_i - x_j + t_x)^2 + (y_i - y_j + t_y)^2}\right. \\ &\quad \left.\sqrt{4 - (x_i - x_j + t_x)^2 - (y_i - y_j + t_y)^2}\right). \end{aligned}$$

Figure 4.4, (a) plots this function; it has a circular base of radius 2 and a pointed top that corresponds to the center of the base where the maximum area of overlap π is achieved.

The partial derivatives of v_{ij} have the form:

$$\partial f_{ij}/\partial t_x = \frac{(x_j - x_i - t_x)\sqrt{4 - (x_i - x_j + t_x)^2 - (y_i - y_j + t_y)^2}}{\sqrt{(x_i - x_j + t_x)^2 + (y_i - y_j + t_y)^2}}$$



(a) (b)

Figure 4.4: The function of the area of overlap of two unit disks and its derivative.

and

$$\partial v_{ij}/\partial t_x = \frac{(y_j - y_i - t_y)\sqrt{4 - (x_i - x_j + t_x)^2 - (y_i - y_j + t_y)^2}}{\sqrt{(x_i - x_j + t_x)^2 + (y_i - y_j + t_y)^2}}.$$

We denote with $d = \sqrt{(x_i - x_j + t_x)^2 + (y_i - y_j + t_y)^2}$ the distance between the centers of A_i and B_j. Then, $v_{ij}(d)$ is just any 2-D vertical to the (t_x, t_y)-plane slice of $v_{ij}(t_x, t_y)$. Figure 4.4, (b) shows $\partial f_{ij}(d)/\partial d$. It is clear that $v_{ij}(d)$ is not differentiable when $d = 0$ where the centers of the two disks align and the area of overlap is π; at this position an infinitesimally small move dx towards any direction will instantly decrease the overlap by $2dx$; reaching this position from any direction instantly increases the overlap by $2dx$. However v_{ij} is differentiable at all other positions for which $d \leq 2$; using standard methods (conjugate, l'hospital) it can be easily shown that $\partial f_{ij}(d)/\partial d = 0$ when $d = 2$.

As already explained in the previous section, the system $\partial \mathcal{V}/\partial t_x = \partial \mathcal{V}/\partial t_y = 0$ can give rise to $\Theta(n^2 m)$ different systems of two equations each where each equation is a sum of $\Theta(n)$ terms which are radicals in our case (of course one has to carefully single out the $\Theta(n^2)$ center alignment translations where the partial derivatives of the terms are not defined). Using extra variables we can convert each such system into a system that consists of $2n$ equations of polynomials of quadratic degree each and two equations of polynomials of $n+1$ degree each. Bezout's theorem [55] gives an exponential $2^{2n}(n+1)^2$ upper bound on the number of roots(the polynomials have no common factors). It would be interesting to see whether the theory of mixed volume of the Newton polytopes [47, 55] gives a better upper bound, and whether sparse elimination techniques could solve the system faster, but this falls out of the scope of this thesis. Using numerical approximations, the cost of the solution could be lowered but most techniques give rather limited control on the error bound.

A geometric interpretation

Let $A_i(x_i, y_i), B_j(x_j, y_j)$ be a pair of intersecting disks. Let us first assume, without loss of generality, that $\vec{t}(t_x, t_y)$ is such that $y_j = y_i + t_y$ and $x_j \neq x_i + t_x$ that is, the centers of both disks lie on a line parallel to the x-axis but don't coincide.

Then

$$\partial v_{ij}/\partial t_y = 0$$

and

$$\partial v_{ij}/\partial t_x = ((x_j - x_i - t_x)/|x_i - x_j + t_x|)\sqrt{4 - (x_i + t_x - x_j)^2} = \pm\alpha.$$

where α is the height of the lens of the intersection of the two disks. In general, an infinitesimally small move in the direction of the line connecting the two centers will increase or decrease the area of overlap by α–the change depends only on the distance between the two centers–and in a direction perpendicular to that line the change is zero.

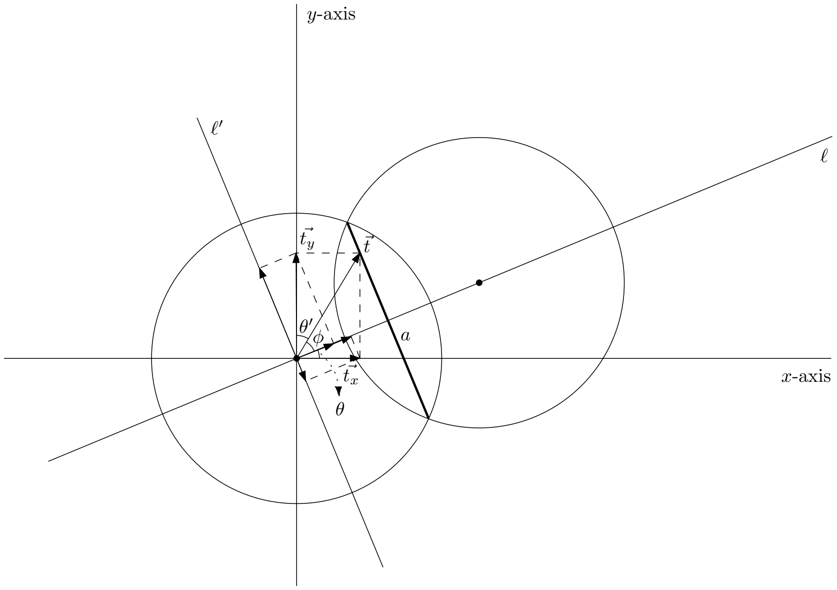


Figure 4.5: Geometric interpretation of ∂f_{ij}.

Generalizing this to arbitrary translations, see Figure 4.5, we have:

$$\partial v_{ij}/\partial t_x = \alpha \cos(\theta),$$

$$\partial v_{ij}/\partial t_y = \alpha \cos(\theta')$$

and

$$\partial v_{ij}/\partial t_x + v_{ij}/\partial t_y = \alpha \cos(\phi).$$

Alternatively we can give the following geometric interpretation of the first-order necessary condition whenever $\partial \mathcal{V}$ is defined. We illustrate the idea for four pairs of overlapping disks $(A_i, B_i), i = 1, \ldots, 4$, see Figure 4.6. Let $\vec{v_i}$ be a vector pointing from the center of A_i to the center of B_i–a move in that direction will instantly increase the area of overlap for each pair individually–such that $|\vec{v_i}| = \alpha_i$, then: $\sum_{i=1}^{4} \vec{v_i} = \vec{0}$.

4.4 A lower bound on the area of overlap

This section gives a lower bound on the maximum area of overlap, expressed in the number of pairs of objects that contribute to that area. First, we present the case of sets of convex homothets in detail, and then we extend the result to sets of fat convex objects. The main result is the following theorem:

Theorem 4.4 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$ each be a set of homothets of a convex object C in $\mathbb{R}^2$, such that (i) the overall depth is β (ii) the*

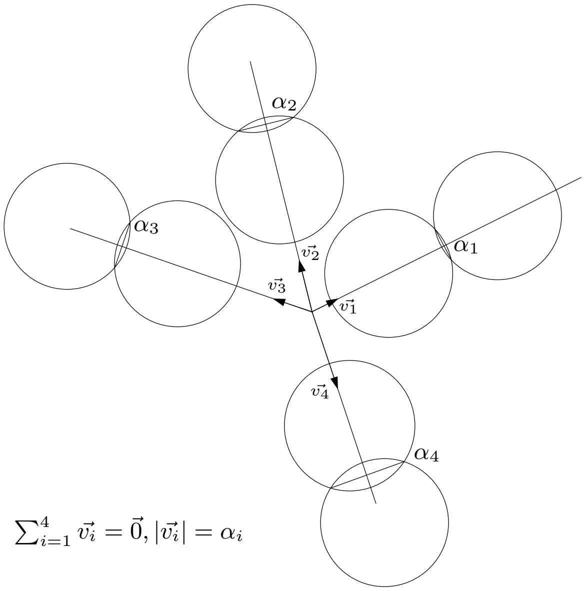


Figure 4.6: Geometric interpretation of critical translations.

scaling ratio is λ*, for some constants* $\beta, \lambda > 0$*. Let* $\vec{t}_{\mathrm{opt}}$ *be the translation that maximizes the area of overlap* $\mathcal{V}(\vec{t})$ *of* $A(\vec{t})$ *and* B *over all possible translations* $\vec{t}$ *of the set* A*. If* k *is the number of overlapping pairs* $A_i(\vec{t}_{\mathrm{opt}})$ *and* B_j*, then* $\mathcal{V}(\vec{t}_{\mathrm{opt}})$ *is* $\Theta(k)$*.*

To prove this theorem, we need to introduce some additional concepts, notation, and two lemmas.

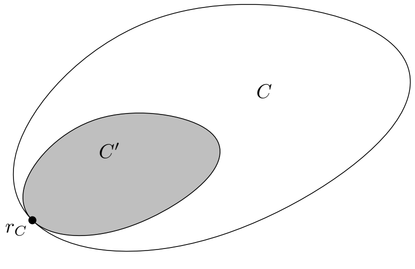


Figure 4.7: A convex object C, its reference point r_C, and its kernel C' (shaded).

Pick an arbitrary point on the boundary of C, and let this be the *reference point* of C, denoted with r_C, see Figure 4.7. The reference point for C also fixes the reference points for all objects in A and B. Furthermore, the choice of the reference point defines the *kernel* C' of C: it is a scaled version of C, scaled with a factor $1/2\sqrt{V(C)}$, with r_C as the center of scaling. The kernels of the homothets $A_i(\vec{t})$ in $A(\vec{t})$ and B_j in B are defined similarly and denoted by $A'_i(\vec{t})$ and B'_j correspondingly; their reference points are denoted by $r_{A_i(\vec{t})}$ and r_{B_j}. Note that while the objects $A_i(\vec{t})$ and B_j may all have different areas, the areas of the kernels $A'_i(\vec{t})$ and B'_j are all the same, namely, 1/4 (we have assumed that $V(C) = 1$).

The idea now is to consider other translations than the optimal one, namely, translations for which the number of reference points r_{A_i} that lie inside or on the boundary of kernels B'_j is large. The area of overlap $V(\vec{t})$ at any such translation $\vec{t}$ is obviously a lower bound on the area of overlap $V(\vec{t}_{\text{opt}})$ at $\vec{t}_{\text{opt}}$. Furthermore, for a pair $A_i(\vec{t})$ and B_j for which r_{A_i} lies inside or on the boundary of B'_j, we establish a lower bound on the contribution of the pair to the overlap at $\vec{t}$:

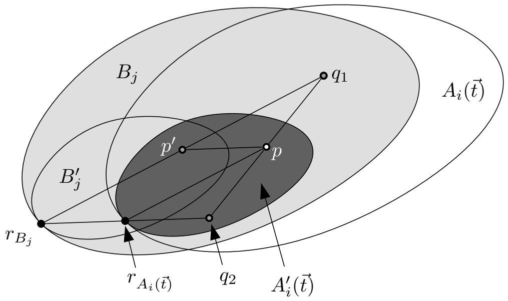


Figure 4.8: If the reference point of $A_i(\vec{t})$ lies inside the kernel of B_j, then the kernel of $A_i(\vec{t})$ (darkly shaded) lies inside B_j (lightly shaded). In this illustration, both $A_i(\vec{t})$ and B_j have an area of 1. However, the claim still holds if either of $A_i(\vec{t})$ or B_j is larger, as the kernels have a fixed area of 1/4.

Lemma 4.2 *If the reference point $r_{A_i(\vec{t})}$ of $A_i(\vec{t})$ lies inside or on the boundary of the kernel B'_j of B_j, then $V(A_i(\vec{t}) \cap B_j)$ is at least* 1/4*.*

Proof: Assume that the reference point $r_{A_i(\vec{t})}$ of $A_i(\vec{t})$ lies inside or on the boundary of the kernel B'_j of B_j. Let p be an arbitrary point in the kernel $A'_i(\vec{t})$ of $A_i(\vec{t})$ (see Figure 4.8). We construct three additional points p', q_1, and q_2 as follows: $p' = p + (r_{B_j} - r_{A_i(\vec{t})})$. In other words: the relative position of p' with respect to the kernel of B_j is the same as the position of p w.r.t. the kernel of A_i, so p' lies inside or on the boundary of B'_j. Next, we define $q_1 = p' + (p' - r_{B_j})$. Since the distance from r_{B_j} to q_1 is exactly twice the distance from r_{B_j} to p', and p' lies in the kernel of B_j or on its boundary, and B'_j is a scaled version of B_j with a factor of at most 1/2 with r_{B_j} as the center of scaling, q_1 must lie inside B_j or on its boundary. Next, we define $q_2 = r_{A_i(\vec{t})} + (r_{A_i(\vec{t})} - r_{B_j})$. Since $r_{A_i(\vec{t})}$ lies inside B'_j or on its boundary, q_2 must lie inside B_j or on its boundary, by similar reasoning as above. Finally, since $\triangle(r_{A_i(\vec{t})}, q_2, p)$ is congruent with $\triangle(r_{B_j}, q_2, q_1)$, we have that p lies on the segment $\overline{q_1 q_2}$. Since both q_1 and q_2 lie inside or on the boundary of B_j, and B_j is convex, p must lie inside B_j or on its boundary as well. This holds for all points p inside $A'_i(\vec{t})$, and since the area of $A'_i(\vec{t})$ is 1/4 (recall that all kernels have area 1/4), the lemma follows. $\Box$

Now let us focus on those pairs $A_i(\vec{t}_{\text{opt}})$ and B_j that contribute to the overlap at the optimal translation $\vec{t}_{\text{opt}}$, and disregard all other pairs of objects. For all pairs $A_i(\vec{t}_{\text{opt}})$ and B_j under consideration, we look at the translations that bring the reference point of $A_i(\vec{t}_{\text{opt}})$ into the kernel of B_j. For any such pair $A_i(\vec{t}_{\text{opt}})$ and B_j, these translations form a region $\mathcal{K}_{ij}$ in configuration space that has exactly the size and shape of a kernel B'_j (or $A'_i(\vec{t}_{\text{opt}})$ for that matter, since all kernels have the same area). The precise locations of all these regions $\mathcal{K}_{ij}$ depend on the relative positions of the $A_i(\vec{t}_{\text{opt}})$ with their corresponding B_j. However, it is not difficult to determine a region $\mathcal{R}$ in configuration space, of bounded size, that contains all regions $\mathcal{K}_{ij}$. Consider a pair of objects $\mathcal{C}_1$ and $\mathcal{C}_2$ of area λ (the largest possible area for any object in A or B). Figure 4.9 (top) shows $\mathcal{C}_1$ (lightly shaded) and its kernel $\mathcal{C}'_1$ (darkly shaded), and a subset of the *possible* locations for $\mathcal{C}_2$ (only some extreme locations, where $\mathcal{C}_2$ just touches $\mathcal{C}_1$ are shown). The bottom half of the figure shows the region $\mathcal{R}$ of translations that bring the reference points of all possible $\mathcal{C}_2$ into the kernel of $\mathcal{C}_1$ (this is the darkly shaded region; it includes the lightly shaded region and the black region). The small dark area in the bottom half of the figure is the set of translations that bring the reference point of $\mathcal{C}_2$ at the “black position” (in the top half of the figure) into the kernel of $\mathcal{C}_1$.

Since the size of any object $A_i(\vec{t}_{\text{opt}})$ or B_j is less than or equal to the size of $\mathcal{C}_1$ (and $\mathcal{C}_2$), any $\mathcal{K}_{ij}$ under consideration must be contained in $\mathcal{R}$. The region $\mathcal{R}$ has area $V(\mathcal{R}) = V(\mathcal{C}_1 \ominus \mathcal{C}_2 \oplus \mathcal{C}'_1)$. Since $\mathcal{C}_1$ and $\mathcal{C}_2$ have the same shape and area (namely, λ), and $\mathcal{C}'_1$ is fully contained in $\mathcal{C}_1$, this area is at most $V(\mathcal{C}_1 \ominus \mathcal{C}_1 \oplus \mathcal{C}_1)$.

Now instead of looking at all *possible* locations of the $A_i(\vec{t}_{\text{opt}})$ with respect to their corresponding B_j, look at the *actual* positions: for each $A_i(\vec{t}_{\text{opt}})$ that has a non-zero overlap with some B_j, we draw the region of translations $\mathcal{K}_{ij}$ that brings $r_{A_i(\vec{t}_{\text{opt}})}$ into B'_j. Such a region has the same shape and size as B'_j, and it is fully contained in $\mathcal{R}$. By Lemma 4.1, the area of $\mathcal{R}$ is at most 13λ, where the constant λ is the area of the largest object in $A \cup B$. On the other hand, the area of each $\mathcal{K}_{ij}$ is 1/4. A simple volume argument tells us that there must be a point in $\mathcal{R}$ that is covered by at least $k/4 \cdot 1/13\lambda$ regions $\mathcal{K}_{ij}$, so we have:

Lemma 4.3 *If the number of pairs $A_i(\vec{t}_{\text{opt}})$ and B_j that have non-zero overlap is denoted with k, then there exists a translation $\vec{t}_\sharp$ such that the number of reference points $r_{A_i(\vec{t}_\sharp)}$ that are contained in some kernel B'_j is $\Omega(k)$.*

We are now ready to prove Theorem 4.4:

Proof: According to Lemma 4.3, there is a translation $\vec{t}_\sharp$ that brings $\Omega(k)$ reference points $r_{A_i(\vec{t}_\sharp)}$ into some kernels B'_j. Each of the corresponding $A_i(\vec{t}_\sharp)$ and B_j have an overlap of at least 1/4 by Lemma 4.2. There may be regions in the overlap that are covered by more than one A_i or B_j. However, since the

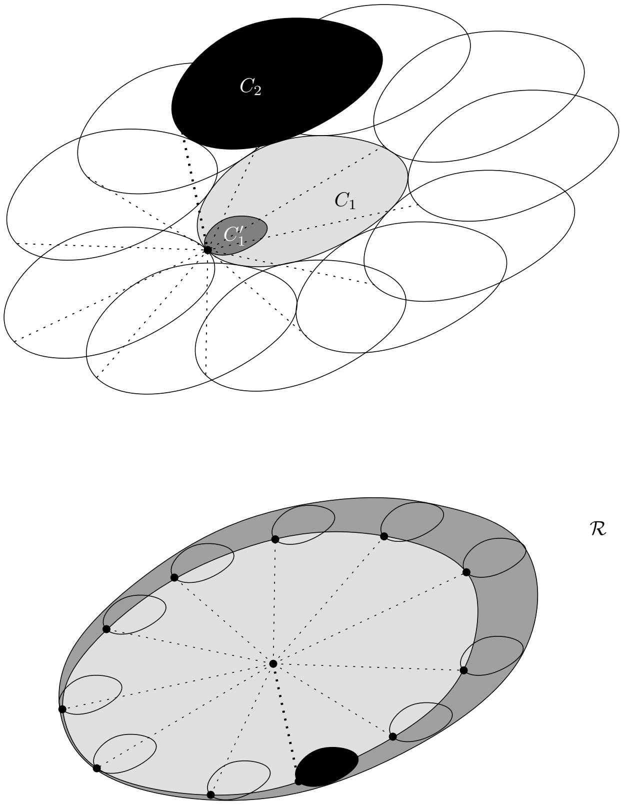


Figure 4.9: Extremal positions of $\mathcal{C}_2$ w.r.t. $\mathcal{C}_1$ in workspace (top), and the set of translations in translation space that bring the reference points of $\mathcal{C}_2$ into the kernel of $\mathcal{C}_1$, for all possible positions of $\mathcal{C}_2$ (bottom).

depth of both A and B is bounded by a constant β, any region in the overlap is double-counted at most β^2 times. It follows that total area of overlap is $\Omega(k)$ for the translation $\vec{t}_\sharp$. This area of overlap is a lower bound for the area of overlap at $\vec{t}_{\text{opt}}$. Since each pair $A_i(\vec{t})$ and B_j that overlaps contributes at most a constant λ to the area of overlap, the total area of overlap is $O(k)$, and the theorem follows. □

Basic arithmetic reveals that the constant hidden in the Ω-notation for the lowerbound on the optimum area of overlap is $1/(208\lambda\beta^2)$.

For sets of fat convex objects Theorem 4.4 becomes:

Theorem 4.5 *Let* $A = \{A_1, \ldots, A_n\}$ *and* $B = \{B_1, \ldots, B_m\}$ *each be a set of* α*-fat convex objects in* $\mathbb{R}^2$*, for some fixed* $\alpha > 1$*, such that (i) the overall depth is* β *(ii) the scaling ratio is* λ*, for some constants* $\beta, \lambda > 0$*. Let* $\vec{t}_{\text{opt}}$ *be the translation that maximizes the area of overlap* $\mathcal{V}(\vec{t})$ *of* $A(\vec{t})$ *and* B *over* *all possible translations* $\vec{t}$ *of the set* A. *If* k *is the number of overlapping pairs* $A_i(\vec{t}_{\mathrm{opt}})$ *and* B_j, *then* $\mathcal{V}(\vec{t}_{\mathrm{opt}})$ *is* $\Theta(k)$.

Proof: We define the reference point $r_{\mathcal{C}}$ of $\mathcal{C} \in A \cup B$ to be the center of its largest enclosed disk. The kernel $\mathcal{C}'$ of $\mathcal{C}$ is defined as the disk D, as this was defined in Section 4.3.2, centered at $r_{\mathcal{C}}$; that is D is the smallest largest enclosed disk $D_{\mathcal{C}}$ over all $\mathcal{C} \in A \cup B$. For each of the k pairs $(A_i(\vec{t}_{\mathrm{opt}}), B_j)$ such that $V(A_i(\vec{t}_{\mathrm{opt}}) \cap B_j) \neq 0$ we consider the set of translations that bring the reference point of $A_i(\vec{t}_{\mathrm{opt}})$ into the B'_j; these form a region $\mathcal{K}_{ij}$ in configuration space that is a disk with $V(\mathcal{K}_{ij}) \geq 1/\alpha$. All these k regions $\mathcal{K}_{ij}$ can be easily shown to be contained in a region $\mathcal{R} = \mathcal{R}(D', D')$ in configuration space with $V(\mathcal{R}) \leq 9\lambda\alpha$, where D' is the largest smallest enclosing disk $D'_{\mathcal{C}}$ over all $\mathcal{C} \in A \cup B$. Thus, there is a translation $\vec{t}_\sharp$ that brings at least $k/(9\lambda\alpha^2)$ reference points $r_{A_i(\vec{t}_\sharp)}$ into some kernels B'_j. By Lemma 4.4, see next section, each of the corresponding $A_i(\vec{t}_\sharp)$ and B_j have an overlap of at least $1/(4\alpha)$ and the theorem follows. The constant hidden here in the Ω-notation for the lowerbound is $1/(36\lambda\alpha^3\beta^2)$. ◻

4.5 Constant factor approximation schemes

Theorem 4.4 suggests a constant-factor approximation algorithm: compute the arrangement $\mathcal{A}$ of the regions $\mathcal{K}_{ij}$, $i = 1, \ldots, n$, $j = 1, \ldots, m$. Find a cell of maximum depth in $\mathcal{A}$, and pick any point in this cell. Such a point corresponds to a translation $\vec{t}$ that gives a constant-factor approximation. In the most general setting, where we look at homothets of an arbitrary convex shape with bounded size ratio and bounded depth in both A and B, the approximation factor is too small for practical purposes. In the special case where A and B are sets of disjoint unit disks, a slightly different analysis leads to an approximation factor of $(2/3 - \sqrt{3}/2\pi) \approx 0.39$ as the following theorem shows.

Theorem 4.6 *Let* $A = \{A_1, \ldots, A_n\}$ *and* $B = \{B_1, \ldots, B_m\}$ *each be a set of disjoint unit disks. A translation that puts as many centers of disks of* A *as possible inside or on the boundary of disks of* B *realizes at least* $(2/3 - \sqrt{3}/2\pi) \approx 0.39$ *of the maximum possible overlap under translations. The approximation factor is tight.*

Proof: We follow the general approach presented in Section 4.4 and define the kernel of a unit disk to be the disk itself with its center being its reference point. Each of the k regions $\mathcal{K}_{ij}$ is a unit disk and they are all contained in a region $\mathcal{R}$ that is a disk of radius 3. The center of $\mathcal{R}$ corresponds to $\vec{t}_{\mathrm{opt}}$.

The distance of the center of each disk $\mathcal{K}_{ij}$ to the center of $\mathcal{R}$ equals the distance between the centers of the disks in the corresponding overlapping pair (A_i, B_j) at $\vec{t}_{\mathrm{opt}}$. We draw a unit disk, denoted as W, centered at $\vec{t}_{\mathrm{opt}}$. Then $V(\mathcal{K}_{ij} \cap W) = V(A_i(\vec{t}_{\mathrm{opt}}) \cap B_j)$ and the sum of all these areas gives exactly $\mathcal{V}(\vec{t}_{\mathrm{opt}})$. This is shown in Figure 4.10: every 'shadowed' area is part of one or more terms in the sum. The darker the area, the more the disks $\mathcal{K}_{ij}$ that cover

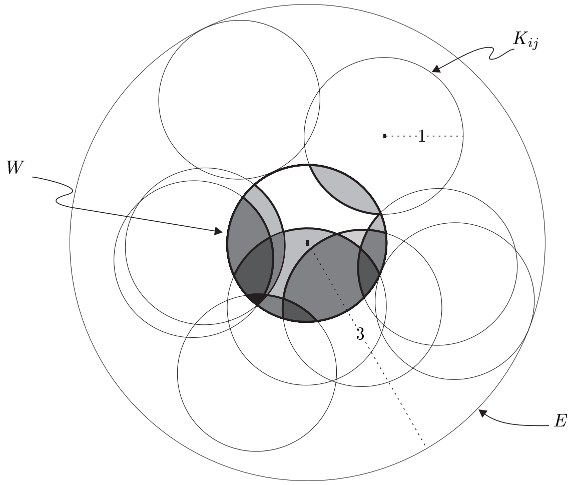


Figure 4.10: Disks $\mathcal{K}_{ij}$ are confined within an area of 9π and the area of their intersection with disk W sums up to $\mathcal{V}(\vec{t}_{\text{opt}})$.

it; an area is counted exactly as many times as its 'depth', that is, we sum up the areas with multiplicities and that's exactly what we need to construct $\mathcal{V}(\vec{t}_{\text{opt}})$ inside W since the disks in $A \cup B$ are disjoint.

Thus, there is a point in W that is covered by at least $\mathcal{V}(\vec{t}_{\text{opt}})/\pi$ disks $\mathcal{K}_{ij}$. Hence, there is a translation $\vec{t}_\sharp$ which will bring at least $\mathcal{V}(\vec{t}_{\text{opt}})/\pi$ pairs of disks (A_i, B_j) into overlap of area at least $(2\pi/3 - \sqrt{3}/2)$ each. Thus, for this translation, A and B have an overlap of area at least $(2/3 - \sqrt{3}/2\pi)\mathcal{V}(\vec{t}_{\text{opt}})$.

It can be easily seen that this approach cannot give a better approximation guarantee: consider two sets A, B with one unit disk each. The maximum area of overlap is π and a translation that brings the center of one disk on the boundary of the other has maximum depth in the arrangement of the regions $\mathcal{K}_{ij}$; this translation gives an area of overlap of $(2\pi/3 - \sqrt{3}/2)$ but $(2\pi/3 - \sqrt{3}/2)/\pi$ is exactly the algorithm's approximation factor. ◻

This approximation scheme generalizes in a straightforward way to the case of sets of disjoint unit point symmetric convex objects. Using Lemma 4.4, see next page, an approximation factor of 0.25 can be easily shown; this is tight as an example of two sets of one unit square each easily shows.

We continue by discussing on another constant-factor approximation scheme. We can parameterize the distance between the centers of two unit disks as $d = 2 - r$, $0 \leq r \leq 2$. A straightforward generalization of the lower bound technique presented in Section 4.4 shows that a translation that brings as many centers of disks in A as possible into distance of at most d from the centers of disks in B gives an approximation factor of

$$[(2-r)^2/(4-r)^2][2cos^{-1}(1-r/2) - (2-r)/2\sqrt{4-(2-r)^2}].$$

This function observes its maximum at $r \approx 1.023$, as a simple numerical analysis shows, and gives 0 for $r = 2$. The latter can be easily justified: Take the two input sets of Figure 4.3; there can be no more than one concurrent center alignment positions and since any such translation is valid in this approach, the one that aligns the center of the first disk of A with the center of the last disk of B can be arbitrarily bad. Naturally, the following question arises: Is there a center alignment translation that gives a non-zero approximation factor? The answer is positive as the following theorem (generalized to sets of convex homothets), of rather theoretical importance, shows.

Theorem 4.7 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$ each be a set of homothets of a convex object C in $\mathbb{R}^2$, such that (i) the overall depth is β (ii) the scaling ratio is λ, for some constants $\beta, \lambda > 0$. Let $\vec{t}_{i\to j}$ be the translation that aligns the centers of symmetry of the largest enclosed ellipses of two objects A_i and B_j. There exists a constant α, $0 < \alpha < 1$, such that for any two such sets A, B, there exists a pair of homothets (A_i, B_j), for which $\mathcal{V}(\vec{t}_{i\to j}) \geq \alpha \mathcal{V}(\vec{t}_{\text{opt}})$.*

We first need the following lemma:

Lemma 4.4 *Let S_1, S_2 be two translates of a compact, planar, point symmetric convex object S. For all translations $\vec{t} \in \mathbb{R}^2$ such that the center of symmetry of $S_1(\vec{t})$ is inside or on the boundary of S_2, $V(S_1(\vec{t}) \cap S_2) \geq V(S)/4$.*

Proof: Since both objects are convex and point symmetric we need to check only the translations $\vec{t}$ for which the center $c_{S_1(\vec{t})}$ of $S_1(\vec{t})$ lies on the boundary of S_2, for any move of $c_{S_1(\vec{t})}$ towards the interior of S_2 can only increase the overlap. Note that any such translation puts also the center c_{S_2} of S_2 on the boundary of $S_1(\vec{t})$, see Figure 4.11. It is easy to see that for any such translation, $S_1(\vec{t}) \cap S_2$ is a convex, point symmetric object with its point of symmetry c lying in the middle of the segment $\overline{c_{S_1(\vec{t})} c_{S_2}}$. We call the *width*, $w_{\vec{h}(S)}$ of a point symmetric object S with respect to some direction $\vec{h}$, the length of the straight line segment $h \cap S$, where h is the straight line passing through the center of symmetry with direction $\vec{h}$. We draw an arbitrary line h passing through c and a line h_1 passing through $c_{S_1(\vec{t})}$ with the same direction as h; since $S_1(\vec{t}) \cap S_2$ is convex and symmetric, $w_{\vec{h}}(S_1(\vec{t}) \cap S_2) = |\overline{e_1 e_2}| \geq |\overline{c_{S_1(\vec{t})} e_3}| = w_{\vec{h}}(S_1)/2$. Since the later holds for any direction $\vec{h}$, $V(S_1(\vec{t}) \cap S_2) \geq V(S_1)/4$. □

We can now start the proof of Theorem 4.7.

Proof: We define the kernel C' of C to be the largest inscribed ellipse of C and its reference point r_C to be the center of symmetry of C'. As mentioned in Section 4.2, we assume that C is the smallest object in $A \cup B$. The kernel of every $\mathcal{C} \in A \cup B$ is identical to the kernel of C. The reference point of every homothet is the center of symmetry of its kernel and is placed on the center of symmetry of the largest inscribed ellipse of the homothet. A classic result in convex geometry gives that $V(C') \geq V(C)/4 = 1/4$, see for example John's Theorem in Matoušek [91, Chapter 13].

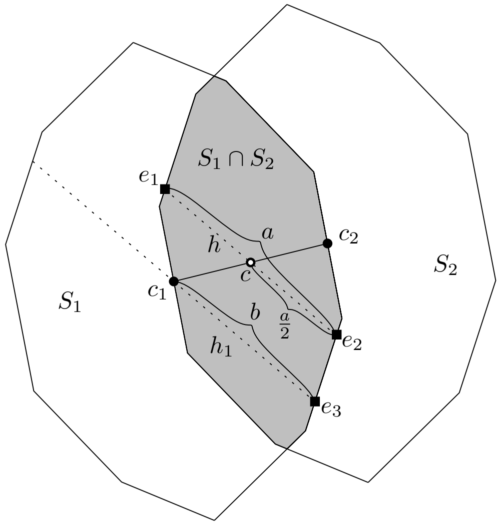


Figure 4.11: Two point symmetric sets $S_1(\vec{t})$ and S_2 with the point of symmetry of one on the boundary of the other.

As before, we assume that there are k pairs $A_i(\vec{t}_{\text{opt}})$ and B_j such that $V(A_i(\vec{t}_{\text{opt}}) \cap B_j) \neq 0$. For every such pair we consider the translations that bring the reference point of $A_i(\vec{t}_{\text{opt}})$ into the kernel of B_j; these form a region $\mathcal{K}_{ij}$ in configuration space that is an ellipse. Similarly to the proof of Theorem 4.4, all these k regions $\mathcal{K}_{ij}$ can be shown to be contained in a region $\mathcal{R}$ in configuration space with $V(\mathcal{R}) \leq 13\lambda$.

We are now interested only in the centers of the regions $\mathcal{K}_{ij}$ that correspond to the translations $\vec{t}_{i\to j}$. We will show that there exists such a translation for which the number of reference point-kernel incidences is $\Theta(k)$.

Imagine that all $\mathcal{K}_{ij}$ regions are scaled down by $1/2$ with their center as the center of scaling. Then, by a simple volume argument, there must be a point in $\mathcal{R}$ covered by at least $k/16 \cdot 1/(13\lambda)$ regions $\mathcal{K}_{ij}$. Since all these $k/(208\lambda)$ regions $\mathcal{K}_{ij}$ are copies of the same point symmetric object and have a non-empty intersection, by scaling up all regions $\mathcal{K}_{ij}$ to their normal size, we make sure that there are at least $k/16 \cdot 1/13\lambda$ centers of symmetry each covered by at least $k/16 \cdot 1/(13\lambda)$ regions $\mathcal{K}_{ij}$. These centers of symmetry correspond to translations $\vec{t}_{i\to j}$. Using the facts that $\mathcal{V}(\vec{t}_{\text{opt}}) \leq k\lambda$ an that the area of overlap is double-counted at most β^2 times, and using Lemma 4.4, we conclude that there exists a translation $\vec{t}_{i\to j}$ that brings the sets A and B into overlap of at least $\mathcal{V}(\vec{t}_{\text{opt}})/(832\lambda^2\beta^2)$. □

The approximation factor α we have computed is too small for practical purposes even for the case of sets of disks; an interesting open question is whether this factor can be improved.

4.6 An $(1-\epsilon)$-approximation algorithm

In this section we give an algorithm that computes a translation $\vec{t}_{\text{apx}}$ that gives an overlap of at least $(1-\epsilon)$ times the maximum overlap at $\vec{t}_{\text{opt}}$, for any input parameter $\epsilon > 0$. We first give a detailed description of the algorithm for the case of sets of convex homothets; the case of fat objects is shortly described afterwards.

The basic idea is simple: we impose a grid over the configuration space, compute the overlap at candidate grid points, and return the translation that corresponds to the grid point giving the maximum overlap over all grid points. We refine this simple idea in two ways in order to guarantee both the approximation bound and an efficient running time.

The first refinement is that we do not use an axis parallel uniform grid of spacing $\Theta(\epsilon)$, as would be the straightforward approach. Instead, we define the grid as follows: we determine a rectangular bounding box of a largest object in A and B (its area is bounded by a constant λ) with the longest side parallel to the diameter of the object. The grid cells (in configuration space) are scaled versions of the bounding box, scaled with a factor $c\epsilon$, with the constant $c = 1/(3328\lambda^2\beta^2)$ (recall that β denotes the maximum depth of the sets A and B).

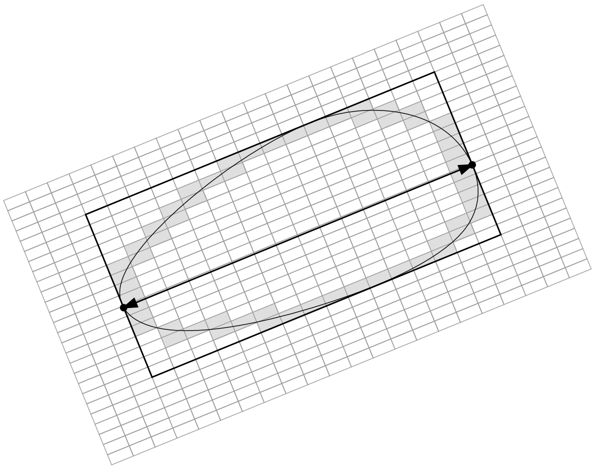


Figure 4.12: Overlaying a non-uniform orthogonal grid over a convex object.

Overlaying a grid with similar dimensions and orientation over the objects in work space gives us the following lemma:

Lemma 4.5 *The total area of the grid cells intersected by the boundary of any object in* $A(\vec{t})$ *or* B *is at most* $8\lambda c\epsilon$.

Proof: Because of the definition of the orientation of the bounding box of the largest object, its area is at most 2λ. Hence, the area of a single grid cell is at most $2\lambda(c\epsilon)^2$. The bounding box of *any* object in A or B has at most $1/c\epsilon$ grid cells along each of its axes. Now imagine that we rotate our input such that the grid becomes axis parallel. If we trace the boundary of any object, then a step from one grid cell to the next one is either horizontal or vertical (a diagonal step can be treated as a horizontal step followed by a vertical step). Because our input objects are convex, it follows that if we trace the complete boundary of an object, then the total number of grid cells visited is at most $4/c\epsilon$. Hence, the total area of these cells is $4/c\epsilon \cdot 2\lambda(c\epsilon)^2 = 8\lambda c\epsilon$. ◻

Now suppose that we move A from $\vec{t}_{\text{opt}}$ to the nearest grid point in configuration space, g. For some of the pairs $A_i(\vec{t})$ and B_j that overlap at $\vec{t}_{\text{opt}}$, the area of overlap may decrease if we move A_i to g. We refer to the difference $V(A_i(\vec{t}_{\text{opt}}) \cap B_j) - V(A_i(g) \cap B_j)$ as the *loss* of the pair. An easy upper bound on the loss of a pair can be derived using Lemma 4.5: the area that is lost when we move A_i from $\vec{t}_{\text{opt}}$ to g is completely covered by the grid cells intersected by the boundary of $A_i(\vec{t}_{\text{opt}})$ plus the next layer of grid cells in the interior of $A_i(\vec{t}_{\text{opt}})$. The number of grid cells in this second layer is at most the number of grid cells intersected by the boundary of $A_i(\vec{t}_{\text{opt}})$, and so by Lemma 4.5 the loss of the pair is at most $16\lambda c\epsilon$.

Lemma 4.6 *For the choice of the grid in configuration space as defined above and the translation g in the grid nearest to $\vec{t}_{\text{opt}}$, $\mathcal{V}(g) \geq (1-\epsilon)\mathcal{V}(\vec{t}_{\text{opt}})$.*

Proof: If we denote the number of pairs $A_t(\vec{t}_{\text{opt}}), B_j$ that contribute to the area of overlap at $\vec{t}_{\text{opt}}$ with k, then according to Theorem 4.4, $\mathcal{V}(\vec{t}_{\text{opt}}) = \Omega(k)$. A precise evaluation of the constant hidden in the Ω-notation shows that $\mathcal{V}(\vec{t}_{\text{opt}}) \geq k/(208\lambda\beta^2)$, where β is the maximum depth of A and B. Moving from $\vec{t}_{\text{opt}}$ to g, we lose an area of at most $16\lambda c\epsilon$ for each of the k pairs, so we have $\mathcal{V}(g) \geq \mathcal{V}(\vec{t}_{\text{opt}}) - 16k\lambda c\epsilon \geq \mathcal{V}(\vec{t}_{\text{opt}}) - 3328\lambda^2\beta^2 c\epsilon\mathcal{V}(\vec{t}_{\text{opt}})$. With $c = 1/(3328\lambda^2\beta^2)$, the right hand side of the latter equation becomes $(1-\epsilon)\mathcal{V}(\vec{t}_{\text{opt}})$. ◻

A straightforward approach to computing a $(1-\epsilon)$ approximation of the maximum area of overlap would be the following: for each pair A_i and B_j, we compute the region T_{ij} in configuration space; this is the region of all translations $\vec{t} \in \mathcal{T}$ for which the overlap of $A_i(\vec{t})$ and B_j is non-zero. Overlaying a grid over configuration space as explained above, we have that every such region contains $O(1/\epsilon^2)$ grid points. Thus, in total we have $O(nm/\epsilon^2)$ grid points. We can compute $V(\bigcup A_i)$ in $O(n \log^2 n)$ time, by computing a vertical decomposition of $\bigcup A_i$ (see steps 1, 2 of the algorithm below) and then summing up the areas of all the $O(n)$ cells in the decomposition (the latter can be done in $O(n)$ time since each cell has $O(1)$ complexity and thus its area can be computed in constant time). Similarly we compute $V(\bigcup B_j)$ in $O(m \log^2 m)$ time and for each grid point g $V((\bigcup A_i(g)) \cup (\bigcup B_j))$ in $O((n+m)\log^2(n+m))$ time. It follows that for each grid point g we can compute $V((\bigcup A_i(g)) \cap (\bigcup B_j))$ in $O((n+m)\log^2(n+m))$ time. If we simply keep track of the maximum area of overlap (and the corresponding translation) computed over all grid points, the whole algorithm runs in $O((nm^2 \log^2 m)/\epsilon^2)$ time.

We can, however, do better by employing a "voting scheme"; this is our second refinement of the simple idea presented in the beginning of this section. Algorithm TRANSLATION(A, B, ϵ) is given in Figure 4.13.

```
TRANSLATION$(A, B, \epsilon)$:
  1. Compute the boundaries of $\bigcup A_i$ and $\bigcup B_j$.
  2. Compute the vertical decompositions $\mathcal{VD}(A)$ of $\bigcup A_i$ and $\mathcal{VD}(B)$ of $\bigcup B_j$.
  3. Initialize an empty binary search tree $\mathcal{S}$ with entries of the form $(\vec{t}, \mathcal{V}(\vec{t}))$
     where $\vec{t}$ is the key. Let $G$ be a grid on configuration space, as explained
     above.
  4. For each pair of cells $c_i \in \mathcal{VD}(A)$ and $c_j \in \mathcal{VD}(B)$ do:
       (a) Determine all grid points $\vec{t}_\mathrm{g} \in G$ such that $\vec{t}_\mathrm{g} \in c_i \ominus c_j$.
           For each such $\vec{t}_\mathrm{g}$ do:
             • Let $\mathcal{V}_{ij}(\vec{t}_\mathrm{g})$ be the area of $c_i(\vec{t}_\mathrm{g}) \cap c_j$.
             • If $\vec{t}_\mathrm{g}$ is in $\mathcal{S}$, then $\mathcal{V}(\vec{t}_\mathrm{g}) := \mathcal{V}(\vec{t}_\mathrm{g}) + \mathcal{V}_{ij}(\vec{t}_\mathrm{g})$;
               otherwise, insert $\vec{t}_\mathrm{g}$ in $\mathcal{S}$ with $\mathcal{V}(\vec{t}_\mathrm{g}) := \mathcal{V}_{ij}(\vec{t}_\mathrm{g})$.
  5. Report the grid point $\vec{t}_\mathrm{apx}$ that maximizes $\mathcal{V}(\vec{t}_\mathrm{g})$.
```

Figure 4.13: Algorithm TRANSLATION(A, B, ϵ).

Since the objects in $A \cup B$ are homothets, which are pseudodisks, the boundaries of $\bigcup A_i$ and $\bigcup B_j$ have $O(n)$ size and can be computed in $O(n \log^2 n)$ time [85]. The vertical decompositions $\mathcal{VD}(A)$ and $\mathcal{VD}(B)$ have $O(n)$ (resp. $O(m)$) cells, and can be constructed in $O(n \log n)$ (reps. $O(m \log m)$) time [53]. Note that each cell in the decompositions has at most two vertical walls, and at most two segments that are part of the boundary of an input object. Since the base object C has constant complexity boundary, the cells in the trapezoidal decompositions are sufficiently simple to determine the grid translations for which two cells overlap in time proportional to the number of translations returned. For the same reason, we can compute the area of overlap of two cells $c_i(\vec{t}) \cap c'_j$ at a given translation $\vec{t}$ in constant time. Since the total number of translations tested is $O(mn/\epsilon^2)$, the size of $\mathcal{S}$ is $O(mn/\epsilon^2)$, and the operations on $\mathcal{S}$ take $O((mn/\epsilon^2) \log(mn/\epsilon))$ time in total. Hence the algorithm runs in $O((mn/\epsilon^2) \log(mn/\epsilon))$ time. Together with Lemma 4.6, this gives us the following theorem:

Theorem 4.8 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$ each be a set of homothets of a convex object C in $\mathbb{R}^2$, such that (i) the overall depth is β and (ii) the scaling ratio is λ, for some constants $\beta, \lambda > 0$. Let $\vec{t}_\mathrm{opt}$ be the translation that maximizes the area of overlap $\mathcal{V}(\vec{t})$ of $A(\vec{t})$ and B over all possible translations $\vec{t}$ of the set A. For any given $\epsilon > 0$, TRANSLATION(A, B, ϵ) computes a translation $\vec{t}_\mathrm{apx}$ such that $\mathcal{V}(\vec{t}_\mathrm{apx}) \geq (1-\epsilon)\mathcal{V}(\vec{t}_\mathrm{opt})$ in $O((mn/\epsilon^2) \log(mn/\epsilon))$ time.*

In the case where the two sets A and B consist of α-fat objects, imposing a uniform grid of spacing $\Theta(\epsilon)$ will do, since there are no 'skinny' objects in work space. We use a spacing of $c\epsilon$ where $c = \sqrt{2\lambda\pi\alpha}/(72\lambda^2\alpha^4\beta^2)$. First, note that, for the k overlapping pairs at $\vec{t}_{\text{opt}}$, $V(A_i(\vec{t}_{\text{opt}}) \cap B_j) \neq 0$ implies that $V(D'_{A_i(\vec{t}_{\text{opt}})} \cap D'_{B_j}) \neq 0$. Thus, when moving A from $\vec{t}_{\text{opt}}$ to the nearest grid point g the loss per pair of objects is bounded from above by the loss of their corresponding smallest enclosing disks. Since for any object $\mathcal{C} \in A \cup B$, $V(D'_{\mathcal{C}}) \leq \lambda\alpha$, a simple calculation gives that the loss per pair is at most $c\epsilon\sqrt{2\lambda\alpha/\pi}$. According to Theorem 4.5, $\mathcal{V}(\vec{t}_{\text{opt}}) \geq k/c'$ where $c' = 36\lambda\alpha^3\beta^2$. In total, $\mathcal{V}(g) \geq \mathcal{V}(\vec{t}_{\text{opt}}) - c\epsilon k\sqrt{2\lambda\alpha/\pi} \geq \mathcal{V}(\vec{t}_{\text{opt}}) - cc'\epsilon\sqrt{2\lambda\alpha/\pi}\mathcal{V}(\vec{t}_{\text{opt}})$. Setting c, c' to their actual values, the right hand side of the latter equation becomes $(1-\epsilon)\mathcal{V}(\vec{t}_{\text{opt}})$.

As before, each region T_{ij} in configuration space contains $O(1/\epsilon^2)$ grid points and hence we have $O(nm/\epsilon^2)$ grid points in total. The union of a collection of n fat convex objects in the plane has $O(n^{1+\gamma})$ complexity, for any arbitrarily small $\gamma > 0$ [58] and can be computed in $O(n^{1+\gamma}\log^2 n)$ time [2]. A vertical decomposition of such a union into $O(n^{1+\gamma})$ simple cells can be computed in $O(n^{1+\gamma}\log n)$ time [2]. The algorithms proceeds exactly as described above. The running time of the algorithm is dominated by its last step which takes $O((nm)^{1+\gamma} + (mn/\epsilon^2)\log(mn/\epsilon))$ time that becomes $O((mn/\epsilon^2)\log(mn/\epsilon))$ for $\gamma < \log_{nm}(1/\epsilon)$.

We have proved the following:

Theorem 4.9 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$ each be a set of α-fat objects in $\mathbb{R}^2$, for some fixed $\alpha > 1$, such that (i) the overall depth is β and (ii) the scaling ratio is λ, for some constants $\beta, \lambda > 0$. Let $\vec{t}_{\text{opt}}$ be the translation that maximizes the area of overlap $\mathcal{V}(\vec{t})$ of $A(\vec{t})$ and B over all possible translations $\vec{t}$ of the set A. Then, for any given $\epsilon > 0$, a translation $\vec{t}_{\text{apx}}$ such that $\mathcal{V}(\vec{t}_{\text{apx}}) \geq (1-\epsilon)\mathcal{V}(\vec{t}_{\text{opt}})$ can be computed in $O((mn/\epsilon^2)\log(mn/\epsilon))$.*

4.7 Concluding remarks

The main result of this chapter is an efficient $(1-\epsilon)$-approximation algorithm for computing the maximum area of overlap under translation of two sets of either convex homothets or fat objects in two dimensions.

Theorem 4.4 can be generalized to three dimensions for sets of homothets in a straightforward way. For sets of spheres the approximation algorithm generalizes as well in the following way: the boundary of the union of n spheres (under the assumptions (i) and (ii) of Section 4.2, page 76) is $O(n)$ and can be computed in $O(n\log n)$ time [74]. There exists a decomposition of the union into $O(n)$ simple cells that can be computed in $O(n\log n)$ time [74]. By using these cells in the voting scheme, the running time of the algorithm is $O((mn/\epsilon^3)\log(mn/\epsilon))$.

An important direction for further research is looking into transformations other than translations only, most notably rotations. In the next chapter we will see that this problem is not trivial and we will present several approximation algorithms for rigid motions.

Depending on the particular matching applications, an approximation of the maximum area of overlap does not necessarily guarantee a good match; an alternative measure could be the area of the symmetric difference of the two unions. Note that an approximation of the maximum area of overlap does not result in an approximation of the minimum area of symmetric difference. Unfortunately, our ideas do not work for the area of symmetric difference since, for this case, the lower bound theorem in Section 4.4 does not hold: for some problem instances the symmetric difference is empty. Moreover, for such instances any constant factor or $(1 + \epsilon)$-approximation algorithm must give exact results, and this seems to make this problem harder than the one we studied.

Finally, it would be worthwhile to study these problems in a more general context, i.e., with sets of arbitrary objects instead of homothets and fat objects.

Chapter 5

Maximizing the area of overlap of two unions of disks under rigid motions

5.1 Introduction

In the previous chapter, we studied the problem of maximizing the area of overlap of two unions of convex objects under translations. We mainly focused on generalizing the type of objects for which our simple $(1-\epsilon)$-approximation scheme holds. In this chapter, we expand the transformation group to rigid motions. We will see that similar ideas to those given in the previous chapter apply to the case of unions of disks and rigid motions as well, resulting in an initial $(1-\epsilon)$-approximation scheme of relatively high complexity. Given this, our research focus is different now: we deal only with unions of disks under the same conditions of bounded scaling ratio and bounded depth as before, and we present other deterministic and randomized algorithms that improve over the initial result. Moreover, for simplicity, we omit the tedious calculations of the constant factors involved in the running times of the algorithms.

5.1.1 Results

Our contributions are the following. First, in Section 5.2 ,we show that the maximum number of combinatorially distinct rigid motions of A with respect to B is $O(n^3m^2)$. As in the case of translations, the function describing the area of overlap is quite complex, even for combinatorially equivalent placements. Therefore, we focus again on approximation algorithms. Our algorithms are given in the remaining sections. For the sake of clarity we describe the algorithms for the case of disjoint unit disks. It is not hard to adapt them to sets of disks satisfying the assumptions of bounded scaling ratio and depth; the necessary changes are described in Section 5.6. For any $\epsilon > 0$, our algorithms can compute a $(1-\epsilon)$-approximation of the optimum overlap. First, we present a deterministic algorithm which runs in $O((n^2m^2/\epsilon^3)\log m)$ time. If Δ is the diameter of set A—recall that we are dealing with unit disks—the running time of the latter becomes $O((m^2n^{4/3}\Delta^{1/3}/\epsilon^3)\log n \log m)$, which yields an improvement when $\Delta = o(n^2/\log^3 n)$. Note that in many applications the union will be connected, which implies that the diameter will be $O(n)$. If the area of overlap is a constant fraction of the area of the union of A we can get a probabilistic algorithm that runs in $O((m^2/\epsilon^4)\log(m/\epsilon)\log^2 m)$ time, and succeeds with high probability. This algorithm uses the technique by Cheong et al. [39] for the rotational part. We will see that a direct application of this technique to rigid motions gives an $O((m^2/\epsilon^6)\log(m/\epsilon)\log^3 m)$ time algorithm that requires the computation of intersection points of algebraic curves, which is not very practical. Our algorithm avoids that complication.

Our algorithms for rigid motion are based on a simple two-step framework in which an approximation of the best translation is followed by an approximation of the best rotation. This way, we first achieve an absolute error on the optimum, which we then turn into a relative error using the lower bound theorem for rigid motions given in Section 5.2. The deterministic algorithm employs a clever sampling of transformation space, directed by some special properties of the function of the area of overlap of two disks. The probabilistic algorithm is a combination of sampling of translation space using a uniform grid, random sampling of both input sets, and the technique by Cheong et al..

5.2 Basic properties of the overlap function

We start by introducing some notation. Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$, be two sets of disjoint unit disks in the plane, with $n \leq m$. We consider the disks to be closed. We consider B to be fixed, while A can be translated and/or rotated relative to B.

Let $\mathcal{I}$ be the infinite set of all possible rigid motions—also called isometries—in the plane; we call $\mathcal{I}$ the *configuration space*. For convenience, we denote by R_θ a rotation about the origin by some angle $\theta \in [0, 2\pi)$ and by $T_{\vec{t}}$ a translation by some $\vec{t} \in \mathbb{R}^2$. It will be also convenient to model the space $[0, 2\pi)$ of rotations by points on the circle S^1. For simplicity, rotated only versions of A are denoted by $A(\theta) = \{A_1(\theta), \ldots, A_n(\theta)\}$. Any rigid motion $I \in \mathcal{I}$ can be uniquely defined as a translation followed by a rotation, that is, $I = I_{\vec{t},\theta} = R_\theta \circ T_{\vec{t}}$, for some $\theta \in S^1$ and $\vec{t} \in \mathbb{R}^2$. Alternatively, a rigid motion can be seen as a rotation followed by some translation; it will be always clear from the context which definition is used. In general, transformed versions of A are denoted by $A(\vec{t},\theta) = \{A_1(\vec{t},\theta), \ldots, A_n(\vec{t},\theta)\}$ for some $I_{\vec{t},\theta} \in \mathcal{I}$.

Let $\mathcal{V}_{ij}(\vec{t},\theta) = V(A_i(\vec{t},\theta) \cap B_j)$. The *area of overlap* of $A(\vec{t},\theta)$ and B, as $\vec{t},\theta$ vary, is a function $\mathcal{V} : \mathcal{I} \to \mathbb{R}$ with $\mathcal{V}(\vec{t},\theta) = V((\bigcup A(\vec{t},\theta)) \cap (\bigcup B))$. Thus the problem that we are studying can be stated as follows:

Given two sets A, B, defined as above, compute a rigid motion $I_{\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}}$ that *maximizes* $\mathcal{V}(\vec{t}, \theta)$.

Let $d_{ij}(\vec{t}, \theta)$ be the Euclidean distance between the centers of $A_i(\vec{t}, \theta)$ and B_j. For simplicity, we write $\mathcal{V}(\vec{t}), \mathcal{V}_{ij}(\vec{t}), d_{ij}(\vec{t})$ when θ is fixed and $\mathcal{V}(\theta), \mathcal{V}_{ij}(\theta), d_{ij}(\theta)$ when $\vec{t}$ is fixed. Also, let r_i be the Euclidean distance of A_i's center to the origin.

Theorem 5.1 *Let* A *be a set of* n *disjoint unit disks in the plane, and* B *a set of* m *disjoint unit disks, with* $n \leq m$. *The maximum number of combinatorially distinct rigid motions of* A *with respect to* B *is* $O(n^3m^2)$.

Proof: Let us assume for a moment that A is first rotated about the origin by some fixed angle $\theta \in [0, 2\pi)$. We define $T_{ij}(\theta) = B_j \ominus A_i(\theta)$; $\mathcal{V}_{ij}(\vec{t}, \theta) > 0$ if and only if $\vec{t} \in \mathrm{Int}(T_{ij}(\theta))$. Let $T(A, B)(\theta) = \{T_{ij}(\theta) : A_i \in A \text{ and } B_j \in B\}$. Then, $\mathcal{V}(\vec{t}, \theta) > 0$ if and only if $\vec{t} \in \mathrm{Int}(T(A, B)(\theta))$. The boundaries of the Minkowski differences $T_{ij}(\theta) \in T(A, B)(\theta)$ induce a planar subdivision $\mathcal{T}(\theta)$. Each cell in this arrangement is a set of combinatorially equivalent translations of $A(\theta)$ relative to B, that is, the set of all overlapping pairs $(A_i(\vec{t}, \theta), B_j)$ is the same for all $\vec{t}$ in the cell. As we have seen in Section 4.3.1, $\mathcal{T}(\theta)$ can be non-simple and non-connected, and its maximum complexity is $\Theta(n^2m)$ [52].

As θ varies, the combinatorial structure of $\mathcal{T}(\theta)$ changes: each $T_{ij}(\theta)$ rotates about the center of B_j and, as a result, new cells are created or existing cells disappear. Such a change occurs in one of the following two cases: (i) when two arcs in $\mathcal{T}(\theta)$ become tangent at some θ (double event) or (ii) three arcs in $\mathcal{T}(\theta)$ intersect at a point (triple event). By the analysis of Chew et al. [41], the number of double events is $O(n^2m^2)$ and the number of triple events is $O(n^3m^2)$. Thus, the complexity of the configuration space is $O(n^3m^2)$ as well. ◻

This theorem implies that explicitly computing the subdivision of the configuration space into cells with combinatorially equivalent placements is highly expensive. Moreover, the computation for rigid motions can cause non-trivial numerical problems since it requires the computation of intersection points between algebraic curves of degree six and circles [11]. Finally, as in the case of translations-only, the optimization problem in a cell of this decomposition is far from easy: one has to maximize a function consisting of a linear number of terms. Therefore we turn our attention again to approximation algorithms. Since any rigid motion can be uniquely defined by a rotation followed by some translation, the lower bound theorem of Section 4.4 holds also for the optimal rigid motion. Hence, we have the following:

Theorem 5.2 *Let* $A = \{A_1, \ldots, A_n\}$ *and* $B = \{B_1, \ldots, B_m\}$ *be two sets of disjoint unit disks in the plane. Let* $I_{\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}}$ *be a rigid motion that maximizes* $\mathcal{V}(\vec{t}, \theta)$. *If* k_{opt} *is the number of overlapping pairs* $A_i(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}), B_j$, *then* $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ *is* $\Theta(k_{\mathrm{opt}})$.

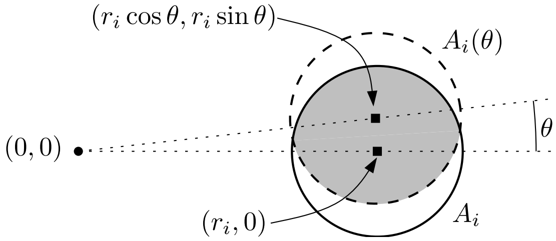


Figure 5.1: Notation in Lemma 5.1. The area of the grey region corresponds to $v(\theta)$.

5.3 The rotational case

This section considers the following restricted scenario: set B is fixed, and set A can be rotated around the origin. This will be used in the next section, where we consider general rigid motions.

Observe that this problem has a one-dimensional configuration space: the angle of rotation. Consider the function $\mathcal{V} : [0, 2\pi) \to \mathbb{R}$ with

$$\mathcal{V}(\theta) := V((\bigcup A(\theta)) \cap (\bigcup B)) = \sum_{A_i \in A, B_j \in B} \mathcal{V}_{ij}(\theta).$$

For now, our objective is to guarantee an absolute error on $\mathcal{V}$ rather than a relative one. We start with a result that bounds the difference in overlap for two relatively similar rotations. Recall that r_i is the distance of A_i's center to the origin.

Lemma 5.1 *Let A_i, B_j be any fixed pair of disks. For any given $\delta > 0$ and any θ_1, θ_2 for which $|\theta_1 - \theta_2| \le \delta/(2r_i)$, we have $|\mathcal{V}_{ij}(\theta_1) - \mathcal{V}_{ij}(\theta_2)| \le 2\delta$.*

Proof: Without loss of generality, we assume that $\theta_1 = 0$ and that A_i is centered at $(r_i, 0)$ with $r_i > 0$; see Figure 5.1. We want to see that $V(A_i \cap B_j) - V(A_i(\theta) \cap B_j) \le 2\delta$ for any $0 \le \theta \le \delta/(2r_i)$. Consider the function $v(\theta) = V(A_i \cap A_i(\theta))$ with $\theta \in [0, \pi/2]$. We will prove that if $0 \le \theta \le \delta/(2r_i)$ then $v(\theta) \ge \pi - \delta$, and therefore $V(A_i \setminus A_i(\theta)) = V(A_i(\theta) \setminus A_i) \le \delta$. Using that for any sets X, Y we have $V(X) - V(Y) = V(X \setminus Y) - V(Y \setminus X)$, then for any $0 \le \theta \le \delta/(2r_i)$ it holds

$$\begin{aligned}
&|V(A_i \cap B_j) - V(A_i(\theta) \cap B_j)| \\
&\qquad = |V((A_i \cap B_j) \setminus (A_i(\theta) \cap B_j)) - V((A_i(\theta) \cap B_j) \setminus (A_i \cap B_j))| \\
&\qquad \le |V((A_i \cap B_j) \setminus (A_i(\theta) \cap B_j))| + |V((A_i(\theta) \cap B_j) \setminus (A_i \cap B_j))| \\
&\qquad \le V(A_i \setminus A_i(\theta)) + V(A_i(\theta) \setminus A_i) \\
&\qquad \le 2\delta,
\end{aligned}$$

and the lemma follows.

We will show that $v(\theta) \ge \pi - \delta$ using the mean-value theorem. The center of $A_i(\theta)$ is positioned at $(r_i \cos(\theta), r_i \sin(\theta))$ and the distance between the centers of A_i and $A_i(\theta)$ is

$$\sqrt{r_i^2(1-\cos\theta)^2 + r_i^2\sin^2\theta} = r_i\sqrt{2(1-\cos(\theta))}.$$

The area of overlap of two unit disks whose centers are d apart is

$$2\arccos\frac{d}{2} - \frac{d\sqrt{4-d^2}}{2},$$

and therefore we get

$$\begin{aligned}\frac{\partial v(\theta)}{\partial\theta} &= \frac{\partial v(\theta)}{\partial d}\cdot\frac{\partial d}{\partial\theta}\\ &= -\sqrt{4-2r_i^2(1-\cos(\theta))}\cdot\frac{r_i\sin(\theta)}{\sqrt{2(1-\cos(\theta))}}\\ &= -r_i\sqrt{2+2\cos(\theta)-r_i^2\sin^2(\theta)} \geq -2r_i,\end{aligned}$$

where in the last inequality we used $2 \geq 2\cos(\theta) - r_i^2\sin^2(\theta)$. We conclude that if $0 \leq \theta \leq \delta/(2r_i)$ then $\partial v(\theta)/\partial\theta \geq -\delta/\theta$.

Using the mean-value theorem we see that, for any $\theta \in [0, \delta/(2r_i)]$ there exists $\theta' \in [0,\theta]$ such that

$$\frac{v(\theta)-v(0)}{\theta-0} = \frac{v(\theta)-\pi}{\theta} = \frac{\partial v(\theta')}{\partial\theta}.$$

Since, $0 \leq \theta' \leq \delta/(2r_i)$, we have $\partial v(\theta')/\partial\theta \geq -\frac{\delta}{\theta}$ and so we conclude that $v(\theta) - \pi \geq -\delta$. ⊔⊓

For a pair A_i, B_j, we define the interval $R_{ij} = \{\theta \in [0, 2\pi) : A_i(\theta) \cap B_j \neq \emptyset\}$ on S^1, the circle of rotations. We denote the length of R_{ij} by $|R_{ij}|$. Instead of computing $\mathcal{V}_{ij}(\theta)$ at each $\theta \in R_{ij}$, we would like to sample it at regular intervals whose length is at most $\delta/(2r_i)$. At first, it looks as if we would have to take an infinite number of sample points as $r_i \to \infty$. However, as the following lemma shows, $|R_{ij}|$ decreases as r_i increases, and the number of samples we need to consider is bounded.

Lemma 5.2 *For any A_i, B_j with $r_i > 0$, and any given given $\delta > 0$, we have $|R_{ij}|/(\delta/(2r_i)) = O(1/\delta)$.*

Proof: Without loss of generality, we can assume that A_i is centered at $(r_i, 0)$ and B_j is centered at $(r_j, 0)$. Note that the distance between the center of $A_i(\theta)$ and B_j is

$$d_{ij}(\theta) = \sqrt{(r_i\cos\theta - r_j)^2 + (r_i\sin\theta)^2} = \sqrt{r_i^2 + r_j^2 - 2r_ir_j\cos\theta}.$$

Under these assumptions, R_{ij} is of the form $[-\theta_{ij}, \theta_{ij}]$, where θ_{ij} is the largest value for which $A_i(\theta_{ij}) \cap B_j \neq \emptyset$, that is, $d_{ij}(\theta_{ij}) = 2$. We have $\theta_{ij} = \arccos\frac{r_i^2+r_j^2-4}{2r_ir_j}$.

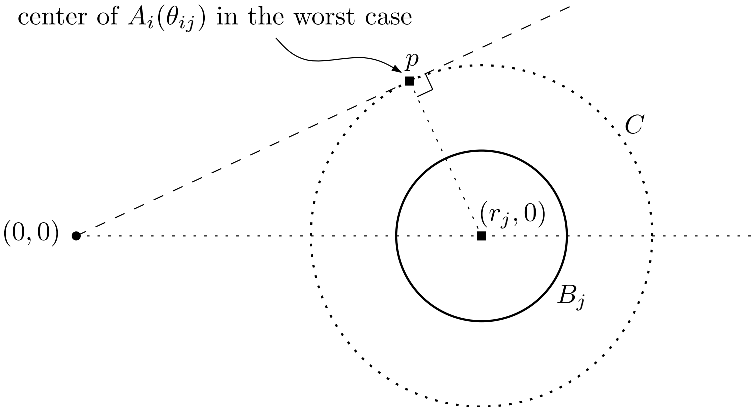


Figure 5.2: Notation in Lemma 5.2. The center of $A_i(\theta_{ij})$ is placed in the circle C. Therefore, θ_{ij} is maximized for the dashed line through the origin and tangent to C.

As shown in Figure 5.2, the center of $A_i(\theta_{ij})$ is always placed on C, the circle of radius two and concentric with B_j. Therefore, the value θ_{ij} is maximized when it equals the slope of the line through the origin and tangent to C. Let p be the point of tangency. Since the triangle $p,(0,0),(r_j,0)$ is right on p, we conclude that θ_{ij} is maximized when $r_j = \sqrt{r_i^2+4}$. Therefore

$$|R_{ij}| = 2\arccos\frac{r_i^2+r_j^2-4}{2r_ir_j} \leq 2\arccos\sqrt{1-\frac{4}{r_i^2+4}}.$$

Using L'Hôpital's rule we can compute that

$$\lim_{r_i\to\infty}\frac{|R_{ij}|}{1/r_i} = \lim_{r_i\to\infty}\frac{4}{1+\frac{4}{r_i^2}} = 4.$$

It follows that the function $|R_{ij}|\cdot r_i$ is bounded for any $r_i>0$, and so $\frac{|R_{ij}|}{\delta/(2r_i)} = O(1/\delta)$. ⊔⊓

This lemma implies that we have to consider only $O(1/\delta)$ sample rotations per pair of disks. Thus we need to check $O(nm/\delta)$ rotations in total. It seems that we would have to compute all overlaps at every rotation from scratch, but here Lemma 5.1 comes to the rescue: in between two consecutive rotations θ,θ' defined for a given pair A_i,B_j there may be many other rotations, but if we conservatively estimate the overlap of A_i,B_j as the minimum overlap of θ and θ', we do not loose too much. In Figure 5.3, algorithm ROTATION is described in more detail; the value $\tilde{\mathcal{V}}(\theta)$ is the conservative estimate of $\mathcal{V}(\theta)$, as just explained.

Lemma 5.3 *Let θ_{opt} be a rotation that maximizes $\mathcal{V}(\theta)$ and let k_{opt} be the number of overlapping pairs $A_i(\theta_{\mathrm{opt}}),B_j$. For any given $\delta>0$, the rotation*

```
ROTATION$(A, B, \delta)$:
  1. For each pair of disks $A_i \in A$ and $B_j \in B$, choose a set $\Theta_{ij} := \{\theta_{ij}^1, \dots, \theta_{ij}^{s_{ij}}\}$ of rotations as follows. First put the midpoint of $R_{ij}$ in $\Theta_{ij}$, and then put all rotations in $\Theta_{ij}$ that are in $R_{ij}$ and are at distance $k \cdot \delta/(2r_i)$ from the midpoint for some integer $k$. Finally, put both endpoints of $R_{ij}$ in $\Theta_{ij}$. In other words, $\Theta_{ij}$ consists of rotations with a uniform spacing of $\delta/(2r_i)$— except for the cases of endpoints whose distance to their neighbor rotations is less than $\delta/(2r_i)$— with the midpoint of $R_{ij}$ being one of them.
  2. Sort the values $\Theta := \bigcup_{i,j} \Theta_{ij}$, keeping repetitions and solving ties arbitrarily. Let $\theta_0, \theta_1, \dots$ be the ordering of $\Theta$. In steps 3 and 4, we will compute a value $\tilde{\mathcal{V}}(\theta)$ for each $\theta \in \Theta$.
  3. (a) Initialize $\tilde{\mathcal{V}}(\theta_0) := 0$.
     (b) For each pair $A_i \in A, B_j \in B$ for which $\theta_0 \in R_{ij}$ do:
         • If $\mathcal{V}_{ij}$ is decreasing at $\theta_0$, or $\theta_0$ is the midpoint of $R_{ij}$, then $\tilde{\mathcal{V}}(\theta_0) := \tilde{\mathcal{V}}(\theta_0) + \mathcal{V}_{ij}(\tilde{\theta}_{ij})$, where $\tilde{\theta}_{ij}$ is the closest value to $\theta_0$ in $\Theta_{ij}$ with $\tilde{\theta}_{ij} > \theta_0$.
         • If $\mathcal{V}_{ij}$ is increasing at $\theta_0$, then $\tilde{\mathcal{V}}(\theta_0) := \tilde{\mathcal{V}}(\theta_0) + \mathcal{V}_{ij}(\tilde{\theta}_{ij})$, where $\tilde{\theta}_{ij}$ is the closest value to $\theta_0$ in $\Theta_{ij}$ with $\tilde{\theta}_{ij} < \theta_0$.
  4. For each $\theta_l$ in increasing order of $l$, compute $\tilde{\mathcal{V}}(\theta_l)$ from $\tilde{\mathcal{V}}(\theta_{l-1})$ by updating the contribution of the pair $A_i, B_j$ defining $\theta_l$, as follows. Let $\theta_l$ be the $s$-th point in $\Theta_{ij}$, that is, $\theta_l = \theta_{ij}^s$
     • If $\mathcal{V}_{ij}$ is increasing at $\theta_{ij}^s$, then $\tilde{\mathcal{V}}(\theta_l) := \tilde{\mathcal{V}}(\theta_{l-1}) - \mathcal{V}_{ij}(\theta_{ij}^{s-1}) + V_{ij}(\theta_{ij}^s)$
     • If $\mathcal{V}_{ij}$ is the midpoint of $R_{ij}$, then $\tilde{\mathcal{V}}(\theta_l) := \tilde{\mathcal{V}}(\theta_{l-1}) - \mathcal{V}_{ij}(\theta_{ij}^{s-1}) + \mathcal{V}_{ij}(\theta_{ij}^{s+1})$
     • If $\mathcal{V}_{ij}$ is decreasing at $\theta_{ij}^s$, then $\tilde{\mathcal{V}}(\theta_l) := \tilde{\mathcal{V}}(\theta_{l-1}) - \mathcal{V}_{ij}(\theta_{ij}^s) + \mathcal{V}_{ij}(\theta_{ij}^{s+1})$
  5. Report the $\theta_{\text{apx}} \in \Theta$ that maximizes $\tilde{\mathcal{V}}(\theta)$.
```

Figure 5.3: Algorithm ROTATION(A, B, δ).

θ_{apx} *reported by* ROTATION*(A, B, δ) satisfies $\mathcal{V}(\theta_{\text{opt}}) - \mathcal{V}(\theta_{\text{apx}}) = O(k_{\text{opt}}\delta)$, and can be computed in $O((mn/\delta)\log m)$ time.*

Proof: First, we show that $\mathcal{V}(\theta) \geq \tilde{\mathcal{V}}(\theta) \geq \mathcal{V}(\theta) - 2k_\theta\delta$ for any $\theta \in \Theta$ where k_θ is the number of overlapping pairs between $A(\theta)$ and B. That is, $\tilde{\mathcal{V}}$ is a fair approximation of $\mathcal{V}$ from below for the values in Θ.

By checking whether $\mathcal{V}_{ij}$ increases or decreases at θ_{ij}^s and adding the appropriate value to $\tilde{\mathcal{V}}(\theta)$, each pair A_i, B_j contributes $\mathcal{V}_{ij}(\theta_{ij}^s) \leq \mathcal{V}_{ij}(\theta)$ to $\tilde{\mathcal{V}}(\theta)$ for some θ_{ij}^s for which $|\theta - \theta_{ij}^s| \leq \delta/(2r_i)$. By Lemma 5.1 we have $\mathcal{V}_{ij}(\theta) - \mathcal{V}_{ij}(\theta_{ij}^s) \leq 2\delta$. Thus, in total, $0 \leq \mathcal{V}(\theta) - \tilde{\mathcal{V}}(\theta) \leq 2k_\theta\delta$.

In a similar fashion, consider now the k_{opt} overlapping pairs of disks at θ_{opt}, and let A_M be the disk furthest from the origin that participates in the optimal solution, i.e. $A_M(\theta_{\mathrm{opt}}) \cap (\bigcup B) \neq \emptyset$. Let $\tilde{\theta} \in \Theta$ be the closest value to θ_{opt}. We have

$$|\tilde{\theta} - \theta_{\mathrm{opt}}| \leq \delta/(2r_M) \leq \delta/(2r_i)$$

for all A_i in the optimal solution. Again, according to Lemma 5.1, the loss per pair A_i, B_j is $\mathcal{V}_{ij}(\theta_{\mathrm{opt}}) - \mathcal{V}_{ij}(\tilde{\theta}) \leq 2\delta$. In total, $\mathcal{V}(\theta_{\mathrm{opt}}) - \mathcal{V}(\tilde{\theta}) \leq 2k_{\mathrm{opt}}\delta$.

Observe that since both endpoints of every interval R_{ij} are in Θ, no new pairs with non-zero overlap are formed when 'moving' from θ_{opt} to $\tilde{\theta}$. Hence, for our purpose, we can assume that $k_{\tilde{\theta}} = k_{\mathrm{opt}}$.

Putting it all together we get

$$\begin{aligned} \mathcal{V}(\theta_{\mathrm{opt}}) - \mathcal{V}(\theta_{\mathrm{apx}}) &= \big(\mathcal{V}(\theta_{\mathrm{opt}}) - \mathcal{V}(\tilde{\theta})\big) + \big(\mathcal{V}(\tilde{\theta}) - \tilde{\mathcal{V}}(\tilde{\theta})\big) + \big(\tilde{\mathcal{V}}(\tilde{\theta}) - \tilde{\mathcal{V}}(\theta_{\mathrm{apx}})\big) \\ &\quad + \big(\tilde{\mathcal{V}}(\theta_{\mathrm{apx}}) - V(\theta_{\mathrm{apx}})\big) \\ &\leq 2k_{\mathrm{opt}}\delta + 2k_{\tilde{\theta}}\delta + 0 + 0 \leq 4k_{\mathrm{opt}}\delta. \end{aligned}$$

The running time is dominated by the time to sort the values in Θ. The set Θ consists of $O(nm)$ subsets Θ_{ij}, each having $O(1/\delta)$ rotations by Lemma 5.2. Each subset Θ_{ij} can easily be generated as a sorted sequence, so what remains is to merge the sorted sequences, which can be done in $O((nm/\delta)\log m)$ time. ◻

5.4 A $(1-\epsilon)$-approximation algorithm for rigid motions

As noted in the introduction, any rigid motion can be described as a translation plus a a rotation about the origin. This fact is used in the algorithm Rigid-Motion given in Figure 5.4. First, we start with the following lemma which implies that, in terms of absolute error, it is not too bad if we choose a translation which is close to the optimal one. A proof of this lemma was already given in Section 4.6; for completeness, a simpler proof for disks is given below.

Lemma 5.4 *Let k be the number of overlapping pairs $A_i(\vec{t},\theta), B_j$ for some $\vec{t} \in \mathbb{R}^2, \theta \in [0, 2\pi)$. For any given $\delta > 0$ and any $\vec{t}' \in \mathbb{R}^2$ for which $|\vec{t} - \vec{t}'| = O(\delta)$, we have $\mathcal{V}(\vec{t}', \theta) = \mathcal{V}(\vec{t}, \theta) - O(k\delta)$.*

Proof: Consider a pair of disks $A_i(\vec{t},\theta)$ and B_j for which $\mathcal{V}_{ij}(\vec{t},\theta) \neq 0$. If A_i is translated by $\vec{t}'$, instead of $\vec{t}$, then $d_{ij}(\vec{t}',\theta) - d_{ij}(\vec{t},\theta) = |\vec{t} - \vec{t}'|$. Observe that the biggest *loss per pair*, $\mathcal{V}_{ij}(\vec{t},\theta) - \mathcal{V}_{ij}(\vec{t}',\theta)$, occurs when A_i moves in the direction of the line connecting the centers of A_i and B_j, and away from B_j. Since the diameter of both disks is equal to 2, we have that $\mathcal{V}_{ij}(\vec{t},\theta) - \mathcal{V}_{ij}(\vec{t}',\theta) < 2|\vec{t} - \vec{t}'| = O(\delta)$. We have k such pairs, hence[1] , $\mathcal{V}(\vec{t},\theta) - \mathcal{V}(\vec{t}',\theta) = O(k\delta)$. ◻

---

[1] Note that by translating A by $\vec{t}'$ instead of $\vec{t}$ and then rotating it by θ, new pairs might appear but this can only decrease the total loss.

```
RigidMotion($A, B, \epsilon$):
  1. Let $G$ be a uniform grid of spacing $c\epsilon$, where $c$ is a suitable constant. For
     each pair of disks $A_i \in A$ and $B_j \in B$ do:
     (a) Set the center of rotation, i.e. the origin, to be $B_j$'s center by translating
         $B$ appropriately.
     (b) Let $T_{ij} = B_j \ominus A_i$, and determine all grid points $\vec{t}_{\mathrm{g}}$ of $G$ such that
         $\vec{t}_{\mathrm{g}} \in T_{ij}$. For each such $\vec{t}_{\mathrm{g}}$ do:
           • run Rotation($A(\vec{t}_{\mathrm{g}}), B, c'\epsilon$), where $c'$ is an appropriate constant.
             Let $\theta^{\mathrm{g}}_{\mathrm{apx}}$ be the rotation returned. Compute $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}})$.
  2. Report the pair $(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}})$ that maximizes $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}})$.
```

Figure 5.4: Algorithm RigidMotion(A, B, ϵ).

Theorem 5.3 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$, with $n \leq m$, be two sets of disjoint unit disks in the plane. Let $I_{\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}}$ be a rigid motion that maximizes $\mathcal{V}(\vec{t}, \theta)$. Then, for any given $\epsilon > 0$,* RigidMotion(A, B, ϵ) *computes a rigid motion $I_{\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}}$ such that $\mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) \geq (1-\epsilon)\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ in $O((n^2m^2/\epsilon^3)\log m)$ time.*

Proof: We will show that $\mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}})$ approximates $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ up to an absolute error. To convert the absolute error into a relative error, and hence show the algorithm's correctness, we use again Theorem 4.4. Let A_{opt} be the set of disks in A that participate in the optimal solution and let $|A_{\mathrm{opt}}| = \bar{k}_{\mathrm{opt}}$. Since the 'kissing' number of unit open disks is six, we have that $k_{\mathrm{opt}} < 6\bar{k}_{\mathrm{opt}}$, where k_{opt} is the number of overlapping pairs in the optimal solution. Next, imagine that RigidMotion$(A_{\mathrm{opt}}, B, \epsilon)$ is run instead of RigidMotion(A, B, ϵ). Of course, an optimal rigid motion for A_{opt} is an optimal rigid motion for A and the error we make by applying a non-optimal rigid motion to A_{opt} bounds the error we make when applying the same rigid motion to A.

Consider a disk $A_i \in A_{\mathrm{opt}}$ and an intersecting pair $A_i(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$, B_j. Since, at some stage, the algorithm will use B_j's center as the center of rotation, and $I_{\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}} = R_{\theta_{\mathrm{opt}}} \circ T_{\vec{t}_{\mathrm{opt}}}$, we have that $A_i(\vec{t}_{\mathrm{opt}}) \cap B_j \neq \emptyset$ if and only if $A_i(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \cap B_j \neq \emptyset$. Hence, we have that $\vec{t}_{\mathrm{opt}} \in T_{ij}$ and the algorithm will consider some grid translation $\vec{t}_{\mathrm{g}} \in T_{ij} = B_j \ominus A_i$, for which $|\vec{t}_{\mathrm{opt}} - \vec{t}_{\mathrm{g}}| = O(\epsilon)$. By Lemma 5.4 we have $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) = O(k_{\mathrm{opt}}\epsilon) = O(\bar{k}_{\mathrm{opt}}\epsilon)$.

Let $\theta^{\mathrm{g}}_{\mathrm{opt}}$ be the optimal rotation for $\vec{t}_{\mathrm{g}}$. Then, $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) \leq \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}})$. The algorithm computes, in its second loop, a rotation $\theta^{\mathrm{g}}_{\mathrm{apx}}$ for which $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) = O(k^{g}_{\mathrm{opt}}\epsilon)$, where k^{g}_{opt} is the number of pairs at the optimal rotation $\theta^{\mathrm{g}}_{\mathrm{opt}}$ of $A_{\mathrm{opt}}(\vec{t}_{\mathrm{g}})$. Since we are only considering A_{opt} we have that $k^{g}_{\mathrm{opt}} < 6\bar{k}_{\mathrm{opt}}$, thus, $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) = O(\bar{k}_{\mathrm{opt}}\epsilon)$.

Now, using the fact that $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) \leq \mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}})$ and that $\bar{k}_{\mathrm{opt}} \leq k_{\mathrm{opt}}$, and putting it all together we get

$$\begin{aligned}\mathcal{V}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})-\mathcal{V}(\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}) &= \big(\mathcal{V}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})-\mathcal{V}(\vec{t}_{\mathrm{g}},\theta_{\mathrm{opt}})\big)\\ &\quad+\big(\mathcal{V}(\vec{t}_{\mathrm{g}},\theta_{\mathrm{opt}})-\mathcal{V}(\vec{t}_{\mathrm{g}},\theta^{\mathrm{g}}_{\mathrm{opt}})\big)\\ &\quad+\big(\mathcal{V}(\vec{t}_{\mathrm{g}},\theta^{\mathrm{g}}_{\mathrm{opt}})-\mathcal{V}(\vec{t}_{\mathrm{g}},\theta^{\mathrm{g}}_{\mathrm{apx}})\big)\\ &\quad+\big(\mathcal{V}(\vec{t}_{\mathrm{g}},\theta^{\mathrm{g}}_{\mathrm{apx}})-\mathcal{V}(\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}})\big)\\ &< O(\bar{k}_{\mathrm{opt}}\epsilon)+0+O(\bar{k}_{\mathrm{opt}}\epsilon)+0=O(k_{\mathrm{opt}}\epsilon).\end{aligned}$$

From Theorem 5.2 we have that $\mathcal{V}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})=\Theta(k_{\mathrm{opt}})$ and the approximation bound follows.

Finally, the running time of the algorithm is dominated by its first step. We can compute $\mathcal{V}(\vec{t}_{\mathrm{g}},\theta^{\mathrm{g}}_{\mathrm{apx}})$ by a simple plane sweep in $O(m\log m)$ time. Since there are $\Theta(\epsilon^{-2})$ grid point in each T_{ij}, each execution of the loop in the first step takes $O(m+1/\epsilon^2+(1/\epsilon^2)(nm/\epsilon)\log m+(1/\epsilon^2)m\log m)=O((nm/\epsilon^3)\log m)$ time. The step is executed nm times, thus the algorithm runs in $O((n^2m^2/\epsilon^3)\log m)$ time. ◻

5.4.1 An improvement for sets with small diameter

We can modify the algorithm such that its running time depends on the diameter Δ of the set A. The main idea is to convert our algorithm into one that is sensitive to the number of pairs of disks in A and B that have approximately the same distance, and then use the combinatorial bounds by Gavrilov et al. [64]. Namely, we will use the following result (note that an extra $\log n$ factor is missing in the reference due to a typographic error).

Lemma 5.5 *[64, Theorem 4.1] Given a S set of n points whose closest pair is at distance at least 2, there are $O(n^{4/3}t^{1/3}\log n)$ pairs of points in S whose distance is in the range $[t-4,t+4]$.*

This lemma and a careful implementation of Rotation allows us to improve the analysis of the running time of RigidMotion for small values of Δ. In many applications it is reasonable to assume bounds of the type $\Delta=O(n)$ [64], and therefore the result below is relevant. For example, if $\Delta=O(n)$ this result shows that we can compute a $(1-\epsilon)$-approximation in $O((m^2n^{5/3})/\epsilon^3\log n\log m)$ time.

Theorem 5.4 *Let $A=\{A_1,\dots,A_n\}$ and $B=\{B_1,\dots,B_m\}, n\le m$ be two sets of disjoint unit disks in the plane. Let Δ be the diameter of A, and let $I_{\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}}}$ be the rigid motion maximizing $\mathcal{V}(\vec{t},\theta)$. For any $\epsilon>0$, we can find in $O((m^2n^{4/3}\Delta^{1/3}/\epsilon^3)\log n\log m)$ time a rigid motion $I_{\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}}}$ such that $\mathcal{V}(\vec{t}_{\mathrm{apx}},\theta_{\mathrm{apx}})\ge(1-\epsilon)\mathcal{V}(\vec{t}_{\mathrm{opt}},\theta_{\mathrm{opt}})$.*

Proof: Observe that when RigidMotion calls Rotation the origin is set at the center of some B_j, and some A_i intersects B_j. We denote by c_{A_i} the center of A_i and similarly by c_{B_j} the center of B_j. Inside Rotation, the pairs of disks $A_{i'}$ and $B_{j'}$ such that $R_{i'j'} \neq \emptyset$ must satisfy

$$d(c_{B_j}, c_{B_{j'}}) - 4 \leq d(c_{A_i}, c_{A_{i'}}) \leq d(c_{B_j}, c_{B_{j'}}) + 4$$

We store in a balanced tree the distances from the origin to the centers of A. This can be done in $O(n \log n)$ time, and then we can report, for any given pair $B_j, B_{j'} \in B$ and any $A_i \in A$, all the pairs $A_i, A_{i'} \in A$ satisfying the relation above in $O(k_{j,j',i} + \log n)$ time, where $k_{j,j',i}$ is the number of reported pairs. Therefore, when we have fixed $\vec{t}_{\mathrm{g}} \in T_{ij}$ in RigidMotion, we can implement the call to Rotation$(A(\vec{t}_{\mathrm{g}}), B, c\epsilon)$ in $O(m \log n + \frac{\sum_{j'} k_{j,j',i}}{\epsilon} \log m)$ time.

If A has diameter Δ, then Lemma 5.5 implies $\sum_i k_{j,j',i} = O(n^{4/3}\Delta^{1/3} \log n)$. This means that, overall, RigidMotion can be implemented in time

$$O\left(\frac{1}{\epsilon^2}\sum_{i,j}\left(m \log n + \frac{\sum_{j'} k_{j,j',i}}{\epsilon} \log m\right)\right) =$$

$$O\left(\frac{nm^2 \log n}{\epsilon^2} + \frac{\sum_{i,j,j'} k_{j,j',i}}{\epsilon^3} \log m\right) =$$

$$O\left(\frac{nm^2 \log n}{\epsilon^2} + \frac{\sum_{j,j'} \sum_i k_{j,j',i}}{\epsilon^3} \log m\right) =$$

$$O\left(\frac{nm^2 \log n}{\epsilon^2} + \frac{m^2 n^{4/3} \Delta^{1/3} \log n}{\epsilon^3} \log m\right) = O\left(\frac{m^2 n^{4/3} \Delta^{1/3} \log n \log m}{\epsilon^3}\right).$$

□

5.5 A Monte Carlo algorithm

In this section we present a Monte Carlo algorithm that computes a $(1-\epsilon)$-approximation for rigid motions in $O((m^2/\epsilon^4)\log(m/\epsilon)\log^2 m)$ time. The algorithm works under the condition that the maximum area of overlap of A and B is at least some constant fraction of the area of A.

The algorithm is simple and follows the two-step framework of Section 5.4 in which an approximation of the best translation is followed by an approximation of the best rotation. However, now, the first step is a combination of grid sampling of the space of translations and random sampling of set A. This random sampling is based on the observation that the deterministic algorithm of Section 5.4 will compute a $(1-\epsilon)$-approximation k_{opt} times, where k_{opt} is the number of pairs of overlapping disks in an optimal solution. Intuitively, the larger this number is, the quicker such a pair will be tried out in the first step. Similar observations were made by Akutsu et al. [8] who gave exact Monte Carlo algorithms for the largest common point set problem.

The second step is based on a direct application of the technique by Cheong et al. that allows us to maximize, up to an absolute error, the area of overlap under rotation in almost linear time, by computing a point of maximum depth in a one dimensional arrangement.

Rotations. For a given $\epsilon > 0$, we choose a uniform random sample S of points in A with $|S| = \Theta(\epsilon^{-2} \log m)$. For a point $s \in S$, we define $W(s) = \{\theta \in [0, 2\pi) | s(\theta) \in B\}$ where $s(\theta)$ denotes a copy of s rotated by θ. Let $\Gamma_B(S)$ be the arrangement of all regions $W(s), s \in S$; it is a one-dimensional arrangement of unions of rotational intervals.

Lemma 5.6 *Let θ_{opt} be the rotation that maximizes $\mathcal{V}(\theta)$. For any given $\epsilon > 0$, let S be a uniform random sample of points in A with $|S| \geq c_1 \frac{\log m}{\epsilon^2}$ where c_1 is an appropriate constant. A vertex θ_{apx} of $\Gamma_B(S)$ of maximum depth satisfies $\mathcal{V}(\theta_{\mathrm{opt}}) - \mathcal{V}(\theta_{\mathrm{apx}}) \leq \epsilon V(A)$ with probability at least $1 - 1/m^6$.*

Proof: The proof is identical to the proof of and Lemma 4.2 by Cheong et al. [39]. $\square$

The arrangement $\Gamma_B(S)$ has $O((m/\epsilon^2) \log m)$ complexity and can be computed in $O((m/\epsilon^2) \log(m/\epsilon) \log m)$ time by sorting. A vertex θ_{apx} of $\Gamma_B(S)$ of maximum depth can be found by a simple traversal of this arrangement.

We could apply the idea above directly to rigid motions and compute the arrangement of all regions $W(s)$ with respect to rigid motions of S. Lemma 5.6 holds for this arrangement, and a vertex of maximum depth gives an absolute error on $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$. This arrangement has $O(|S|^3 m^2) = O((m^2/\epsilon^6) \log^3 m)$ vertices [41] that correspond — in workspace — to combinations of triples of points in S and triples of disks in B such that each point lies on the boundary of a disk. All such combinations can be easily found in $O((m^2/\epsilon^6) \log(m/\epsilon) \log^3 m)$ time. However, computing the actual rigid motion for any such combination is not trivial, as already explained in section 5.2. This complication is avoided by applying the technique to rotations only, thus computing a one-dimensional arrangement instead.

Rigid motions. Since we assume that $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \geq \alpha V(A)$, for some given constant $0 < \alpha \leq 1$, we have that $k_{\mathrm{opt}} \geq \alpha n$. Based also on the fact that the number of disks in A that participate in an optimal solution is at least $k_{\mathrm{opt}}/6$, we can easily prove that the probability that $\Theta(\alpha^{-1} \log m)$ uniform random draws of disks from A will all fail to give a disk participating in an optimal solution is at most $1/m^6$. Algorithm RandomRigidMotion is given in Figure 5.5.

Theorem 5.5 *Let $A = \{A_1, \ldots, A_n\}$ and $B = \{B_1, \ldots, B_m\}$, be two sets of disjoint unit disks in the plane and $I_{\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}}$ be a rigid motion that maximizes $\mathcal{V}(\vec{t}, \theta)$. Assume that $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \geq \alpha V(A)$, for some given constant $0 < \alpha \leq 1$. For any given $\epsilon > 0$, RandomRigidMotion(A, B, α, ϵ) computes a rigid motion $I_{\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}}$ such that $\mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) \geq (1 - \epsilon) \mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}})$ in*

```
RandomRigidMotion($A, B, \alpha, \epsilon$):
  1. Choose a uniform random sample $S$ of points in $A$, with $|S| = \Theta(\epsilon^{-2} \log m)$.
  2. Let $G$ be a uniform grid of spacing $c\epsilon$, where $c$ is a suitable constant.
     Repeat $\Theta(\alpha^{-1} \log m)$ times:
       (a) Choose a random $A_i$ from $A$.
       (b) For each $B_j \in B$ do:
             i. Set the center of rotation, i.e. the origin, to be $B_j$'s center by
                translating $B$ appropriately.
            ii. Let $T_{ij} = B_j \ominus A_i$, and determine all grid points $\vec{t}_{\mathrm{g}}$ of $G$ such that
                $\vec{t}_{\mathrm{g}} \in T_{ij}$. For each such $\vec{t}_{\mathrm{g}}$ do:
                  • Compute a vertex $\theta^{\mathrm{g}}_{\mathrm{apx}}$ of maximum depth in $\Gamma_B(S(\vec{t}_{\mathrm{g}}))$, and
                    $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}})$.
  3. Report the pair $(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}})$ that maximizes $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}})$.
```

Figure 5.5: Algorithm RandomRigidMotion(A, B, α, ϵ).

$O((m^2/\epsilon^4)\log(m/\epsilon)\log^2 m)$ *time. The algorithm succeeds with probability at least* $1 - 2/m^6$.

Proof: Recall that $\bar{k}_{\mathrm{opt}}$ is the number of disks A_i that participate in an optimal solution. Since $\bar{k}_{\mathrm{opt}} > k_{\mathrm{opt}}/6$, we have that $\mathbf{Pr}[(A_i \notin A_{\mathrm{opt}})] < 1 - \frac{k_{\mathrm{opt}}}{6n}$, for a random $A_i \in A$. Let R_A be the set of all disks A_i chosen in step 2. The probability that all $|R_A|$ random draws from A will fail to give a disk that belongs to an optimal pair is

$$\mathbf{Pr}[R_A \cap A_{\mathrm{opt}} = \emptyset] \leq (1 - \frac{k_{\mathrm{opt}}}{6n})^{|R_A|} \leq e^{-k_{\mathrm{opt}}|R_A|/(6n)} \leq e^{-\alpha|R_A|/6}.$$

By choosing $|R_A| \geq (36/\log e)\alpha^{-1}\log m$, we have that

$$\mathbf{Pr}[R_A \cap A_{\mathrm{opt}} = \emptyset] \leq m^{-6}.$$

If $R_A \cap A_{\mathrm{opt}} \neq \emptyset$, then at least one intersecting pair $A_i(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}), B_j$ will be identified in the first loop. Then, in the second loop, the algorithm finds a $\theta^{\mathrm{g}}_{\mathrm{apx}}$ for which, by Lemma 5.6, $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) \leq \epsilon V(A)$, for some $\vec{t}_{\mathrm{g}} \in T_{ij}$ with $|\vec{t}_{\mathrm{opt}} - \vec{t}_{\mathrm{g}}| = O(\epsilon)$, and with probability at least $1 - m^{-6}$. As in the proof of Theorem 5.3, we have that $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) = O(k_{\mathrm{opt}}\epsilon)$, $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta_{\mathrm{opt}}) \leq \mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{opt}})$ and $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}}) \leq \mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}})$. Hence,

$$\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) - \mathcal{V}(\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}) = O(k_{\mathrm{opt}}\epsilon) + \epsilon V(A).$$

Using that $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) \geq \alpha V(A)$ and $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) = \Theta(k_{\mathrm{opt}})$, the approximation bound follows. The algorithm fails to return such a pair $\vec{t}_{\mathrm{apx}}, \theta_{\mathrm{apx}}$ if and only if any of its two random sampling steps fail. That is, the algorithm fails with probability at most $2m^{-6}$.

Regarding the running time, the random sampling of set A can be easily done in $O((n/\epsilon^2)\log m)$ time. In the second step, for each of the $O(m/\epsilon^2)\log m$ grid translations $\vec{t}_{\mathrm{g}}$, the one dimensional arrangement $\Gamma_{A(\vec{t}_{\mathrm{g}})}(S)$ of $O((m/\epsilon^2)\log m)$ complexity is computed, a vertex of maximum depth $\theta^{\mathrm{g}}_{\mathrm{apx}}$ is returned and $\mathcal{V}(\vec{t}_{\mathrm{g}}, \theta^{\mathrm{g}}_{\mathrm{apx}})$ is evaluated; this takes in $O((m/\epsilon^2)\log(m/\epsilon)\log m) + m\log m)$ time. In total the running time is $O((m^2/\epsilon^4)\log(m/\epsilon)\log^2 m)$. $\square$

5.6 Sets of intersecting disks with different radii

We can generalize our results to the case where A and B consist of possibly intersecting and various size disks under the conditions of bounded scaling ratio λ and bounded depth β.

First, we show that the assumptions result in denser sampling of configuration space with constants that depend on the parameters λ and β as well. Then, we discuss their algorithmic implications.

Translations. First, consider Lemma 5.4. The maximum loss per pair is now determined by a pair of disks of radius λ each: $\mathcal{V}_{ij}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) - \mathcal{V}_{ij}(\vec{t}, \theta_{\mathrm{opt}}) < 2\lambda|\vec{t}_{\mathrm{opt}} - \vec{t}| = O(\delta)$. Therefore, $\mathcal{V}(\vec{t}_{\mathrm{opt}}, \theta_{\mathrm{opt}}) - \mathcal{V}(\vec{t}, \theta_{\mathrm{opt}}) < 2k_{\mathrm{opt}}\lambda|\vec{t}_{\mathrm{opt}} - \vec{t}| = O(k_{\mathrm{opt}}\delta)$ and the lemma holds. Moreover, Theorem 4.4 holds as well, with the constant in the Θ-notation depending on both λ and β.

Regarding the algorithm TRANSLATION, special care needs to be taken to avoid overcounting $\mathcal{V}(\vec{t}_{\mathrm{g}})$. We can do this in the following way: Consider the arrangement $\mathcal{A}$ of all disks $A_i \in A$ in the work space. Since the maximum depth in $\mathcal{A}$ is constant, $\mathcal{A}$ has $O(n)$ complexity and can be computed in $O(n\log n)$ time. Next, we compute a vertical decomposition $\mathcal{VD}(\mathcal{A})$ of $\mathcal{A}$; $\mathcal{VD}(\mathcal{A})$ has $O(n)$ disjoint cells[2] each of constant complexity and can be computed in $O(n\log n)$ time. Similarly, we compute $\mathcal{B}$ and $\mathcal{VD}(\mathcal{B})$ both in $O(m\log m)$ time. The loop in step 2 is now executed for every pair of cells $c_i \in \mathcal{VD}(\mathcal{A})$ and $c_j \in \mathcal{VD}(\mathcal{B})$ and instead of computing $\mathcal{V}_{ij}(\vec{t}_{\mathrm{g}})$, we compute $V(c_i(\vec{t}_{\mathrm{g}}) \cap c_j)$. The voting scheme proceeds as before and runs within the same time bounds.

Rotations. Consider Lemma 5.1 and its proof: the length of sampling intervals is now determined by a pair of disks of radius λ each. For such a pair A_i, B_j we have $\frac{\partial v(\theta)}{\partial\theta} \geq -2r_i\lambda$. Therefore, for any pair of disks $A_i \in A$ and $B_j \in B$, we can sample $\mathcal{V}_{ij}(\theta)$ at regular intervals whose length is at most $\delta/(2r_i\lambda)$ assuring that the loss per pair is at most 2δ. We also have to make sure that the number of samples per pair remains bounded, see Lemma 5.2. Indeed, $|R_{ij}|$ is maximized for the 'worst case' pair of disks of radius λ each; this is a scaled, by λ, version of the original problem.

---

[2] For our purpose, we only consider cells that are inside $\bigcup A$.

Regarding algorithm Rotation(A, B, δ), we use spacing of $\delta/(2r_i\lambda)$ in its first step. Unfortunately, the simple technique used in the algorithm of Figure 5.3 to approximate $\mathcal{V}(\theta)$ for all the values $\theta \in \Theta$ does not work here since the disks in each set are possibly intersecting and the area of overlap accumulated in $\tilde{\mathcal{V}}(\theta)$ can be a bad approximation of $\mathcal{V}(\theta)$. We can overcome this problem in the following way. We compute $\mathcal{VD}(\mathcal{A})$ and $\mathcal{VD}(\mathcal{B})$ as before. Observe that every cell $c_i \in \mathcal{VD}(\mathcal{A})$ is fully contained in some disk in A; similarly, every cell $c_j \in \mathcal{VD}(\mathcal{B})$ is fully contained in some disk in B. Consider the function $V(c_i(\theta) \cap c_j)$, $\theta \in [0, 2\pi)$; for every pair (c_i, c_j), the error in $V(c_i(\theta) \cap c_j)$ is bounded by the error in the pair of their corresponding disks. Since each cell in both decompositions has at most two vertical walls and at most two circular segments, the function has a bounded number of local minima/maxima. We insert all these values, for every pair of cells, in set Θ. The algorithm proceeds as before by considering whether $V(c_i(\theta) \cap c_j)$ increases, decreases or reaches an optimum at each $\theta \in \Theta$. Rotation now runs again in $O((mn/\delta)\log m)$ time and its correctness can be shown as in the proof of Lemma 5.3.

Rigid Motions. In addition to the relevant changes mentioned in the previous paragraphs, observe that a simple volume argument shows that any disk $A_i(\vec{t}, \theta)$ cannot intersect more than $9\lambda^2\beta$ disks B_j for any $\vec{t}, \theta$. Thus, $|A_{\text{opt}}| > k_{\text{opt}}/(9\lambda^2\beta)$.

In RigidMotion, we can compute $\mathcal{V}(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})$ for each pair $(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})$ in a straightforward way as follows: we compute $V(\bigcup A)$ in $O(n \log n)$ time, by computing $\mathcal{VD}(\mathcal{A})$ and summing up the areas of all its $O(n)$ cells. Similarly, we compute $V(\bigcup B)$ in $O(m \log m)$ time and, for each pair $(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})$, $V((\bigcup A(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})) \cup (\bigcup B))$ in $O(m \log m)$ time. It follows that for each pair $(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})$ we can compute $V((\bigcup A(\vec{t}_{\text{g}}, \theta^{\text{g}}_{\text{apx}})) \cap (\bigcup B))$ in $O(m \log m)$ time. By incorporating all these changes, we can prove Theorem 5.3 as before.

Regarding the extension of Theorem 5.4, we apply the same method that we use in its proof, namely computing in Rotation the disks $A_{i'}, B_{j'}$ such that $R_{i'j'}$ is not empty. Then, for each cell $c_{i'} \in A_{i'} \cap \mathcal{VD}(\mathcal{A})$ and each cell $c_{j'} \in B_{j'} \cap \mathcal{VD}(\mathcal{B})$ we proceed like before. We also need to keep track of the pairs $c_{i'}, c_{j'}$ that have been already added to avoid overcounting them. To show that the same time bound holds, we need to argue that there are asymptotically the same number of pairs $c_{i'} \in \mathcal{VD}(\mathcal{A})$ and $c_{j'} \in \mathcal{VD}(\mathcal{B})$ with $R_{i'j'} \neq \emptyset$ than we had for the case of disjoint unit disks. For this, we first observe that each disk $A_{i'}$ is decomposed into $O(1)$ cells in $\mathcal{VD}(\mathcal{A})$ because it intersects at most $9\lambda^2\beta$ other disks of A. The same holds for any $B_{j'}$. Therefore, each pair of disks $A_{i'}, B_{j'}$ that we need to consider, gives rise to $O(1)$ pairs of cells $c_{i'}, c_{j'}$.

It remains to bound the number of pairs $A_{i'}, B_{j'}$ such that $R_{i',j'} \neq \emptyset$. For this, observe that set A can be decomposed into $O(1)$ disjoint groups of disjoint disks. This can be shown using a greedy procedure: compute a maximal set of disjoint disks $\tilde{A} \subset A$, that is, any disk in A intersects some disk in $\tilde{A}$; then take $\tilde{A}$ as a disjoint group and proceed recursively with $A \setminus \tilde{A}$. After $9\lambda^2\beta + 1$ steps all the disks must be in some group, as any remaining disk must intersect a disk in each of the $9\lambda^2\beta+1$ groups, which is not possible. We can then apply Lemma 5.5 to each of the disjoint groups, and because there are a constant number of groups, we get the same asymptotic value for $\sum_i k_{j,j',i}$ as we had for the case of disjoint, unit disks. The result follows.

In RandomRigidMotion the size of R_A has to be at least $(54\lambda^4\beta/\log e)\alpha^{-1}\log m$ since the condition $\mathcal{V}(\vec{t}_{\text{opt}}, \theta_{\text{opt}}) \geq \alpha V(A)$ now gives that $k_{\text{opt}} \geq \alpha n/\lambda^2$. Note that Lemma 5.6 holds for any two planar regions A and B and thus for the two unions $\bigcup A$ and $\bigcup B$ as well. We can compute the sample points in A using $\mathcal{VD}(\mathcal{A})$. Last we compute each $W(s)$ by checking all disks in B in $O(m\log m)$ time. The running time of the algorithm stays the same and Theorem 5.5 can now be proven as before.

Chapter 6

Further research

We have examined similarity measures for weighted point sets and their use in geometric pattern matching. In addition to the immediate open problems posed at the end of each chapter, we conclude this thesis by giving a few general research directions.

In Chapter 2, we dealt with similarity measures based on weight transportation. The normalized version of the EMD, i.e., the PTD, is a straightforward variation that, of course, obeys triangle inequality, cancelling out positivity and partial matching. It seems to be quite difficult to come up with a metric for sets with different total weights. It would thus be interesting to find alternative ways of weight distribution between such point sets that result in a (pseudo) metric.

In Chapter 2, we also convinced ourselves of the potential use of the EMD and PTD in shape matching and retrieval by performing simple experiments on a variety of data sets. However, the best practical shape matching methods in computer vision, heuristics based on curvature scale space [94] and shape contexts [24] give better matching results than our simple method. One reason for that is the difficulty in deriving good point set signatures from the shapes. Therefore, one should look for more careful methods in choosing the signatures, maybe by combining the heuristics mentioned above with the use of transportation distances. A recent work towards this direction by Grauman and Darell [71], that uses also some other new ideas mentioned shortly below, has given quite promising results.

A second reason for the limitation of our matching method is the use of simple shape pose normalization that accounts for transformation effects. Heuristics normally deal with transformations by employing sophisticated shape processing that derives transformation invariant features; it is reasonable to expect that they perform better. However, these heuristics lack specific properties and do not perform partial matching. It seems that one can always come up with a counter-example and have some difficulties in analyzing the behavior of these methods, especially when they give wrong answers. As already mentioned in the introduction, to deal with this problem properly, computational geometry adopts the approach of similarity measures and their minimization under transformations. This is a theoretical framework that explores the use of appropriate similarity measures for different sets of patterns and gives deep insights to particularly hard shape matching problems. Very few of these methods have found their way into practice. Actually, no comparison has been made by performing extensive experiments on the same data sets used by the practical heuristics. The main reason is that the methods have high running times and can be very complicated to program. Regardless of their high running times, we believe that it is worthwhile doing such experiments in order to check their accuracy.

One way to decrease the running times of these theoretically sound methods is to consider approximations. Approximation algorithms are, sometimes, also much simpler and easier to implement. In Chapter 3, we have given approximation algorithms for the minimum EMD under rigid motions. These algorithms are quite simple but, in general, still quite expensive, however in this case there is no other alternative since no exact algorithms are known. In particular, the approximation algorithm for computing the EMD (for a fixed transformation) is not practical. As an alternative, for experiments, one could use the simplex algorithm for solving the transportation problem. Although the running times will not be polynomial anymore, the approximations still hold and simplex performs well in practice for reasonably sized point sets. One additional problem here is that even if the similarity measure is a metric, the approximate values do not necessarily follow the metric properties. Even simple experiments could reveal the (in)effectiveness of such approximations in shape matching. Moreover, research needs to be done in proving potential 'approximate' properties that could be useful for time-efficient retrieval. To this respect, in an approach quite similar to—and preceding— the one by Grauman and Darell, Indyk and Thaper [81] presented a method that uses a low distortion embedding of the EMD into Euclidean space and a Locality-Sensitive Hashing scheme [36] to obtain an approximate nearest neighbor search algorithm for the EMD on equal weight sets.

In Chapters 4 and 5, we presented approximation algorithms for the maximum area of overlap of two unions of disks or balls under translations and rigid motions. This is the first work dealing with matching collections of shapes. These algorithms are in general faster than the algorithms for the minimum EMD under rigid motions. However, because of the particular similarity measure used and the special assumptions on the input, they are less applicable. Any object can be efficiently approximated by a finite union of disks or balls. Our algorithms assume bounded scaling ratio and overall depth; see again Section 4.2. As Amenta noted [15], it is not clear that one can bound the error of shape approximation under these assumptions; it would be interesting to check whether such a guarantee can be given. On the other hand, the algorithms by Cheong et al. [39] for the same matching problem do not need the assumptions but give an absolute error on the maximum area of overlap. Can we design $(1-\epsilon)$- approximation algorithms that run without the assumptions in the same time bounds as the ones we give ?

Bibliography

[1] P. K. Agarwal, S. Har-Peled, M. Sharir, and Y. Wang. Hausdorff distance under translation for points, disks, and balls. In *Proc. of the 19th Annu. ACM Sympos. Comput. Geom.*, pages 282–291, 2003.

[2] P. K. Agarwal and M. Sharir. Arrangements and their applications. In J.R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 59–119. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 1999.

[3] P.K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. In *Proc. of the 11th Annu. ACM Sympos. Comput. Geom.*, pages 39–50, 1995.

[4] P.K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Comput. Surv.*, 30:412–458, 1998.

[5] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, 1993.

[6] O. Aichholzer, H. Alt, and G. Rote. Matching shapes with a reference point. *Int. J. Comp. Geom. Appl.*, 4:349–363, 1997.

[7] T. Akutsu. On determining the congruence of point sets in d-dimensions. *Comput. Geom. Theory Appl.*, 9(4):247–256, 1998.

[8] T. Akutsu, H. Tamaki, and T. Tokuyama. Distribution of distances and triangles in a point set and algorithms for computing the largest common point sets. *Discrete Comput. Geom.*, 20, 1998.

[9] H. Alt, B. Behrends, and J. Blömer. Approximate matching of polygonal shapes. *Annals of Mathematics and Artificial Intelligence*, 13:251–266, 1995.

[10] H. Alt, U. Fuchs, G. Rote, and G. Weber. Matching convex shapes with respect to the symmetric difference. *Algorithmica*, 21:89–103, 1998.

[11] H. Alt and L.J. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation. In J.R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 121–153. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 1999.

[12] H. Alt, C. Knauer, and C. Wenk. Matching polygonal curves with respect to the Fréchet distance. In *Proc. of the 18th Int. Sympos. Theoretical Aspects of Computer Science*, volume 2010 of *LNCS*, pages 63–74, 2001.

[13] H. Alt, K. Mehlhorn, H. Wagener, and E. Welzl. Congruence, similarity, and symmetries of geometric objects. *Discrete Comput. Geom.*, 3, 1988.

[14] C. Ambühl, S. Chakraborty, and B. Gärtner. Computing largest common point sets under approximate congruence. In *Proc. of the 8th European Sympos. Algorithms*, volume 1879 of *LNCS*, pages 52–63, 2000.

[15] N. Amenta. Personal communication, 2004.

[16] N. Amenta and R. Kolluri. Accurate and efficient unions of balls. In *Proc. of the 16th Annu. ACM Sympos. Computat. Geom.*, pages 119–128, 2000.

[17] E.M. Arkin, K. Kedem, J.S.B. Mitchell, J. Sprinzak, and M. Werman. Matching points into noise regions: Combinatorial bounds and algorithms. In *Proc. of the 2nd Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 42–51, 1991.

[18] S. Arora. Approximation schemes for NP-hard geometric optimization problems: A survey. `http://www.cs.princeton.edu/~arora/pubs/arorageo.ps`, 2002.

[19] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. In *Proc. of the 5th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 573–582, 1994.

[20] D.S. Atkinson and P.M. Vaidya. Using geometry to solve the transportation problem in the plane. *Algorithmica*, 13:442–461, 1995.

[21] C. Bajaj. The algebraic degree of geometric optimization problems. *Discrete Comput. Geom.*, 3:177–191, 1988.

[22] D.H. Ballard. Generalized Hough transform to detect arbitrary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):111–122, 1981.

[23] J. Barros, J. French, W. Martin, P. Kelly, and M. Cannon. Using the triangle inequality to reduce the number of comparisons required for similarity-based retrieval. In *Proc. of SPIE, vol. 2670, Storage and Retrieval for Still Image and Video Databases IV*, pages 392–403, 1996.

[24] S. Belongie, J. Malic, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, 2002.

[25] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In D. Hochbaum, editor, *Approximation algorithms for NP-hard problems.* PWS Publishing, 1996.

[26] P.J. Besl and N.D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[27] P. Bose, A. Maheshwari, and P. Morin. Fast approximations for sums of distances, clustering and the Fermat-Weber problem. *Comput. Geom. Theory Appl.*, 24:135–146, 2003.

[28] L. Boxer. On Hausdorff-like metrics for fuzzy sets. *Pattern Recognition Letters*, 18, 1997.

[29] P. Braß. Exact point pattern matching and the number of congruent triangles in a three-dimensional point set. In *Proc. of the 8th European Sympos. Algorithms*, volume 1879 of *LNCS*, pages 112–119, 2000.

[30] P. Braß. On the non-existence of Hausdorff-like metrics for fuzzy sets. *Pattern Recognition Letters*, 23(1–3):39–43, 2002.

[31] P. Braß and C. Knauer. Testing the congruence of d-dimensional point sets. *Int. J. of Comp. Geom. Appl.*, 12(1):115–124, 2002.

[32] S. Cabello, P. Giannopoulos, C. Knauer, and G. Rote. Matching point sets with respect to the Earth Mover's Distance. To appear in *Proc. of the 13th Annu. European Sympos. Algorithms (ESA)*, LNCS, October 2005.

[33] P.B. Callahan and S. Rao Kosaraju. Faster algorithms for some geometric graph problems in higher dimensions. In *Proc. of the 4th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 291–300, 1993.

[34] T.M. Chan. Geometric applications of a randomized optimization technique. *Discrete Comput. Geom.*, 22:547–567, 1999.

[35] R. Chandrasekaran and A. Tamir. Algebraic optimization: The Fermat-Weber location problem. *Math. Programming*, 46(2):219–224, 1990.

[36] M.S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. of the 34th Annu. ACM Sympos. Theory of Computing*, pages 380–388, 2002.

[37] B.B. Chaudhuri and A. Rosenfeld. On a metric distance between fuzzy sets. *Pattern Recognition Letters*, 17, 1996.

[38] C.C. Chen. Improved moment invariants for shape discrimination. *Pattern Recognition*, 26(5):683–686, 1993.

[39] O. Cheong, A. Efrat, and S. Har-Peled. On finding a guard that sees most and a shop that sells most. 2004. Manuscript. Submitted to Discrete Comput. Geom.

[40] L.P. Chew, D. Dor, A. Efrat, and K. Kedem. Geometric pattern matching in d-dimensional space. *Discrete Comput. Geom.*, 21, 1999.

[41] L.P. Chew, M. Goodrich, D. P. Huttenlocher, K. Kedem, J. M. Kleinberg, and D. Kravets. Geometric pattern matching under Euclidean motion. *Comput. Geom. Theory Appl.*, 7:113–124, 1997.

[42] L.P. Chew and K. Kedem. Getting around a lower bound for the minimum Hausdorff distance. *Comput. Geom. Theory Appl.*, 10:197–202, 1998.

[43] L.P. Chew, K. Kedem, and S. Schirra. On characteristic points and approximate decision algorithms for the minimum Hausdorff distance. Research Report MPI-I-94-150, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1994.

[44] S.D. Cohen. *Finding Color and Shape Patterns in Images*. PhD thesis, Stanford University, Department of Computer Science, 1999.

[45] S.D. Cohen and L.J. Guibas. The Earth Mover's Distance under transformation sets. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, pages 173–187, September 1999.

[46] R. Cole. Slowing down sorting networks to obtain faster sorting algorithms. *J. ACM*, 31:200–208, 1984.

[47] D. A. Cox, J. B. Little, and D. O'Shea. *Using Algebraic Geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998.

[48] K. Kedem D. P. Huttenlocher and M. Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete Comput. Geom.*, 9:267–291, 1993.

[49] R. H. Davis and J. Lyall. Recognition of handwritten characters—a review. *Image and Vision Computing*, 4:208–218, 1986.

[50] M. de Berg, S. Cabello, P. Giannopoulos, C. Knauer, R. van Oostrum, and R. C. Veltkamp. Maximizing the area of overlap of two unions of disks under rigid motion. In *Proc. of the 9th Scand. Work. Algorithm Theory*, volume 3111 of *LNCS*, pages 138–149, 2004.

[51] M. de Berg, O. Devillers, M. van Kreveld, O. Schwarzkopf, and M. Teillaud. Computing the maximum overlap of two convex polygons under translations. In *Proc. of the 7th Internat. Symp. Algorithms and Computation (ISAAC '96)*, volume 1178 of *LNCS*, pages 126–135, 1996.

[52] M. de Berg, P. Giannopoulos, C. Knauer, R. van Oostrum, and R. C. Veltkamp. The area of overlap of two unions of convex objects under translations. Technical Report UU-CS-2003-025, Institute of Information and Computing Sciences, Utrecht University, The Netherlands, 2003.

[53] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2nd edition edition, 2000.

[54] P.J. de Rezende and D.T. Lee. Point set matching in d-dimensions. *Algorithmica*, 13:387–404, 1995.

[55] A. Diaz, I. Emiris, E. Kaltofen, and V. Pan. Algebraic algorithms. In M. J. Atallah, editor, *Algorithms & Theory of Computation Handbook*, chapter 16, pages 16.1–16.27. CRC Press, 1999.

[56] D. Doerman. The UMD logo database. `http://documents.cfar.umd.edu/resources/database/UMDlogo.html`.

[57] A. Efrat, A. Itai, and M.J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31:1–28, 2001.

[58] A. Efrat and M. Sharir. The complexity of the union of fat objects in the plane. *Discrete Comput. Geom.*, 23:171–189, 2000.

[59] J.-L. Fan. Note on Hausdorf-like metrics for fuzzy sets. *Pattern Recognition Letters*, 19, 1998.

[60] P.W. Finn and L.E. Kavraki. Computational approaches to drug design. *Algorithmica*, 25:347–371, 1999.

[61] G.N. Fredrickson and D.B. Johnson. Finding kth paths and p-centers by generating and searching good data structures. *J. Algorithms*, 4:61–80, 1983.

[62] D. S. Fry. *Shape Recognition using Metric on the Space of Shapes*. PhD thesis, Harvard University, Division of Applied Sciences, 1993.

[63] H.N. Gablow and R.E. Tarjan. Faster scaling algorithms for network problems. *SIAM J. Comput*, 18(5):1013–1036, 1989.

[64] M. Gavrilov, P. Indyk, R. Motwani, and S. Venkatasubramanian. Combinatorial and experimental methods for approximate point pattern matching. *Algorithmica*, 38(2):59–90, 2004.

[65] P. Giannopoulos and R. C. Veltkamp. A pseudo-metric for weighted point sets. In *Proc. of the 7th European Conf. Computer Vision*, volume 2352 of *LNCS*, pages 715–731, 2002.

[66] G.J. Giezeman and P. Giannopoulos. The PTD retrieval demo. `http://give-lab.cs.uu.nl/matching/ptd`, 2002.

[67] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:542–551, 1991.

[68] S. Gold. *Matching and learning structural and spatial representations with neural networks*. PhD thesis, Yale University, 1995.

[69] A. Goldberg and R. Tarjan. Solving minimum-cost flow problems by successive approximation. In *Proc. of the 19th Annu. ACM Sympos. Theory of Computing*, pages 7–18, 1987.

[70] J. E. Goodman and R. Pollack. Multidimensional sorting. *SIAM J. Comput.*, 12:484–507, 1983.

[71] K. Grauman and T. Darell. Fast contour matching using approximate earth mover's distance. In *Proc. of the IEEE Conf. Comp. Vision and Pattern Recognition*, pages 220–227, 2004.

[72] M. Hagedoorn and R.C. Veltkamp. State-of-the-art in shape matching. Technical Report UU-CS-1999-027, Institute of Information and Computing Sciences, Utrecht University, The Netherlands, 1999.

[73] M. Hagerdoorn. *Pattern matching using similarity measures*. PhD thesis, Universiteit Utrecht, Institute of Information and Computer Science, 2000.

[74] D. Halperin and M. H. Overmars. Spheres, molecules, and hidden surface removal. *Comput. Geom. Theory Appl.*, 11(2):83–102, 1998.

[75] P.J. Heffernan and S. Schirra. Approximate decision algorithms for point set congruence. *Comput. Geom. Theory Appl.*, 4(3):137–156, 1994.

[76] S. Hillier and G.J. Lieberman. *Introduction to Mathematical Programming*. McGraw-Hill, 1990.

[77] J.B. Howard. Strategies for sorting melodic incipits. In *Computing in Musicology, Melodic Similarities: Concepts, Procedures and Applications*, volume 11, pages 119–128. MIT Press, Cambridge, Massachusetts, 1998.

[78] J.S. Huang and M.L. Chung. Separating similar complex Chinese characters by walsh transform. *Pattern Recognition*, 20:425–428, 1987.

[79] D. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proc. of the 1st Int. Conf. Computer Vision*, pages 102–111, 1987.

[80] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[81] P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *3rd Int. Workshop on Statistical and Computational Theories of Vision*, 2003.

[82] P. Indyk and S. Venkatasubramanian. Approximate congruence in nearly linear time. *Comput. Geom. Theory Appl.*, 24:115–128, 2003.

[83] C. Jacobs, A. Finkelstein, and D. Salenin. Fast multiresolution image querying. In *Computer Graphics SIGGRAPH*, pages 277–286, 1995.

[84] R. Jain, R. Kasturi, and B. G. Schunck. *Machine Vision*. McGraw-Hill, 1995.

[85] J.Pach K. Kedem, R. Livne and M. Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Comput. Geom.*, 1:59–71, 1986.

[86] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, 1990.

[87] O. Klein and R.C. Veltkamp. Approximation algorithms for the Earth Mover's Distance under transformations using reference points. In *Proc. of the 21st European Work. Comput. Geom.*, pages 53–56, 2004.

[88] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, 1998.

[89] P. Giannopoulos M. Farshi and J. Gudmundsson. Finding the best shortcut in a geometric network. In *Proc. of the 21st Annu. ACM Sympos. Comput. Geom.*, page to appear, Pisa, Italy, June 2005.

[90] J.B.A. Maintz. *Retrospective registration of tomographic brain images.* PhD thesis, Universiteit Utrecht, 1996.

[91] J. Matoušek. *Lectures on Discrete Geometry*, volume 212 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2002.

[92] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983.

[93] F. Mokhtarian and S. Abbasi. The SQUID system and database. `http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html`, 1996.

[94] F. Mokhtarian and S. Abbasi. Shape similarity retrieval under affine transforms. *Pattern Recognition*, 35:31–41, 2002.

[95] G. Monge. Mémoire sur la théorie des déblais et de remblais, 1781.

[96] D. M. Mount, R. Silverman, and A. Y. Wu. On the area of overlap of translated polygons. *Computer Vision and Image Understanding*, 64:53–61, 1996.

[97] D. Mumford. Mathematical theories of shape: Do they model perception. In *Proc. of SPIE, vol. 1570, Geometric Methods in Computer Vision*, pages 2–10, 1991.

[98] M. Novotni and R. Klein. A geometric approach to 3-D object comparison. In *Proc. of the Int. Conf. Shape Modelling and Appl.*, pages 167–175, 2001.

[99] J.B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research*, 41(2):338–350, 1993.

[100] J. O'Rourke and N. Badler. Decomposition of three-dimensional objects into spheres. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1(3):295–305, July 1979.

[101] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3-D models with shape distributions. In *Proc of the Int. Conf. on Shape Modeling and Appl.*, pages 154–166, 2001.

[102] M. Sharir P. K. Agarwal and S. Toledo. Applications of parametric searching in geometric optimization. *J. Algorithms*, 17:292–318, 1994.

[103] M. Petrou and P. Bosdogianni. *Image Processing: The Fundamentals.* John Wiley, 1999.

[104] Plato. Meno, 380 bc. In W. Lamb, editor, *Plato in Twelve Volumes*, volume 3. Harvard University Press, 1967.

[105] N.B. Preparata and M.I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, 2nd edition, 1990.

[106] S. T. Rachev and L. Rüschendorf. *Mass Transportation Problems*, volume I: Theory. Springer, 1998.

[107] S. T. Rachev and L. Rüschendorf. *Mass Transportation Problems*, volume II: Applications. Springer, 1998.

[108] V. Ranjan and A. Fournier. Matching and interpolation of shapes using unions of circles. *Computer Graphics Forum*, 15(3):129–142, August 1996.

[109] A. Rosenfeld and J.S. Weszka. An improved method of angle detection on digital curves. *IEEE Transactions on Computers*, 24(9):940–941, 1975.

[110] G. Rote. A new metric between polygons, and how to compute it. In *Proc. of the 19th Int. Colloquium Automata, Languages and Programming*, volume 623 of *LNCS*, pages 404–415, 1992.

[111] Y. Rubner. Linear programming code for computing the Earth Mover's Distence. `http://vision.stanford.edu/~rubner/emd/default.htm`, 1998.

[112] Y. Rubner. *Perceptual Metrics for Image Database Navigation.* PhD thesis, Stanford University, Department of Computer Science, 1999.

[113] Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image retrieval. In *Proc. of the 6th Int. Conf. Computer Vision*, pages 59–66, 1998.

[114] Y. Rubner, C. Tomasi, and L.J. Guibas. The Earth Mover's Distance as a metric for image retrieval. *Int. J. of Computer Vision*, 40(2):99–121, 2000.

[115] W.J. Rucklidge. Lower bounds for the complexity of the Hausdorff distance. In *Proc. of the 5th Canad. Conf. Comput. Geom.*, pages 145–150. University of Waterloo, August 1993.

[116] E.J. Russell. Extension of Dantzig's algorithm to finding an initial near-optimal basis for the transportation problem. *Operations Research*, 17:187–191, 1969.

[117] R. Schneider. *Convex Bodies - the Brunn-Minkowski Theory*, volume 44 of *Encyclopedia of Mathematics and its Applications.* Cambridge University Press, Cambridge, 1993.

[118] S. Sclaroff and A.P. Pentland. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6), June 1995.

[119] M. Sharir. On k-sets in arrangements of curves and surfaces. *Discrete Comput. Geom.*, 6:593–613, 1991.

[120] C.G. Small. *The statistical theory of shapes.* Springer Series in Statistics. Springer-Verlag, 1996.

[121] S. M. Smith. SUSAN low level image processing. `http://www.fmrib.ox.ac.uk/~steve/susan/index.html`, 1997.

[122] S.M. Smith and J. M. Brady. SUSAN - a new approach to low level image processing. *Int. J. of Computer Vision*, 23(1):45–78, May 1997.

[123] M. Tanase and R.C. Veltkamp. A survey of content-based image retrieval systems. In O. Marques and B. Furht, editors, *Content-based image and video retrieval*, pages 47–101. Kluwer Academic Publishers, 2002.

[124] H. Tangelder. 3-D mathcing on curvature demo. `http://give-lab.cs.uu.nl/matching/3dmatch.html`, 2002.

[125] J.W.H. Tangelder and R.C. Veltkamp. Polyhedral model retrieval using weighted point sets. *Int. J. of Image and Graphics*, 3(1):209–229, 2003.

[126] C.-H. Teh and R.T. Chin. On the detection of dominant points on digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):859– 872, 1989.

[127] G. Fejes Tóth. Packing and covering. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 2, pages 19–42. CRC Press, 1997.

[128] R. Typke. The Orpheus retrieval system. `http://give-lab.cs.uu.nl/MIR/orpheus.html`, 2004.

[129] R. Typke, P. Giannopoulos, R.C. Veltkamp, F. Wiering, and R. van Oostrum. Using transportation distances for measuring melodic similarity. In *Proc. of the 4th Int. Conf. Music Information Retrieval*, pages 107–114, 2003.

[130] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[131] F. van der Stappen. *Motion Planning amidst Fat Obstacles*. PhD thesis, Universiteit Utrecht, Institute of Information and Computer Science, 1994.

[132] R. van Oostrum and R.C. Veltkamp. Parametric search made practical. *Comput. Geom. Theory Appl.*, 28:75–88, 2002.

[133] K.R. Varadarajan and P.K. Agarwal. Approximation algorithms for bipartite and non-bipartite matching in the plane. In *Proc. of the 10th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 805–814, 1999.

[134] V.V. Vazirani. *Matrix Computations*. The Johns Hopkins University Press, 1989.

[135] V.V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.

[136] R.C. Veltkamp. Shape matching: similarity measures and algorithms. In *Proc. of the Int. Conf. on Shape Modeling and Appl.*, pages 188–197, 2001.

[137] R.C. Veltkamp. Shape algorithmics. *Algorithmica*, 38:1–4, 2004.

[138] J. Vleugels. *On Fatness and Fitness — Realistic Input Models for Geometric Algorithms*. PhD thesis, Universiteit Utrecht, Institute of Information and Computer Science, 1994.

[139] J. Vleugels and R. C. Veltkamp. Efficient image retrieval through vantage objects. *Pattern Recognition*, 35(1):69–80, 2002.

[140] E.V. Weiszfeld. Sur le point par leqeul la somme des distances de n points donnés est minimum, 1937.

[141] H. Wolfson and I. Rigoutsos. Geometric hashing: an overview. *IEEE Computational Science & Engineering*, pages 10–21, October-December 1997.

[142] M. Worring and A.W.M. Smeulders. Digital curvature estimation. *Computer Vision, Graphics, and Image Processing. Image Understanding*, 58(3):366–382, 1993.

[143] A. Zisserman and G.A.W. West. Canny edge detection software. `http://www.computing.edu.au/~geoff/ftp.html`, 1992.

Samenvatting

Geometrische patroonvergelijking is een fundamenteel vraagstuk in *computer vision*, patroonherkenning en robotica: men wil van twee gegeven patronen weten in hoeverre ze op elkaar lijken wanneer een bepaalde afstandsmaat tussen de patronen in beschouwing wordt genomen. Dit proefschrift behandelt het vergelijken van verzamelingen van gewogen punten, waarbij de punten in een metrische ruimte liggen, en waarbij het gewicht een extra attribuut vertegenwoordigt. Zulke verzamelingen dringen zich op natuurlijke wijze op in vele toepassingen, van kleurgebaseerd zoeken naar afbeeldingen tot het vergelijken van muziekpartituren. De keuze van de afstandsmaat hangt telkens af van de gewenste eigenschappen ervan, die gesteld worden door de toepassing.

De *Earth Mover's Distance* (EMD) is een gelijkheidsmaat voor verzamelingen van gewogen punten die gebaseerd is op het verplaatsen van gewicht. De EMD meet de minimum inspanning die nodig is om te ene verzameling in de andere om te zetten. Inherent aan deze afstandsmaat is dat de vergelijking niet noodzakelijk alle punten in ogenschouw neemt, met als gevolg dat deze afstandsmaat niet aan de driehoeksongelijkheid voldoet. De driehoeksongelijkheid is een eigenschap die nuttig is voor het snel zoeken in grote databases. In het tweede hoofdstuk presenteren we een variant op de EMD, namelijk, de *Proportional Transportation Distance* (PTD), die een pseudo-metriek is. Daarnaast tonen we de effectiviteit van de EMD en de PTD in het vergelijken van en zoeken naar patronen door beide te testen op verschillende gegevensverzamelingen, zoals bedrijfslogo's en muziekpartituren. Er wordt aangetoond dat de PTD een duidelijk voordeel heeft ten opzichte van de EMD voor wat betreft efficient zoeken.

Voor het effectief vergelijken van vormen zou men de overeenkomst tussen vormen bij voorkeur onafhankelijk van transformaties willen meten. Dat wil zeggen: voor gegeven vormen A en B, een afstandsmaat en een transformatiegroep, wil men een getransformeerde versie van A bepalen die de afstand tot B minimaliseert. Dit is een geometrisch optimaliseringsprobleem dat behandeld wordt in de rest van dit proefschrift. We concentreren ons daarbij op translaties en op combinaties van translaties en rotaties. De kostenfuncties van de optimaliseringsproblemen die we beschouwen blijken nogal gecompliceerd te zijn. We richten onze aandacht daarom op het benaderen van de optimale oplossing.

In het derde hoofdstuk behandelen we het vraagstuk van het minimaliseren van de EMD onder transformaties. Met behulp van eenvoudige ondergrenzen voor de kostenfunctie geven we de eerste $(1+\epsilon)$- en $(2+\epsilon)$-benadering voor de minimum EMD onder translaties en translaties/rotaties. Als een speciaal geval bekijken we nog het (partiële) toewijzingsprobleem onder dezelfde tranformaties.

In de laatste twee hoofdstukken is het gewicht van een punt afhankelijk van de oppervlakte van een convex object waarin het punt ligt. Een verzameling van gewogen punten definieert dan een vorm die gegeven is door de vereniging van de objecten. De mate van overeenkomst van twee van dergelijke verzamelingen wordt bepaald door de oppervlakte van de overlap van beide verenigingen. Eerst laten we zien dat de complexiteit van de transformatieruimte, uitgedrukt in termen van combinatorisch verschillende plaatsingen, nogal groot is, zowel voor translaties als voor combinaties van translaties en rotaties. Vervolgens geven we deterministische en probabilistische $(1-\epsilon)$-benaderingsalgoritmen voor het maximaliseren van de overlap tussen twee verenigingen van cirkelschijven onden translaties en combinaties van translaties en rotaties.

Curriculum Vitae

Panagiotis Giannopoulos

1 March 1973	Born in Athens, Greece
1988 - 1990	1st Lyceum of Kifisia, Athens
1991 - 1997	Diploma in Computer Engineering and Informatics, Department of Computer Engineering and Informatics, University of Patras, Greece
1997 - 1998	M.Sc. in Advanced Computing, Imperial College, University of London, UK
1998 - 1999	Software Engineer, Orchestream Ltd., London, UK
2000 - 2004	PhD student (AIO), Institute of Information and Computing Sciences, Utrecht, The Netherlands
2005 -	Guest, Institut für Informatik, Freie Universität Berlin, Germany