The directed Hausdorff distance between imprecise point sets

Christian Knauer[1], Maarten Löffler[2], Marc Scherfenberg[1], and Thomas Wolle[3]

[1] Institute of Computer Science, Freie Universität Berlin, Germany.
scherfen@mi.fu-berlin.de
[2] Dep. of Information and Computing Sciences, Utrecht University, the Netherlands.
loffler@cs.uu.nl
[3] NICTA Sydney, Australia. thomas.wolle@nicta.com.au

Abstract. We consider the directed Hausdorff distance between point sets in the plane, where one or both point sets consist of imprecise points. An imprecise point is modelled by a disc given by its centre and a radius. The actual position of an imprecise point may be anywhere within its disc. Due to the direction of the Hausdorff Distance and whether its tight upper or lower bound is computed there are several cases to consider. For every case we either show that the computation is NP-hard or we present an algorithm with a polynomial running time. Further we give several approximation algorithms for the hard cases and show that one of them cannot be approximated better than with factor 3, unless P=NP.

1 Introduction

The analysis and comparison of geometric shapes are essential tasks in various application areas within computer science, such as pattern recognition and computer vision. Beyond these fields also other disciplines evaluate the shape of objects such as cartography, molecular biology, medicine, or biometric signal processing. In many cases patterns and shapes are modeled as finite sets of points.

The *Hausdorff* distance is an important tool to measure the similarity between two sets of points (or, more generally, any two subsets of a metric space). It is defined as the largest distance from any point in one of the sets, to the closest point in the other set (see Section 1.3 for a formal definition). This distance is used extensively in pattern matching.

Data imprecision is a phenomenon that has existed as long as data is being collected. In practice, data is often sensed from the real world, and as a result has a certain error region. On the one hand, many application fields of computational geometry use algorithms that take this into account. However, these algorithms are mostly heuristics, and do not benefit from theoretical guarantees. On the other hand, algorithms from computational geometry are provably correct and efficient, often under the assumption that the input data is correct. If we want these algorithms to be used in practice, they need to take imprecision into account in the analysis. Thus not surprisingly, data imprecision in computational geometry is receiving more and more attention.

In this paper, we study several variants of the important and elementary problem of computing the Hausdorff distance under the Euclidean metric between *imprecise* point sets.

1.1 Related Work

The Hausdorff distance is one of the most studied similarity measures. For a survey about similarity measures and shape matching refer to [2]. A straightforward, naive algorithm computes the Hausdorff distance between two point sets A and B consisting of m and n points, respectively, in $O(mn)$ time. Using Voronoi diagrams and a more sophisticated approach the running time can be reduced to $O((m+n)\log n)$, [1].

The study of imprecision within computational geometry started around twenty years ago, when Guibas *et al.* [7] introduced *epsilon geometry* as a way to handle computational imprecision. In this model, each point is assumed to be at most ε away from its given location.

For a given measure on a set of imprecise points, one of the simplest questions to ask in this model is what are the possible output values? Each input point can be anywhere in a given region, and depending on where each point is, the output will have a different value. This leads to the problem of placing the points in their regions such that this value is minimised or maximised. One of the first results of this kind is due to Goodrich and Snoeyink [6], who show how to place a set of points on a set of vertical line segments such that the points are in convex position and the area or perimeter of the convex hull is minimised in $O(n^2)$ time. A similar problem is studied by Mukhopadhyay *et al.* [12], and their result was later generalised to isothetic line segments [11].

Nagai and Tokura [13] thoroughly study the efficient computation of lower and upper bounds for a variety of region shapes and measures; in particular they study the diameter, the width, and the convex hull, and all their algorithms run in $O(n\log n)$ time. However, not all of their bounds are tight. Van Kreveld and Löffler [14] study the same problems and give algorithms to compute tight bounds, though the running times of the algorithms can be much higher and some variants are proven to be NP-hard.

Related work concerning the Hausdorff distance in an imprecise context includes [5] and [8].

1.2 Contribution

In this paper, we assume that an imprecise point is modelled by a disc with a given centre and radius. In general, it is possible that the discs intersect. We assume we have two sets of points, P and Q, and that at least one of them is imprecise. We want to compute the directed Hausdorff distance from P to Q. This includes both the tight lower and upper bound on the possible values, for each combination. This leads to six different cases. Additionally, in some settings the problems become easier if we restrict the model of imprecision to disjoint

setting	tight lower bound	tight upper bound
$h(\tilde{P}, Q)$ [general]	$O(n^2)$	$O(n \log n)$
$h(P, \tilde{Q})$ [general]	NP-hard*, 4-APX in $O(n^3 \log^2 n)$	$O(n \log n)$
[disjoint unit discs]	3-APX-hard, 3-APX in $O(n^{10} \log n)$	$O(n \log n)$
$h(\tilde{P}, \tilde{Q})$ [general]	NP-hard	$O(n^2)$
[const. depth in $\tilde{P}$]		$O(n \log n)$

Table 1. P and Q are point sets and $\tilde{P}$ and $\tilde{Q}$ are imprecise point sets. Results are shown for the case when all sets have $O(n)$ elements. *can be computed exactly in $O(n^3)$ if the discs are disjoint and the answer is smaller than $r(\sqrt{5-2\sqrt{3}}-1)/2$ where r is the radius of the smallest disc in $\tilde{Q}$.

discs or discs that all have the same radius; we state these results separately. Our results are summarised in Table 1.

In the next section, we review some definitions and structures that we use to obtain our results. After that, we present our three main results. In Section 2, we give a general algorithm for computing the upper bound, which works in all settings in the table, though it can be simplified (conceptually) in some settings. In Section 3, we prove hardness of computing the lower bound in most settings. Finally, in Section 4, we give algorithmic results for computing the lower bound, exactly in some cases and approximately in others. Due to severe space constraints, we are not able to include most of the details of the algorithms. In what follows, we describe the main ideas and structure of our results, but we encourage the interested reader to consult the full version of this paper instead, which can be found in [9].

1.3 Preliminaries

The directed Hausdorff distance h from a point set $P = \{p_1, \ldots, p_m\}$ to a point set $Q = \{q_1, \ldots, q_n\}$ with an underlying Euclidean metric can be computed in $O((n+m)\log n)$ time, see [1], and is defined as (see Fig. 1 for an example):

$$h(P, Q) = \max_{p \in P} \min_{q \in Q} \|p - q\|$$

Let $\tilde{P}$ and $\tilde{Q}$ denote two imprecise point sets consisting of m and n closed discs respectively. We call a set $P = \{p_1, \ldots, p_m\}$ a *precise realisation* of $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_m\}$ if $p_i \in \tilde{p}_i$ for all i. We also write $P \Subset \tilde{P}$ in this case.

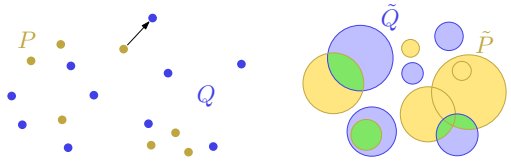

Fig. 1. (a) $h(P, Q)$ is defined by the pair of points indicated by the arrow. (b) An example input of imprecise points.

We define the directed Hausdorff distance between a precise and an imprecise or two imprecise point sets as the interval of all possible outcomes for that distance.

$$h(P, \tilde{Q}) = \{h(P, Q) \mid Q \Subset \tilde{Q}\}, \quad h(\tilde{P}, Q) = \{h(P, Q) \mid P \Subset \tilde{P}\}$$
$$h(\tilde{P}, \tilde{Q}) = \{h(P, Q) \mid P \Subset \tilde{P}, Q \Subset \tilde{Q}\}$$

Further, we denote the tight upper and lower bounds of this interval by $h_{\max}$ and $h_{\min}$ respectively, for example

$$h_{\max}(P, \tilde{Q}) = \max h(P, \tilde{Q}) \quad \text{and hence} \quad h(P, \tilde{Q}) = [h_{\min}(P, \tilde{Q}), h_{\max}(P, \tilde{Q})].$$

2 Algorithm for computing the tight upper bound

In this section, we consider the following problem. Given are two set of discs $\tilde{P}$ and $\tilde{Q}$. The radii may be all different; an example input is shown in Fig. 1(b). We want to place point sets $P \Subset \tilde{P}$ and $Q \Subset \tilde{Q}$ such as to maximise the directed Hausdorff distance $h(P, Q)$. In other words, we want to place the points in P and Q such that one point from P is as far as possible away from all points in Q. The placements of the remaining points of P do not matter. So, we need to identify which point $\hat{p} \in P$ will play this important role. We need to place $\hat{p}$ such that after we placed all points in Q as far away from $\hat{p}$ as possible, this distance is maximised.

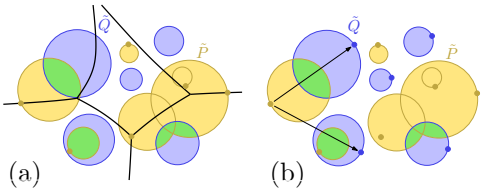

Fig. 2. (a) The inverted additive Voronoi Diagram (iaVD) of $\tilde{Q}$. The point set P placed locally optimal. (b) The points in Q are all placed as far away from $\hat{p}$ as possible.

2.1 Basic algorithm

We will first compute the *inverted additive Voronoi Diagram* (iaVD) of $\tilde{Q}$. This is a subdivision of the plane into regions where each point x in the plane is associated with the disc in $\tilde{Q}$ whose furthest point is closest to x. See Fig. 2(a) for an example. This diagram can be computed in $O(n \log n)$ time [4], since it corresponds to the additively weighted Voronoi Diagram (also known as Apollonius diagram) of the centres of $\tilde{Q}$, where the weight of a point is minus the radius of the corresponding disc.
Using the iaVD, we can place each point $p \in \tilde{p} \in \tilde{P}$ at a locally optimal position, as if it were $\hat{p}$. We identify three possible placement types for p that are locally optimal, as is illustrated in Fig. 2.

1. A vertex of the iaVD.
2. An intersection point between a Voronoi edge and a disc from $\tilde{P}$.
3. A point on the boundary of $\tilde{p}$ that is furthest away from the iaVD site whose cell contains the centre of $\tilde{p}$

We can now iterate over all points in P and their locally optimal placements, and determine $\hat{p}$ by keeping track of the locally optimal placement $p \in \tilde{p}$ such that the shortest distance between p and (the furthest point on) any disc in $\tilde{Q}$ is maximised. Once $\hat{p}$ is known, we place all points in Q as far away from $\hat{p}$ as

possible, and all points in $P \setminus \{\hat{p}\}$ anywhere inside their discs. The result is shown in Fig. 4(b). As it is possible that there are $O(mn)$ locally optimal placements of the second type (namely: an intersection between a disc boundary and a Voronoi edge), we conclude with the following theorem.

Theorem 1. *Given two sets $\tilde{P}$ and $\tilde{Q}$ of imprecise points of size m and n, respectively, we can compute $h_{\max}(\tilde{P}, \tilde{Q})$ and precise realisations $P \Subset \tilde{P}$ and $Q \Subset \tilde{Q}$ with $h(P, Q) = h_{\max}(\tilde{P}, \tilde{Q})$ in $O(nm + n \log n)$ time.*

2.2 Faster algorithms in special cases

In this section we show how the above result can be improved under certain assumptions. To speed up the algorithm, we make some observations about the nature of locally optimal placements.

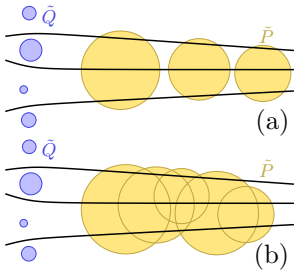

Fig. 3. (a) There could be a quadratic number of intersections between the edges of the iaVD of $\tilde{Q}$ and the discs in $\tilde{P}$. (b) When the discs overlap, the *union* of $\tilde{P}$ has fewer intersections with the iaVD.

Lemma 1. *Let $\tilde{p}$ be a disc in $\tilde{P}$, and let $\tilde{q}_1$ and $\tilde{q}_2$ be two discs in $\tilde{Q}$, such that the part of the bisector of $\tilde{q}_1$ and $\tilde{q}_2$ that is in the iaVD slices through $\tilde{p}$ (that is, it is not connected to a vertex of the iaVD inside $\tilde{p}$). Then the optimal placement of p occurs on the same side of this bisector where the centre of $\tilde{p}$ is, regardless of the rest of the iaVD.*

Proof. Some notation: let p_c be the centre of $\tilde{p}$, q_{c1} the centre of $\tilde{q}_1$ and q_{c2} the centre of $\tilde{q}_2$. Now let f_1 be the point on the boundary of $\tilde{p}$ that is furthest away from q_{c1} (this would be the type 3 placement if $\tilde{q}_1$ was the only player), and similarly let f_2 be the point furthest away from q_{c2}. Now, suppose w.l.o.g. that p_c is on the same side as q_{c1}. Now, suppose that the optimal placement p is on the other side, that is, on the side of q_{c2}. Then we observe that f_2 must be on the side of q_{c1}, because q_{c2}, p_c and f_2 lie on a line. This means that along the boundary of $\tilde{p}$, the intersection points with the bisector have a better value than any other point on the side of q_{c2}, in particular, better than p, which is a contradiction. (Note that if there are other cells of the iaVD involved, the value of p could only be lower). □

This lemma basically says that if we want to place a certain point p locally optimally, we can start looking by walking from the centre of $\tilde{p}$ and never have to cross edges of the iaVD that slice through $\tilde{p}$, like illustrated in Fig. 3. This makes us arrive at the following conclusion.

Corollary 1. *Let $\tilde{p}$ be a disc in $\tilde{P}$, and suppose that the iaVD has t vertices inside $\tilde{p}$. Then we can find the locally optimal placement for p in $O(t)$ time.*

This immediately implies that if the discs of $\tilde{P}$ do not overlap, we can simply place all points p independently in linear time.

Now, assume that the discs of $\tilde{P}$ are disjoint, or that the intersection depth is at most some constant c. Then, clearly, each vertex of the iaVD can appear in at most c discs of $\tilde{P}$. So, if each disc $\tilde{p}_i$ contains t_i vertices of the iaVD, we have $\sum_i t_i \leq cn$, and we can find all locally optimal placements in $O(n)$ time.

Theorem 2. *Given two sets $\tilde{P}$ and $\tilde{Q}$ of imprecise points of size m and n, respectively, where the discs in $\tilde{P}$ have constant intersection depth, we can compute $h_{\max}(\tilde{P}, \tilde{Q})$ and precise realisations $P \Subset \tilde{P}$ and $Q \Subset \tilde{Q}$ with $h(P, Q) = h_{\max}(\tilde{P}, \tilde{Q})$ in $O((m+n)\log(m+n))$ time.*

The algorithm described in this section works in the most general setting. However, in some more specific settings, the algorithm can be simplified. For example, when the discs of $\tilde{Q}$ are unit discs, the iaVD is simply the normal Voronoi diagram. When P is not imprecise, there are of course only m possible locations for $\hat{p}$, and we do not need to look for all three placement types. This results in the running times as indicated in Table 1.

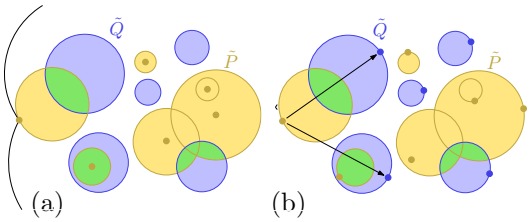

Fig. 4. (a) An example input. (b) The optimal output, shown as a set of circles covering Q.

3 Hardness results for tight lower bounds

In this section, we consider a transformation from the known NP-complete problem PLANAR 3-SAT [10] to the problem of computing $h_{\min}(P, \tilde{Q})$ for a set P of points and a set $\tilde{Q}$ of discs with radius r. In the PLANAR 3-SAT problem, we are given as input a 3-SAT formula f with the additional property that the graph $G(f)$ is planar, where $G(f)$ has a vertex for each variable and each clause in f, and there is an edge between a variable vertex and a clause vertex if the variable occurs in the clause. Having the boolean formula f and a planar embedding of $G(f)$, the transformation is as follows (see Fig. 5(a,b) for a general overview):
For each variable vertex v in $G(f)$, we construct a *cycle* C of alternating points in P and discs in $\tilde{Q}$. The distance between consecutive points and discs is ϵ, such that $r = 2.5\epsilon$ (see Fig. 5(c)). There may be bends up to a certain angle, and also other geometric features necessary to connect cycles and chains. When looking only at the points P^C and discs $\tilde{Q}^C$ corresponding to a cycle C, we observe that by the construction of C, there are two realisations $Q_0^C, Q_1^C \Subset \tilde{Q}^C$ such that $h(P^C, Q_0^C) = \epsilon$ and $h(P^C, Q_1^C) = \epsilon$. These two realisations represent the two possible boolean values the variable for that cycle can have.
For each edge $\{v, c\}$ in $G(f)$, we construct a *chain* of alternating points in P and discs in $\tilde{Q}$ with distance ϵ (see Fig. 5(d)). The chain connects the cycle corresponding to the variable v and the representation of a clause c. One end of this chain is a disc that will be part of a representation of clause c (see Fig. 5(e)),

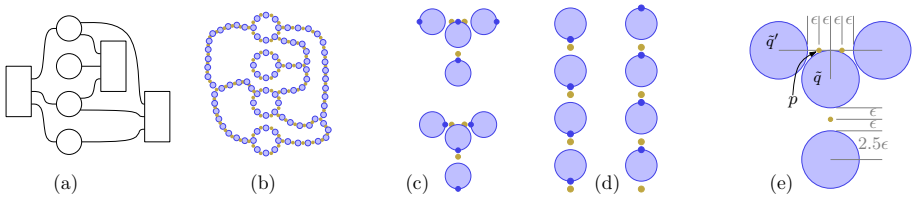

Fig. 5. (a) Planar embedding of $G(f)$, circles represent variables and rectangles represent clauses. (b) Rough overview of how $G(f)$ is transformed into P and $\tilde{Q}$, some details are misrepresented. the chain starts with p followed by $\tilde{q}'$, all other points and discs belong to the cycle. (c,d) Two realisations (representing opposite boolean values) with Hausdorff distance ϵ of chains, cycles and connections. (e) Connection of a chain to a cycle,

the other end is a point p that is placed near a disc $\tilde{q} \in \tilde{Q}$ of a variable cycle such that p has distance ϵ to either $Q_0^C \cap \tilde{q}$ or $Q_1^C \cap \tilde{q}$ (see Fig. 5(e)).
Each clause vertex in $G(f)$ is represented by three discs and one additional point p^*, such that the disc centres lie on the vertices of an equilateral triangle, and the point has distance ϵ to each of the discs. The three discs are ends of chains that connect to cycles that correspond to the three literals in the clause.

Theorem 3. *Let P be a precise point set and $\tilde{Q}$ be an imprecise point set of pairwise disjunct discs. It is NP-hard to compute a δ-approximation of the directed Hausdorff distance $h_{min}(P, \tilde{Q})$ for $1 \leq \delta < 3$.*

4 Algorithms for tight lower bounds

In this section we present algorithms for computing the minimum of $h(\tilde{P}, Q)$ and $h(P, \tilde{Q})$. As we have seen in the previous section, the latter problem is NP-hard and even hard to approximate in some settings. In the following we give a 4-approximation for the general case, an optimal 3-approximation for disjoint discs and an algorithm for the case which is not NP-hard when the Hausdorff distance is small. Many results in this section rely on similar ideas. Therefore, we will describe several (sub-) algorithms with different approximation factors and running times depending on the value d of the optimal solution. Afterwards, we discuss how to apply them to obtain the results claimed in Table 1.

4.1 Algorithm for precise Q

In this section, we describe an algorithm for the case where we have an imprecise point set $\tilde{P}$ and a precise point set Q. We place all points in $\tilde{P}$ as close to a point in Q as possible. Fig. 6(a) shows an example. For each pair $(\tilde{p}, q)$ with $\tilde{p} \in P$ and $q \in Q$ we could simply compute the placement $p \in \tilde{p}$ minimizing the Hausdorff distance and keep track of the longest distance over all pairs. This takes $O(mn)$ time. However, in practice it is probably better to compute the Voronoi diagram of Q first, and locate the discs of $\tilde{P}$ in it. In the worst case, each disc could still intersect linearly many Voronoi cells

(although the input needs to be contrived for this). Also, note that as soon as a disc from $\tilde{P}$ is discovered to contain a point from Q, we can stop the computation and just place the point there.

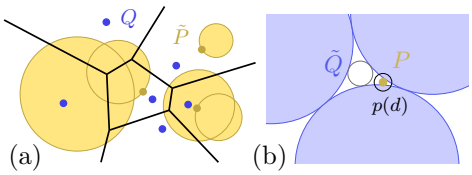

Fig. 6. (a) Placing points in P. (b) Discs of radius at most c can only intersect at most two discs of $\tilde{Q}$.

Theorem 4. *Let $\tilde{P}$ denote an imprecise point set consisting of m discs and Q denote a precise point set consisting of n points. The tight lower bound of $h(P, \tilde{Q})$ can be computed in $O(mn)$ time.*

4.2 Algorithms for imprecise $\tilde{Q}$

In the case where P is precise and $\tilde{Q}$ is imprecise, we have several simple and more involved algorithms. The algorithms are described in detail in the full version [9]; here we merely mention the resulting theorems.
First, we describe a simple subalgorithm Candidates to establish the following lemma.

Lemma 2. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. It is possible to reduce the possible values of $h_{min}(P, \tilde{Q})$ to $O(m^3 + m^2 n)$ many candidates in $O(m^3 + m^2 n)$ time.*

Next, we describe an algorithm IndependentSets , which is summarised in the following theorem.

Theorem 5. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n* disjoint *discs. Algorithm* IndependentSets *computes whether the tight lower bound for $h(P, \tilde{Q})$ is smaller than $r(\sqrt{5 - 2\sqrt{3}} - 1)/2$ where r is the radius of the smallest disc in $\tilde{Q}$. If this is the case, the exact value of $h_{min}(P, \tilde{Q})$ is computed. The running time is $O(m^3 + m^2 n + n \log^2 n)$.*

Finally, we describe another algorithm GrownDiscs .

Theorem 6. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. Given a c-approximation to the geometric k-covering problem that runs in $T(k, m)$ time, we can compute a $(c + 2)$-approximation to the tight lower bound of $h(P, \tilde{Q})$ in $O(m^3 + m^2 n + (n^2 T(k, m) + mn + m\sqrt{m}) \log(m+n))$ time, where $k \leq n$ is an internal parameter of the optimal solution.*

We now proceed to describe how to use those results and combine the algorithms to solve various variants of the problem we are interested in.
For the remainder of this section let $r_{\min}$ and $r_{\max}$ denote the radius of the smallest and largest disc in $\tilde{Q}$. When P is precise and $\tilde{Q}$ is imprecise, we note that

by Theorem 6 Algorithm GrownDiscs immediately presents a 4-approximation for the case when the discs may have different radii and overlap, which we obtain by plugging in a 2-approximation algorithm for geometric k-covering that runs in $O(m \log k)$ time [3]. The running time of the entire algorithm then becomes $O(m^3 + m^2 n + mn^2 \log(m+n) \log n)$ in the worst case.

We can improve this algorithm by first testing whether $v < c \cdot r_{\min}$ using Algorithm IndependentSets and Theorem 5, without increasing the asymptotic running time. If it is, then we can actually compute the exact solution.

Furthermore, when the discs are disjoint and all have the same size, we can improve this result to a 3-approximation by combining Algorithm GrownDiscs and a trivial algorithm called CentrePoints which simply places every imprecise point at the centre of its disc. First we test whether $v > r/2 = r_{\max}/2$, by applying CentrePoints and checking whether the resulting Hausdorff distance is larger than $3/2r$. If it is, we are done. Otherwise, note that each cell of $\mathcal{A}$ is a subset of the intersection of $k \leq 4$ discs, because $\tilde{Q}$'s discs are disjoint and $v < r/2$. Therefore, by Theorem 6 we can obtain a 3-approximation from Algorithm GrownDiscs by plugging in an exact algorithm to solve the geometric k-covering problem.

We can solve the geometric 4-covering problem exactly by computing the arrangement circles around the points to be covered or radius d. The arrangement has quadratic complexity. Then we need to find out whether there are three cells that are together in all cells. There are $O(m^8)$ such combinations to test, and by keeping track of which discs are already taken care of each can be tested in constant time. So, using this algorithm, we have a $1 + 2 = 3$-approximation to the original problem for disjoint unit discs. The total running time now becomes $O(n^2 m^8 \log(m+n))$.

Theorem 7. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n* disjoint *discs of the same radius. The tight lower bound for $h(P, \tilde{Q})$ is 3-approximable in time $O(m^3 + m^2 n + n \log^2 n)$.*

5 Conclusions and Future Work

We studied computing tight lower and upper bounds on the directed Hausdorff distance between two point set, when at least one of the sets has imprecision. We gave efficient exact algorithms for computing the upper bound, prove that computing the lower bound is NP-hard in most settings, and provide approximation algorithms. Furthermore, we show that in one special case, our approximation algorithm is optimal. In other settings, a gap in the factor between the hardness result and approximation still remains. When both sets are imprecise, we don't have any constructive results for the lower bound.

All our results hold for the directed Hausdorff distance. An obvious next step would be to extend them to the undirected Hausdorff distance. We can immediately solve the upper bound problem in that case using our results, since it is just the minimum of the two directed distances. However, computing lower

bounds seems to be more complicated, because there one needs to find a single placement of both point sets that minimises the distance in both directions at the same time.
Other directions of future work include looking at other underlying metrics than the Euclidean metric, other similarity measures than the Hausdorff distance, or, as is common in shape matching, allowing some transformation of the point sets.

References

1. H. Alt, B. Behrends, and J. Blömer. Approximate matching of polygonal shapes. *Ann. Math. Artif. Intell.*, 13:251–266, 1995.
2. H. Alt and L. Guibas. *Handbook on Computational Geometry*, chapter Discrete Geometric Shapes: Matching, Interpolation, and Approximation - A Survey, pages 251–265. 1995.
3. T. Feder and D. H. Greene. Optimal algorithms for approximate clustering. In *Proc. 20th Ann. ACM Symp. on Theory of Comp.*, pages 434–444, 1988.
4. S. J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.
5. M. T. Goodrich, J. S. B. Mitchell, and M. W. Orletsky. Practical methods for approximate geometric pattern matching under rigid motion. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 103–112, 1994.
6. M. T. Goodrich and J. Snoeyink. Stabbing parallel segments with a convex polygon. *Comput. Vision Graph. Image Process.*, 49:152–170, 1990.
7. L. J. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: building robust algorithms from imprecise computations. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 208–217, 1989.
8. P. J. Heffernan and S. Schirra. Approximate decision algorithms for point set congruence. *Comput. Geom. Theory Appl.*, 4:137–156, 1994.
9. C. Knauer, M. Löffler, M. Scherfenberg, and T. Wolle. The directed Hausdorff distance between imprecise point sets, 2009. Preprint, http://arXiv.org/abs/0909.4642.
10. D. Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11:329–343, 1982.
11. A. Mukhopadhyay, E. Greene, and S. V. Rao. On intersecting a set of isothetic line segments with a convex polygon of minimum area. In *Proc. 2007 International Conference on Computational Science and Its Applications*, volume 4705 of *LNCS*, pages 41–54, 2007.
12. A. Mukhopadhyay, C. Kumar, E. Greene, and B. Bhattacharya. On intersecting a set of parallel line segments with a convex polygon of minimum area. *Inf. Proc. Let.*, 105(2):58–64, 2008.
13. T. Nagai and N. Tokura. Tight Error Bounds of Geometric Problems on Convex Objects with Imprecise Coordinates. In *Japanese Conference on Discrete and Computational Geometry*, pages 252–263, 2000.
14. M. van Kreveld and M. Löffler. Largest bounding box, smallest diameter, and related problems on imprecise points. In *Proc. 10th Workshop on Algorithms and Data Structures*, LNCS 4619, pages 447–458, 2007.