

# 1 The directed Hausdorff distance between 2 imprecise point sets

3 *Christian Knauer\**    *Maarten Löffler<sup>†</sup>*    *Marc Scherfenberg\**  
4 *Thomas Wolle<sup>‡</sup>*

## 5 Abstract

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

## 13 1 Introduction

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

19 The *Hausdorff* distance is an important tool to measure the similarity between two sets of points  
20 (or, more generally, any two subsets of a metric space). It is defined as the largest distance from  
21 any point in one of the sets, to the closest point in the other set (see Section 2 for a formal  
22 definition). This distance is used extensively in pattern matching.

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

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

---

\*Institute of Computer Science, Freie Universität Berlin, Germany, {christian.knauer,scherfen}@mi.fu-berlin.de

<sup>†</sup>Computer Science Department, University of California, Irvine, USA, mloffler@uci.edu

<sup>‡</sup>NICTA Sydney, Australia, thomas.wolle@nicta.com.au

## 1 1.1 Related Work

2 The Hausdorff distance is one of the most studied similarity measures. For a survey about simi-  
 3 larity measures and shape matching refer to [3]. A straightforward, naive algorithm computes the  
 4 Hausdorff distance between two point sets  $A$  and  $B$  consisting of  $m$  and  $n$  points, respectively, in  
 5  $O(mn)$  time. Using Voronoi diagrams and a more sophisticated approach the running time can  
 6 be reduced to  $O((m+n)\log n)$  [2].

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

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

18 Nagai and Tokura [13] thoroughly study the computation of lower and upper bounds for a variety  
 19 of region shapes and measures; in particular they study the diameter, the width, and the convex  
 20 hull, and all their algorithms run in  $O(n \log n)$  time. However, not all of their bounds are tight. Van  
 21 Kreveld and Löffler [10] study the same problems and give algorithms to compute tight bounds,  
 22 though the running times of the algorithms can be much higher and some variants are proven to  
 23 be NP-hard.

24 Ahn *et al.* [1] study the efficient computation of the discrete Fréchet distance when the input  
 25 is imprecise. The discrete Fréchet distance is a distance function which measures the similarity  
 26 between two point sequences. The authors give polynomial time algorithms for computing the  
 27 lower bound and approximation algorithms for computing the upper bound and the lower and  
 28 upper bound of the distance under translation.

29 Despite the fact that both the Hausdorff distance and data imprecision are well-studied, we could  
 30 not find any previous work on the combination. Since the application areas that use the Hausdorff  
 31 distance (as well as other comparison measures) have to deal with imprecise data in practice, a  
 32 better understanding of the implications is needed.

## 33 1.2 Contribution

34 In this paper, we assume that an imprecise point is modelled by a disc with a given centre and  
 35 radius, that is, the real location of the point can be anywhere inside the specified disc. In general,  
 36 it is possible that the discs intersect. We assume we have two sets of points,  $P$  and  $Q$ , and that  
 37 at least one of them is imprecise. We want to compute the directed Hausdorff distance from  $P$   
 38 to  $Q$ . If  $P$  or  $Q$  is imprecise, then this does not lead to a unique number as answer anymore;  
 39 instead, we want to compute the smallest (lower bound) and largest (upper bound) value this  
 40 number can attain. Since there are three combinations of  $P$ ,  $Q$  or both being imprecise, and two  
 41 bounds to compute for each combination, we have different problems to solve. Additionally, some  
 42 of these problems become easier if we restrict the model of imprecision in some way, for example  
 43 by requiring the imprecision discs to be disjoint, or at least that they have a limited intersection  
 44 depth, or by requiring that all discs have the same radius. Our results are summarised in Table 1.

45 In the next section, we review some definitions and structures that we use to obtain our results.  
 46 After that, we present our three main results. In Section 3, we give a general algorithm for  
 47 computing the upper bound, which works in all settings in the table, though it can be simplified

setting	tight lower bound	tight upper bound
$h(\tilde{P}, Q)$ [general]	$O(n \log n)$	$O(n \log n)$
$h(P, \tilde{Q})$ [general]	NP-hard*, 4-APX in $O(n^3 \log^3 n)$	$O(n \log n)$
[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 $\tilde{P}$ ]		$O(n \log n)$

Tab. 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  $O(n^3)$  time 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  $\tilde{Q}$ .

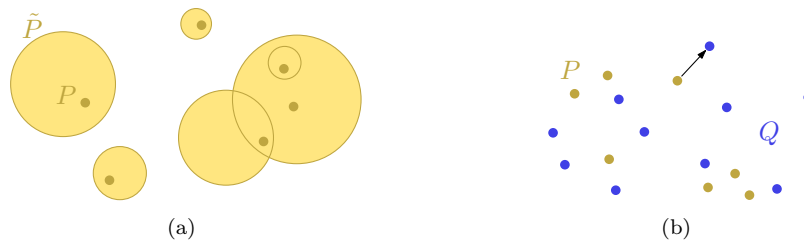


Fig. 1: (a) An example of imprecise point sets. (b)  $h(P, Q)$  is realised by the pair of points indicated by the arrow.

1 (conceptually) in some settings. In Section 4, we prove the hardness of computing the lower bound  
 2 in most settings. Finally, in Section 5, we give algorithmic results for computing the lower bound,  
 3 exactly in some cases and approximately in others.

## 4 2 Preliminaries

5 In this section, we will review a number of known concepts and structures from computational  
 6 geometry, as well as define variations of them that we will need later.

7 **Imprecise Points** An *imprecise point* is a closed disc in the plane. Let  $\tilde{P} = \{\tilde{p}_1, \dots, \tilde{p}_m\}$  and  
 8  $\tilde{Q} = \{\tilde{q}_1, \dots, \tilde{q}_n\}$  denote two imprecise point sets consisting of  $m$  and  $n$  closed discs, respectively.

9 We call a point set  $P = \{p_1, \dots, p_m\}$  a *precise realisation* of  $\tilde{P} = \{\tilde{p}_1, \dots, \tilde{p}_m\}$  if  $p_i \in \tilde{p}_i$  for all  $i$ .  
 10 We also write  $P \in \tilde{P}$  in this case.

11 Figure 1(a) shows an example of such a set of imprecise points and a possible precise realisation.

**Hausdorff Distance** The *directed Hausdorff distance*  $h$  from a point set  $P = \{p_1, \dots, p_m\}$  to a  
 point set  $Q = \{q_1, \dots, q_n\}$  with an underlying Euclidean metric is defined as

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

12 and can be computed in  $O((n + m) \log n)$  time, see Alt *et al.* [2]. An example is shown in Fig-  
 13 ure 1(b).

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.

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

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

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

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

1 **Voronoi Diagrams** The *Voronoi diagram* of a set  $Q$  of  $n$  points (or sites) in the plane is the  
 2 decomposition of the plane into cells such that all points in a cell share a common nearest site  
 3 in  $Q$ . An *additively weighted* Voronoi diagram (also known as Apollonius diagram) of a set of  
 4 weighted sites is a generalisation of the Voronoi diagram. Intuitively, we can view the weighted  
 5 sites as discs whose radii correspond to their weights. If the weights are positive and the resulting  
 6 discs do not overlap then the additively weighted Voronoi diagram corresponds to the Voronoi  
 7 diagram of the discs. Formally, the additively weighted Voronoi diagram (of a set  $Q$  of weighted  
 8 sites) decomposes the plane into cells (such that all points in a cell share a common nearest site  
 9 in  $Q$ ), where the weight of a site is subtracted from its distance to a point in the plane. Note that  
 10 this definition does not require the weights to be positive. Observe that adding a constant amount  
 11 to all weights does not change the structure of the diagram. Hence it is possible to use a Voronoi  
 12 diagram data structure which only handles non-negative weights by adding the absolute amount of  
 13 the largest disc radius to all weights. Voronoi diagrams and additively weighted Voronoi diagrams  
 14 can be computed in  $O(n \log n)$  time [5].

15 For a given set  $\tilde{Q}$  of  $n$  discs in the plane, we define the *inverted additive Voronoi diagram* or *iaVD*  
 16 to be the additively weighted Voronoi diagram whose sites are the centres of the discs, where the  
 17 weight of each site corresponds to the negative radius of the disc. Hence, the iaVD partitions the  
 18 plane into cells, where a point  $x$  in the plane is associated to a disc in  $\tilde{Q}$  in the following way: For  
 19 point  $x$ , we consider the point in each disc in  $\tilde{Q}$  that is furthest away from  $x$ , and among those  
 20 points, the one that is closest to  $x$  determines the disc we associate with  $x$ . See Figure 3(a) for an  
 21 example. The edges of the iaVD are pieces of hyperbolae. Adding the radius of the largest disc to  
 22 the weight of each site we get an additively weighted Voronoi diagram with non-negative weights  
 23 only, and such the iaVD can be computed in  $O(n \log n)$  time.

24 **Geometric  $k$ -Centre** In the *geometric  $k$ -centre* problem, we are given a set  $P$  of points in the  
 25 plane which should be covered by  $k$  discs of the same radius. The objective is to minimise the  
 26 radius of these discs. This problem is known to be NP-hard, but can be approximated within a  
 27 factor 2 [4] in  $O(n \log k)$  time.

28 **Matching** The *maximum matching* problem in a bipartite graph  $G = (V, E)$  is to find a set of  
 29 vertex-disjoint edges that is as large as possible. Using the algorithm of Hopcroft and Karp [8],  
 30 this problem can be solved optimally in  $O(|E|\sqrt{|V|})$  time.

### 31 3 Algorithms for computing the tight upper bound

32 In this section, we consider the following problem. Let  $\tilde{P}$  and  $\tilde{Q}$  be two sets of discs, of size  $m$   
 33 and  $n$ , respectively. The radii may all be different; an example input is shown in Figure 2(a).  
 34 Our aim is to compute point sets  $P \in \tilde{P}$  and  $Q \in \tilde{Q}$  such as to maximise the directed Hausdorff

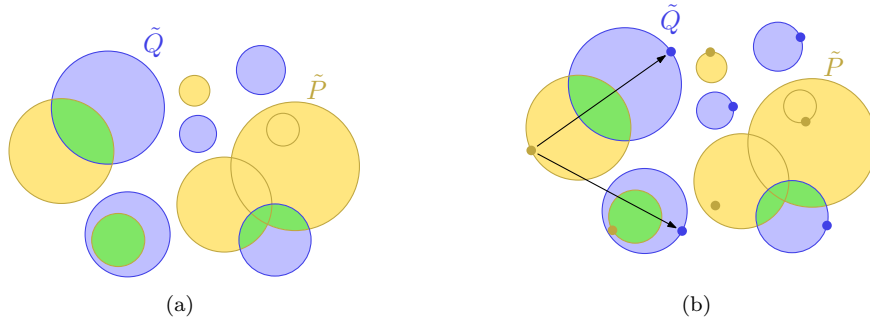


Fig. 2: (a) An example input of two imprecise point sets. (b) The optimal solution. The points in  $Q$  are all placed as far away from  $\hat{p}$  as possible.

1 distance  $h(P, Q)$ . In other words, we want to place the points in  $P$  and  $Q$  such that one point of  
 2  $P$  is as far away as possible from all points in  $Q$ . The placements of the remaining points of  $P$  do  
 3 not matter. So, we need to identify which point  $\hat{p} \in P$  will play this important role. We need to  
 4 place  $\hat{p}$  such that after we placed all points in  $Q$  as far away from  $\hat{p}$  as possible, this distance is  
 5 maximised. Figure 2(b) shows the optimal solution for the example.

6 In the next section, we first describe a basic algorithm that solves the problem in quadratic time.  
 7 After that, we show that under certain conditions, the running time can be improved to  $O(n \log n)$ .

### 8 3.1 Basic algorithm

9 We will first compute the *inverted additive Voronoi Diagram* (iaVD) of  $\tilde{Q}$  in  $O(n \log n)$  time.  
 10 Recall from Section 2 that this is the additively weighted Voronoi diagram of the centres of  $\tilde{Q}$ ,  
 11 weighted by their negative radii. Using this iaVD, we can identify three possible placement types  
 12 for  $p \in \tilde{p} \in \tilde{P}$  that are locally optimal inside  $\tilde{p}$ , as is illustrated in Figure 3(b):

- 13 1. A vertex of the iaVD that lies inside disc  $\tilde{p}$ .
- 14 2. An intersection point between a Voronoi edge and the boundary of  $\tilde{p}$ .
- 15 3. A point on the boundary of  $\tilde{p}$  that is furthest away from a site in the iaVD. This point lies  
 16 in the Voronoi cell that also contains the centre of  $\tilde{p}$ , since it lies on the line going through  
 17 the centre of its Voronoi site and the centre of  $\tilde{p}$  and every Voronoi cell is star shaped.

18 We can then iterate over all  $\tilde{p} \in \tilde{P}$  and its locally optimal placements, and we place a point  $p$   
 19 at the local optimal placements, as if it were  $\hat{p}$ . We determine  $\hat{p}$  by keeping track of the locally  
 20 optimal placement  $p \in \tilde{p}$  such that the shortest distance between  $p$  and (the furthest point on)  
 21 any disc in  $\tilde{Q}$  is maximised.

22 Once  $\hat{p}$  is known, we place all points in  $Q$  as far away from  $\hat{p}$  as possible, and all points in  $P \setminus \{\hat{p}\}$   
 23 anywhere inside their discs. The result is shown in Figure 2(b). As it is possible that there are  
 24  $O(mn)$  locally optimal placements of the second type (namely: an intersection between a disc  
 25 boundary and a Voronoi edge), we conclude with the following theorem.

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

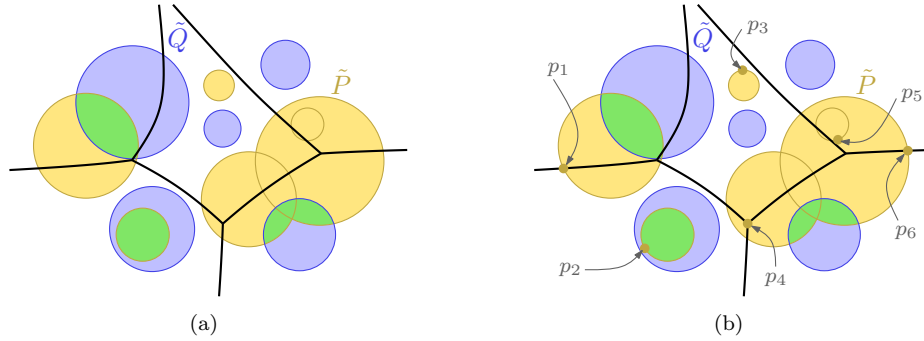


Fig. 3: (a) The inverted additive Voronoi Diagram (iaVD) of  $\tilde{Q}$ . (b) The point set  $P$  placed locally optimal;  $p_4$  is a type 1 placement,  $p_1, p_6$  are type 2 and  $p_2, p_3, p_5$  are type 3.

### 1 3.2 Faster algorithms in special cases

2 In practice, it may be unlikely that we have to consider  $O(mn)$  locally optimal placements. Indeed,  
 3 in this section we show how the above result can be improved under certain assumptions. To speed  
 4 up the algorithm, we make some observations about the nature of locally optimal placements.

5 **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 part of the bisector  
 6 of  $\tilde{q}_1$  and  $\tilde{q}_2$  forms an edge  $e$  of the iaVD that slices through  $\tilde{p}$  (that is,  $e$  is not connected to a  
 7 vertex of the iaVD inside  $\tilde{p}$ ). Then the optimal placement of  $p$  occurs on the same side of  $e$  where  
 8 the centre of  $\tilde{p}$  is.

9 **Proof:** 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  
 10 point on the boundary of  $\tilde{p}$  that is furthest away from  $q_{c1}$  (this would be a type 3 placement if  
 11  $\tilde{q}_1$  was the only element of  $\tilde{Q}$ ), and similarly let  $f_2$  be the point furthest away from  $q_{c2}$ . Observe  
 12 that  $e$  is part of a hyperbolic arc, of which  $q_{c1}$  and  $q_{c2}$  are the foci. Suppose w.l.o.g. that  $p_c$  is on  
 13 the same side of  $e$  as  $q_{c1}$ .

14 Now, suppose that the optimal placement  $p$  is on the other side of  $e$  (that is, on the side of  $q_{c2}$ ).  
 15 Then  $q_{c2}$ ,  $p_c$  and  $f_2$  lie on a line. Because  $q_{c2}$  is a focus of  $e$ , the half-line starting from  $q_{c2}$  in  
 16 the direction of  $p_c$  and  $f_2$  intersects  $e$  only once. Since  $q_{c2}$  and  $p_c$  lie on opposite sides of  $e$ , it  
 17 follows that  $p_c$  and  $f_2$  must lie on the same side of  $e$ . This means that along the boundary of  
 18  $\tilde{p}$ , the intersection points with  $e$  have a better value than any other point on the side of  $q_{c2}$ , in  
 19 particular, better than  $p$ , which is a contradiction. (Note that if there are other cells of the iaVD  
 20 involved, the value of  $p$  could only be lower).  $\square$

21 This lemma basically says that if we want to place a certain point  $p$  locally optimally, we can  
 22 start looking by walking from the centre of  $\tilde{p}$  and never have to cross edges of the iaVD that  
 23 slice through  $\tilde{p}$ . This can still mean, however, that we have to inspect a quadratic number of  
 24 placements, but we can quantify it as follows.

25 **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  
 26 find the locally optimal placement for  $p$  in  $O(t)$  time.

27 This immediately implies that if the discs of  $\tilde{P}$  do not overlap, we can simply place all points  $p$   
 28 independently in linear time. Figure 4 shows an example where there are a quadratic number of  
 29 placements of type 2, but which do not all have to be inspected because of Lemma 1.

30 In fact, we can show the same result for something slightly stronger than disjoint discs. Now,  
 31 assume that the intersection depth of the discs of  $\tilde{P}$  is at most some constant  $c$ . Then, clearly,

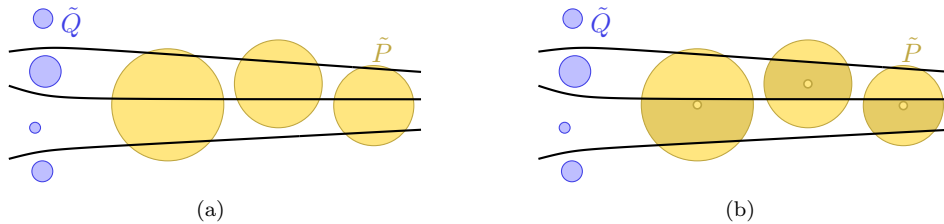


Fig. 4: (a) There could be a quadratic number of intersections between the edges of the iaVD of  $\tilde{Q}$  and the discs in  $\tilde{P}$ . (b) However, when an edge slices through a disc  $\tilde{p}$ , we only need to inspect the cell that contains the centre of  $\tilde{p}$ .

1 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  
 2 of the iaVD, we have  $\sum_i t_i \leq cn$ , and we can find all locally optimal placements in  $O(n)$  time.

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

6 The algorithm described in this section works in the most general setting. In some more specific  
 7 settings, the algorithm can be simplified. For example, if  $Q$  is a set of precise points, or if the  
 8 regions of  $\tilde{Q}$  are unit discs, the iaVD is the normal Voronoi diagram. This, however, does not  
 9 influence the running time. If, on the other hand,  $P$  is a set of precise points, then there are only  
 10  $m$  possible locations for  $\hat{p}$ , and we do not need to look for all three placement types. In this case,  
 11 we can compute  $h_{\max}(P, \tilde{Q})$  more efficiently than in the general case, as stated in the following  
 12 theorem.

13 **Theorem 3:** Given a set  $P$  of points and a set  $\tilde{Q}$  of imprecise points of size  $m$  and  $n$ , respectively,  
 14 we can compute  $h_{\max}(P, \tilde{Q})$  and a precise realisation  $Q \in \tilde{Q}$  with  $h(P, Q) = h_{\max}(P, \tilde{Q})$  in  $O((m+n)\log n)$   
 15 time.

## 16 4 Hardness results for tight lower bounds

17 In this section, we consider a transformation from the known NP-complete problem PLANAR 3-  
 18 SAT [9] 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  
 19  $r$ . In the PLANAR 3-SAT problem, we are given as input a 3-SAT formula  $f$  with the additional  
 20 property that the graph  $G(f)$  is planar, where  $G(f)$  has a vertex for each variable and each clause  
 21 in  $f$ , and there is an edge between a variable vertex and a clause vertex if the variable occurs in  
 22 the clause. Having the boolean formula  $f$  and a planar embedding of  $G(f)$ , the transformation is  
 23 as follows (see Fig. 5(a,b) for a general overview):

24 For each variable in  $f$  (or variable vertex  $v$  in  $G(f)$ ), we construct a *cycle*  $C$  of alternating points in  
 25  $P$  and discs in  $\tilde{Q}$ . The discs have fixed radius  $r$ , and the distance between consecutive points and  
 26 discs along the cycle is  $\epsilon$ , such that  $r = 2.5\epsilon$  (see Fig. 5(c)). There may be bends up to a certain  
 27 angle, and also other geometric features necessary to connect cycles and chains (defined below).  
 28 When looking only at the points  $P^C$  and discs  $\tilde{Q}^C$  corresponding to a cycle  $C$ , we observe that by  
 29 the construction of  $C$ , there are two realisations  $Q_0^C$  and  $Q_1^C$  of  $\tilde{Q}^C$ , such that  $h(P^C, Q_0^C) = \epsilon$  and  
 30  $h(P^C, Q_1^C) = \epsilon$  (see Fig. 5(d) and 6(c)). These two realisations represent the two possible boolean  
 31 values the variable for that cycle can have. Similar observations can also be made about chains to  
 32 be described next.

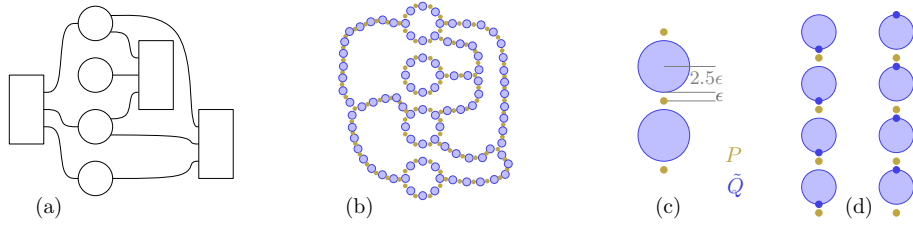


Fig. 5: (a) planar embedding of  $G(f)$ , circles represent variables and rectangles represent clauses; (b) rough overview of how  $G(f)$  is transformed into  $P$  and  $\tilde{Q}$ , some details are misrepresented; (c) alternating points and discs with geometric details; (d) two realisations with Hausdorff distance  $\epsilon$ , representing opposite boolean values;

1 For each edge  $\{v, c\}$  in  $G(f)$ , we construct a *chain* of alternating points in  $P$  and discs in  $\tilde{Q}$ ,  
2 where the discs have radius  $r$ , and the distance between consecutive points and discs along the  
3 chain is  $\epsilon$  (see Fig. 5(c,d)). The chain connects the cycle corresponding to the variable  $v$  and the  
4 representation of clause  $c$ . More precisely, one end of this chain is a disc that will be part of a  
5 representation of clause  $c$  (see below for details). And the other end of this chain is a point  $p$  that  
6 is placed near a disc  $\tilde{q} \in \tilde{Q}$  of the cycle  $C$  for variable  $v$ , such that  $p$  has a distance of  $\epsilon$  to either  
7  $Q_0^C \cap \tilde{q}$  or  $Q_1^C \cap \tilde{q}$ , depending on whether  $v$  occurs negated or non-negated in  $c$  (see Fig. 6(b,d)).  
8 Also chains may have bends.

9 Each clause (or clause vertex  $c$  in  $G(f)$ ) is represented by three discs and one additional point  $p^*$ ,  
10 such that the disc centres lie on the vertices of an equilateral triangle, and the point has distance  
11  $\epsilon$  to each of the discs, and the three discs are ends of chains that connect to cycles that correspond  
12 to the three literals in the clause (see Fig. 6(a)).

13 **Theorem 4:** Let  $P$  be a precise point set and  $\tilde{Q}$  be an imprecise point set of pairwise disjoint discs.  
14 It is NP-hard to compute a  $\delta$ -approximation of the directed Hausdorff distance  $h_{\min}(P, \tilde{Q})$  for  
15  $1 \leq \delta < 3$ .

16 **Proof:** For a given instance  $f$  to the PLANAR 3-SAT problem, let  $G(f)$  be the planar graph  
17 corresponding to  $f$ , embedded such that all variables are on a line, and all clauses are on either  
18 side of them, see Fig 5(a) ( $G(f)$  can always be drawn in this way [9]). From this embedding, we  
19 compute (as described above) a set  $P$  of precise points, a set  $\tilde{Q}$  of imprecise points, and numbers  
20  $\epsilon > 0$  and  $r = 2.5\epsilon$ .

21 **Claim 1:** If  $f$  is satisfiable, then  $h_{\min}(P, \tilde{Q}) \leq \epsilon$ .

22 **Proof of Claim 1:** We consider an assignment with boolean values of the variables in  $f$ , such that  
23  $f$  is satisfied, and we need to show that there exists a realisation  $Q \subseteq \tilde{Q}$ , such that  $h(P, Q) \leq \epsilon$ .  
24 (Note that by construction, there is no realisation  $Q' \subseteq \tilde{Q}$ , such that  $h(P, Q') < \epsilon$ .) For each cycle  
25  $C$  of a variable, we choose either  $Q_0^C$  or  $Q_1^C$  as realisations of the imprecise point set  $\tilde{Q}^C$ , depending  
26 on whether the variable is false or true. Discs on chains are realised in the following way: at the  
27 ending of the chain that connects to a cycle  $C$ , we have a point  $p$  near a disc  $\tilde{q} \in \tilde{Q}^C$ , and  $\tilde{q}$  is  
28 realised by a point  $q$ . The next object along the chain is a disc  $\tilde{q}'$  (see Fig. 6(b)). We realise  $\tilde{q}'$  in  
29 either of two ways as depicted in Fig. 6(c), depending on whether the distance between  $p$  and  $q$   
30 is equal to  $\epsilon$  or greater than  $\epsilon$ . This corresponds to a variable being either true or false. And the  
31 boolean value of the corresponding literal is then propagated to the other end of the chain to a  
32 clause  $c$ . Since  $f$  is satisfiable, there is at least one literal in each clause that satisfies the clause.  
33 Hence, there is at least one chain with a realisation such that the point  $p^*$  has distance at most  $\epsilon$   
34 to a point of this realisation.  $\square$



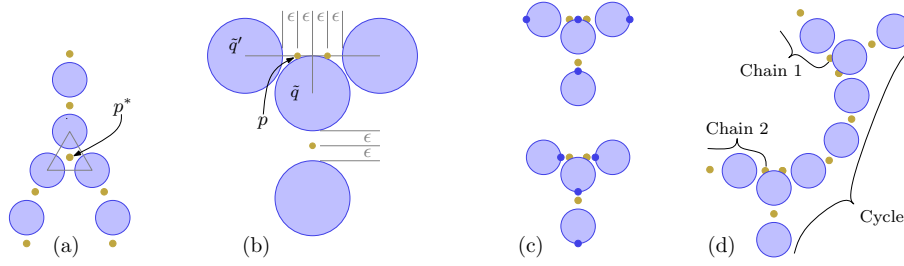


Fig. 6: (a) the endings of three chains arranged to representing a clause; (b) connection of a chain to a cycle with geometric details, the chain starts with  $p$  followed by  $\tilde{q}'$ , all other points and discs belong to the cycle; (c) two realisations with Hausdorff distance  $\epsilon$  of the structure in the left subfigure; (d) two chains connecting to the same cycle where the chains tap opposite boolean values;

1 Claim 2: If  $f$  is not satisfiable, then  $h_{\min}(P, \tilde{Q}) \geq 3\epsilon$ .

2 **Proof of Claim 2:** We will prove the contraposition, namely if  $h_{\min}(P, \tilde{Q}) < 3\epsilon$ , then  $f$  is  
 3 satisfiable. We consider a realisation  $Q \in \tilde{Q}$  with  $h(P, Q) < 3\epsilon$ , and we need to construct a  
 4 variable assignment that satisfies  $f$ . We first observe that the only way where two points in  $P$   
 5 can be matched to the same point  $q \in \tilde{Q}$  is where a chain connects to a cycle. (Otherwise,  
 6 the distance between the two points in  $P$  is larger than  $6\epsilon$ , and hence, they cannot be matched to  
 7 the same point  $q$ .) And in this case, one of the points in  $P$  is the end point of a chain, the other  
 8 point in  $P$  belongs to a cycle, and  $\tilde{q}$  belongs to the same cycle (see Fig. 6(b)). From this we make  
 9 an observation about how the points along chains and cycles are matched to discs along the same  
 10 chains and cycles. Let us consider the sequence  $p_0, \tilde{q}_0, p_1, \tilde{q}_1, p_2, \tilde{q}_2, \dots$  of points and discs ordered  
 11 along a fixed cycle  $C$ . Exactly one of the following two things is true for all  $i = 0, 1, 2, \dots$  (modulo  
 12 length of  $C$ ):

- 13 •  $p_i$  is matched to a point  $q_i \in \tilde{q}_i$ , i.e.  $\|p_i, q_i\| < 3\epsilon$ ; or  
 14 •  $p_i$  is matched to a point  $q_{i-1} \in \tilde{q}_{i-1}$ , i.e.  $\|p_i, q_{i-1}\| < 3\epsilon$

15 In other words, each point on  $C$  is matched to the next disc on  $C$  in clockwise order, or each point  
 16 on  $C$  is matched to the next disc on  $C$  in counter-clockwise order, but there is no mix of these  
 17 along  $C$ . From these two possibilities for cycle  $C$ , we derive the boolean value of the variable  
 18 corresponding to  $C$ . This assignment is in accordance with the two realisations  $Q_0^C$  and  $Q_1^C$  (as  
 19 defined above), which represent false and true. What is left to show is that this assignment satisfies  
 20  $f$ . To see this, we consider any clause  $c$  of  $f$  and argue that  $c$  is satisfied. From the construction,  
 21 we know that  $c$  is represented by one point  $p^* \in P$  and three discs being the endings of three  
 22 chains. There must be a point  $q \in Q$  such that  $\|p^*, q\| < 3\epsilon$ , and  $q$  must lie in one of the discs  
 23 that represent the clause  $c$ . This disc  $\tilde{q}_0$  is the ending of a chain  $\tilde{q}_0, p_0, \tilde{q}_1, p_1, \tilde{q}_2, \dots, p_j$ . In a similar  
 24 way as above, we conclude for this chain that:

- 25 •  $p^*$  is matched to a point  $q_0 \in \tilde{q}_0$ , i.e.  $\|p^*, q_0\| < 3\epsilon$ ; and  
 26 •  $p_i$  is matched to a point  $q_{i+1} \in \tilde{q}_{i+1}$ , for  $i = 0, \dots, j-1$ ; and  
 27 •  $p_j$  is matched to a point  $q_j \in \tilde{q}_j$ , for some disc  $\tilde{q}_j$  on some cycle  $C$

28 The variable corresponding to  $C$  has a boolean value, according to the realisation of the discs  
 29 along  $C$ . Depending on whether this variable occurs negated or non-negated in the clause  $c$ ,

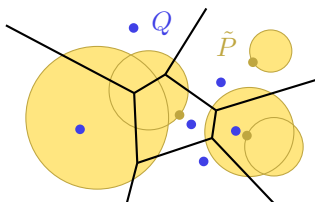


Fig. 7: Placing points in  $P$  as close as possible to their closest neighbour in  $Q$ . A point in  $P$  is hidden by a point in  $Q$  when the points are identical.

- 1 the chain  $\tilde{q}_0, p_0, \tilde{q}_1, p_1, \tilde{q}_2, \dots, p_j$  is connected to the cycle  $C$ , such that “the boolean value true is
- 2 propagated along the chain”. Hence, by construction we have that the boolean value of the variable
- 3 corresponding to  $C$  satisfies the clause  $c$ .  $\boxtimes$
- 4 The proof of Theorem 4 follows now from Claim 1 and 2, and from observing that the construction
- 5 can be done without any intersection between discs and/or points, and such that chains and/or
- 6 cycles are far enough apart from each other not to interfere. We also note that the size of  $P$  and
- 7  $Q$  is polynomial in the size of  $G(f)$ , which follows from our planar embedding of  $G(f)$ .  $\boxtimes$

## 8 5 Algorithms for tight lower bounds

9 In this section we present algorithms for computing the minimum of  $h(\tilde{P}, Q)$  and  $h(P, \tilde{Q})$ . As we  
 10 have seen in the previous section, the latter problem is NP-hard and even hard to approximate  
 11 in some settings. In the following we give a 4-approximation for the general case, an optimal  
 12 3-approximation for disjoint discs and an algorithm for the case which is not NP-hard when the  
 13 Hausdorff distance is small. Many results in this section rely on similar ideas. Therefore, we  
 14 will describe several (sub-) algorithms with different approximation factors and running times  
 15 depending on the value  $d$  of the optimal solution. Afterwards, we discuss how to apply them to  
 16 obtain the results claimed in Table 1.

### 17 5.1 Algorithm PLACETOGETHER

18 In this section, we describe an algorithm for the case where we have an imprecise point set  $\tilde{P}$  and  
 19 a precise point set  $Q$ . We place all points in  $\tilde{P}$  as close to a point in  $Q$  as possible. Fig. 7 shows  
 20 an example. For each pair  $(\tilde{p}, q)$  with  $\tilde{p} \in \tilde{P}$  and  $q \in Q$  we could simply compute the placement  
 21  $p \in \tilde{p}$  minimizing the Hausdorff distance and keep track of the longest distance over all pairs. This  
 22 takes  $O(mn)$  time.

23 However, unless  $n$  is exponentially larger than  $m$  we can do better because of the following obser-  
 24 vation:

25 **Observation 1:** Given a disk  $\tilde{p}$  and a point set  $Q$ . The point in  $Q$  closest to the centre of  $\tilde{p}$  is also  
 26 closest to any point in  $\tilde{p}$ .

27 As a consequence, we can do the following. First we compute, in  $O(n \log n)$  time, the Voronoi  
 28 diagram of  $Q$ . Then, we locate the centre of each point in  $\tilde{P}$  in this Voronoi diagram in total  
 29  $O(m \log n)$  time. Once located, we place in constant time each point  $p \in \tilde{p}$  as close as possible to  
 30 the site whose Voronoi cell contains the centre of  $\tilde{p}$ .

31 **Theorem 5:** Let  $\tilde{P}$  denote an imprecise point set consisting of  $m$  discs and  $Q$  denote a precise point  
 32 set consisting of  $n$  points. The tight lower bound of  $h(\tilde{P}, Q)$  can be computed in  $O(\min\{mn, (m +$   
 33  $n) \log n\})$  time.

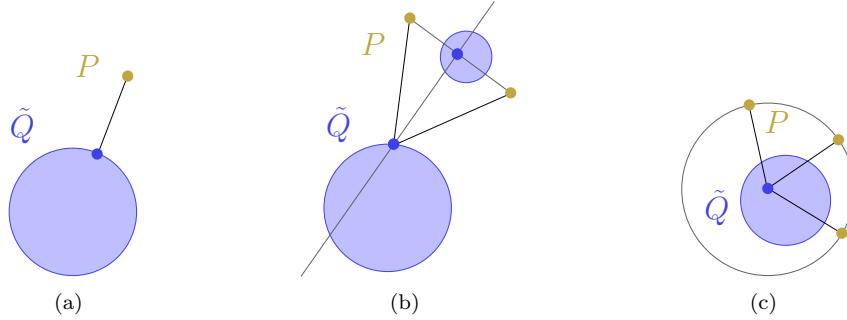


Fig. 8: There are only polynomially many candidates for the infimum of  $h(P, \tilde{Q})$  which are determined by (a) one point of  $P$ , (b) by two points of  $P$  or (c) three or more points of  $P$ .

## 1 5.2 Subalgorithm CANDIDATES

2 In the case where  $P$  is precise and  $\tilde{Q}$  is imprecise, we start with a simple subalgorithm CANDIDATES  
3 to establish Lemma 2. The result will be used later.

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

7 The Hausdorff distance is realised locally at one or several points of  $Q$  and  $P$ , i.e. we only need to  
8 look at the placements for these points of  $Q$ . Let  $q \in \tilde{q} \in \tilde{Q}$  be such a placement of an imprecise  
9 point in  $\tilde{Q}$  which realises the Hausdorff distance  $d$ . The distance  $d$  can be determined by  $q$  together  
10 with one or several points of  $P$ . These points may not be the only points of  $P$  which are matched  
11 to  $q$ , but among the points matched to  $q$  they have the largest distance to  $q$ . If  $d$  is determined  
12 by one point of  $P$  there are  $O(mn)$  possibilities, see Fig. 8(a). If  $d$  is determined by two points  
13  $p_1, p_2 \in P$  the point  $q$  lies on the bisector of the line segment  $p_1p_2$ , see Fig. 8(b), for which  $O(nm^2)$   
14 possibilities exist. Finally, if  $d$  is determined by three or more points, all these points lie on a circle  
15 whose centre is  $q$ . Such a circle is determined by three of its points and thus there are  $O(m^3)$   
16 possible locations, see Fig. 8(c). The algorithm simply computes and returns all  $O(m^3 + m^2n)$   
17 locations in  $O(m^3 + m^2n)$  time.

## 18 5.3 Algorithm INDEPENDENTSETS

19 This algorithm computes exactly the Hausdorff distance from a precise point set  $P$  to an imprecise  
20 point set  $\tilde{Q}$  when the distance is small. This is an exception to the general NP-hardness of that  
21 setting.

22 **Theorem 6:** Let  $P$  denote a precise point set consisting of  $m$  points and  $\tilde{Q}$  denote an imprecise  
23 point set consisting of  $n$  disjoint discs. Algorithm INDEPENDENTSETS computes whether the tight  
24 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  
25 disc in  $\tilde{Q}$ . If this is the case, the exact value of  $h_{\min}(P, \tilde{Q})$  is computed. The running time is  
26  $O(m^3 + m^2n + n \log^2 n)$ .

27 First we compute the set of possible candidates for  $h_{\min}(P, \tilde{Q})$  by CANDIDATES and discard all  
28 values greater or equal than  $c = r(\sqrt{5 - 2\sqrt{3}} - 1)/2$ . Now we perform a binary search on the

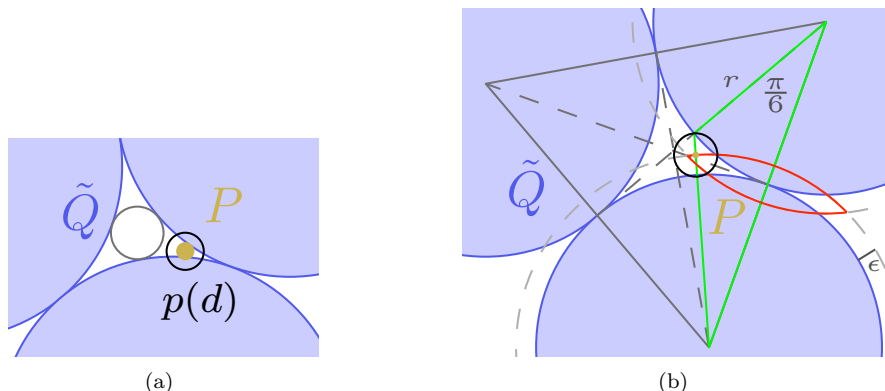


Fig. 9: (a) The gray circle in the middle has the minimal radius of  $r(2/\sqrt{3}-1)$  which is necessary to intersect at least three discs of  $\tilde{Q}$ . Because all considered candidates for the minimal Hausdorff distance are smaller than the radius of the gray disc, there are at most two possible matching partners for each point in  $P$ . (b) The black circle has the maximal radius  $c$  for which no two circles intersecting two different pairs of discs in  $\tilde{Q}$  can be stabbed by only one point  $q \in \tilde{q}$ . Since it has to intersect two discs in  $\tilde{Q}$ , its centre must lie within the red lune of these discs grown by  $c$ . Furthermore, its boundary must not intersect the green segment denoted by  $r$  because it could intersect another black circle intersecting the upper two discs of  $\tilde{Q}$ , otherwise. Using the cosine formula it holds that  $(r+2c)^2 < r^2 + (2r)^2 - 2r2r \cos \frac{\pi}{6}$ . Solving the latter inequality for  $c$  yields  $c < r(\sqrt{5} - 2\sqrt{3} - 1)/2$ .

1 remaining values in order to determine the smallest value  $d$  for which the predicate described  
 2 below evaluates to true. If such a candidate exists, the algorithm returns  $d$  as the value of the  
 3 bound. Otherwise  $h_{\min}(P, \tilde{Q}) \geq r(\sqrt{5} - 2\sqrt{3} - 1)/2$ .

4 Let  $p(d)$  denote the disc of radius  $d$  around a point  $p \in P$ . There must be at least one point of  $Q$   
 5 in  $p(d)$  to which  $p$  can be matched within a distance smaller or equal than  $d$ . The computation of  
 6 the predicate relies on two observations: All considered values are so small that no  $p(d)$  intersects  
 7 more than two discs of  $\tilde{Q}$ . Note that a disc that intersects more than two disjoint discs of  $\tilde{Q}$  has  
 8 a radius of at least  $r(2/\sqrt{3}-1)$ , which is greater than  $c$ , see Fig. 9(a). Thus, there are at most  
 9 two possible matching partners for each point  $p \in P$ .

10 The second observation is that each  $p(d)$  has to intersect at least one disc of  $\tilde{Q}$ , otherwise the  
 11 Hausdorff distance would be greater than  $d$  at  $p$ .

12 We define a point  $p \in P$  to have degree 1 if  $p(d)$  intersects just one disc  $\tilde{q} \in \tilde{Q}$  and to have degree  
 13 2 if it intersects two discs of  $\tilde{Q}$ .

14 The predicate tests, whether  $h_{\min}(P, \tilde{Q}) \leq d$ . To this end we associate with each  $\tilde{q}_i \in \tilde{Q}$  a *feasible*  
 15 *region*  $F_i$  and a set  $C_i$  called *children* of  $\tilde{q}_i$ . The feasible region contains the valid placements of  
 16  $q_i \in \tilde{q}_i$ . The children of a point  $\tilde{q}_i$  are the points of  $P$  that can only be matched to  $\tilde{q}_i$  because  
 17 otherwise  $h_{\min}(P, \tilde{Q})$  would be greater than  $d$ . In other words, the children of  $\tilde{q}_i$  demand that  $q_i$   
 18 is placed in its feasible region  $F_i$ . See Fig. 10(a) for an illustration.

19 We restrict the feasible regions and children in an iterative manner with the help of the sub-  
 20 algorithms REMOVE-DEGREE-1-DISCS and REMOVE-DEGREE-2-DISCS. If a feasible region turns  
 21 out to be empty, the computation stops and the predicate returns false. The first sub-algorithm  
 22 considers only points  $p$  of degree 1 and restricts the feasible regions of the discs intersected by  $p(d)$ .  
 23 The second sub-algorithm restricts the feasible regions of discs intersected by points of degree 2.

24 Before calling REMOVE-DEGREE-1-DISCS we define a set  $P_R$  of all points which are not matched  
 25 so far and set  $P_R := P$ .

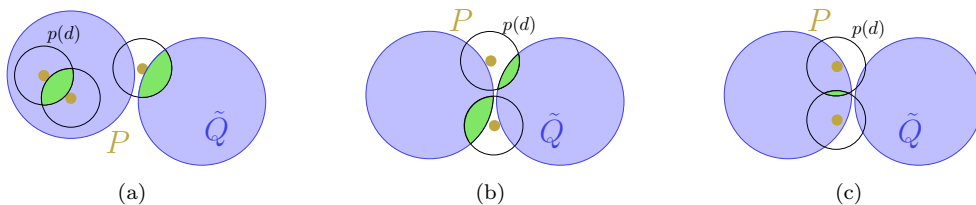


Fig. 10: (a) The green areas are the feasible regions of the discs in  $\tilde{Q}$ . A point  $q \in \tilde{q}$  may only be placed within its feasible region. The two points of  $P$  lying to the left have degree 1, the point lying to right has degree 2. (b,c) The points  $p$  have degree two. Both cases show a scenario which allows to match the points locally, i.e. only considering the set  $D$  and its two corresponding feasible regions. Case (b) shows one of the two possible matchings.

```

1 REMOVE-DEGREE-1-DISCS
1 forall  $\tilde{q}_i \in \tilde{Q}$  do
2   set  $F_i := \tilde{q}_i$ 
3   set  $C_i := \emptyset$ 
4 while there is some  $p \in P$  such that  $p(d)$  intersects only one  $F_i$  do
5   set  $F_i := F_i \cap p(d)$ 
6   if  $F_i = \emptyset$  then
7     return false
8   set  $C_i := C_i \cup \{p\}$ 
9 set  $P_R := P \setminus \bigcup_i C_i$ 
10 REMOVE-DEGREE-2-DISCS
3
4 In line 9 the remaining points  $P_R = P \setminus \bigcup_i C_i$  are points whose disc  $p(d)$  intersects exactly two
5 feasible regions. It is still possible to match points in  $P$  to points in  $\tilde{Q}$  by only analysing their
6 local environment. This is done by REMOVE-2-DISCS.
7 REMOVE-DEGREE-2-DISCS
1 foreach pair of feasible regions  $(F_i, F_j), i \neq j$  do
2   compute the set  $D$  of discs  $p(d)$  intersecting both  $F_i$  and  $F_j$ 
3   if
4      $D$  can be stabbed by one point of either  $F_i$  or  $F_j$  (but not by both of them) or
5      $D$  needs one point of  $F_i$  and one point of  $F_2$  to be stabbed
86  then
7   restrict  $F_i$  and  $F_j$  accordingly
8   if  $F_i = \emptyset \vee F_j = \emptyset$  then
9     return false
10  REMOVE-DEGREE-1-DISCS
11 BUILDGRAPH
9
10 Note that, all sets  $D$  of line 2 partition the set of the discs  $p(d)$  of the points in  $P_R$ . Line 7 restricts
11 the matching for points of degree 2 whose matching does not interfere with the matching of points
12 with other pairs of feasible regions. See Fig. 10(b) and 10(c) for an illustration of the two possible
13 scenarios allowing a local matching.
14 In line 11 all discs of each subset  $D$  can be stabbed by only one point of the two feasible regions  $F_i$ 
15 and  $F_j$  they intersect. Further, it holds that no two discs  $p(d)$  of different sets  $D$  can be stabbed
16 by only one point, because  $d < r(\sqrt{5 - 2\sqrt{3}} - 1)/2$ , see Fig. 9(b).
17 Now, it is possible to check for a valid point matching of the remaining points in  $P_R$  by computing
18 the maximum matching in a bipartite graph as follows: BUILDGRAPH builds a bipartite graph

```

1 on the feasible regions and the sets  $D$  of the partition of the discs  $p(d)$  of the points in  $P_R$ . For  
 2 each cell  $D$  of the partition there are two edges in the graph connecting  $D$  with the two feasible  
 3 regions that the discs in  $D$  intersect. We now compute a maximum matching on that graph. If  
 4 this matching connects all  $D$ -vertices with a feasible region, the predicate returns true and the  
 5 bound for the Hausdorff distance is smaller or equal than  $v$ . Otherwise the predicate returns false.

6 It is simple to return a matching which realises the Hausdorff distance which the predicate proved  
 7 to be realisable. Therefore, we first consider the feasible regions which are adjacent to a vertex  
 8 in the maximum matching. We place the point in such a feasible region such that it intersects all  
 9 discs in the adjacent set  $D$  of discs. For all other  $\tilde{q}_i \in \tilde{Q}$  we place their point  $q_i$  somewhere within  
 10 its feasible region  $F_i$ .

11 **Running time.** The algorithm consists of three phases: It first computes a polynomial set of  
 12 candidates which takes  $O(m^3 + m^2n)$  time. On this set we perform a binary search using the  
 13 predicate. The computation of the predicate is done by some recursive calls of the two sub-  
 14 algorithms REMOVE-DEGREE-1-DISCS and REMOVE-DEGREE-2-DISCS. These need to know the  
 15 intersections of the discs  $p(d)$  with the feasible regions, which are the discs in  $\tilde{Q}$  in the beginning.  
 16 We store these intersections distributed with every point  $p \in P$  and store references with each  
 17 feasible region to the  $p(d)$  it intersects. The initial set of the intersections can be computed using  
 18 a sweep-line in  $O(m+n)\log(m+n)$  time. The restrictions of the feasible regions can take at  
 19 most  $O(m)$  time. Further we maintain one point set containing points  $p \in P$  with degree 1 and a  
 20 second point set for points of degree 2. We move a point from the second to the first point set if its  
 21 degree is decreased. Thus, having the initial intersection set, all calls of REMOVE-DEGREE-1-DISCS  
 22 without line 10 take  $O(m)$  time.

23 The sub-algorithm REMOVE-DEGREE-2-DISCS needs to iterate over all pairs of feasible regions.  
 24 Instead of considering all possible pairs we only maintain a set of region pairs which indeed  
 25 intersect some discs  $p(d)$ . Because all  $D$ 's partition the points in  $P$  there are at most  $m$  discs  
 26 to consider in the stabbing analysis from line 1 to 6, thus REMOVE-DEGREE-2-DISCS needs  $O(m)$   
 27 time per call. Since it is called at most  $m$  times by REMOVE-DEGREE-1-DISCS its overall running  
 28 time is  $O(m^2)$ .

29 Finally we need to compute a maximum matching in the bipartite graph. Using the algorithm of  
 30 Hopcroft and Karp [8] this needs  $O(m\sqrt{m+n^2})$  time.

31 Putting all together we get a running time of  $O(m^3 + m^2n + ((m+n)\log(m+n) + m^2 +$   
 32  $m\sqrt{m+n^2})\log(m^3 + m^2n))$  which can be simplified to  $O(m^3 + m^2n + n\log^2 n)$ .

## 33 5.4 Algorithm GROWNDISCS

34 In this section, we present an approximation algorithm for precise  $P$  and imprecise  $\tilde{Q}$ . As a  
 35 subroutine in this algorithm, we assume that we are given an algorithm that computes a  $c$ -  
 36 approximation to the geometric  $k$ -centre problem (see Section 2), in time  $T(k, n)$ . We need this  
 37 because when we have  $k$  discs of  $\tilde{Q}$  which partially overlap, and there are  $n$  points of  $P$  in the  
 38 overlap, computing a lower bound on the Hausdorff distance for this subset is exactly solving the  
 39 geometric  $k$ -centre problem. Fig. 11(a) shows an example of the problem.

40 We first compute the set of possible values of the Hausdorff distance using the algorithm CAN-  
 41 DIDATES, followed by a binary search on the resulting candidate values in order to determine the  
 42 smallest value  $d$  for which the predicate described below evaluates to true. Therefore, we now first  
 43 describe a decision algorithm.

44 **Decision algorithm.** Let  $d$  be any given positive value. If there exists a solution to our problem  
 45 with distance at most  $d$ , the decision algorithm returns a solution with distance at most  $(c+2)d$ . If

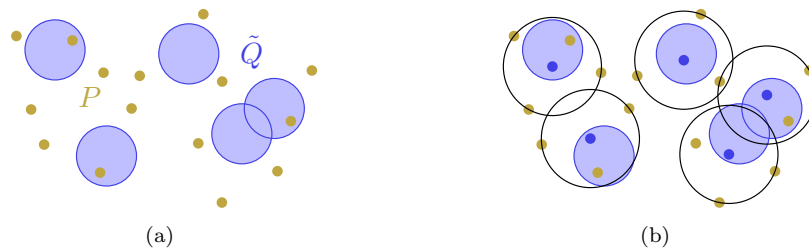


Fig. 11: (a) An example input, consisting of a set  $P$  of precise points and a set  $\tilde{Q}$  of imprecise points. (b) The optimal output. A set of circles of radius  $d$  is shown around the points in  $Q$ , which cover the points in  $P$ .

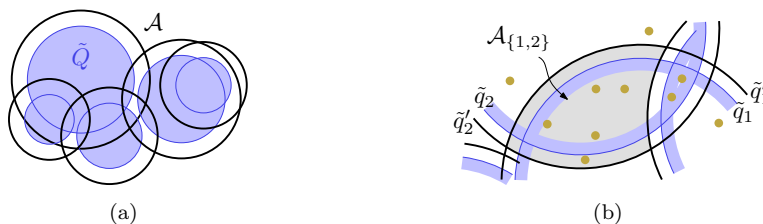


Fig. 12: (a) The black circles form the arrangement  $\mathcal{A}$  of the expanded discs  $\tilde{Q}'$ . (b) Each cell of the arrangement is determined by the indices of the discs it lies inside.

1 no solution of distance at most  $d$  exists, the algorithm either still returns a solution with distance  
2 at most  $(c+2)d$ , or it returns false.

3 Let  $\tilde{q}_1, \dots, \tilde{q}_n$  be the discs in  $\tilde{Q}$  where disc  $\tilde{q}_i$  has radius  $r_i$ . We define the *grown* disc  $\tilde{q}'_i$  to be the  
4 disc with the same centre point as  $\tilde{q}_i$ , but with radius  $r_i + d$ . We call the resulting set  $\tilde{Q}'$ .

5 **Observation 2:** If  $P$  is not covered by  $\tilde{Q}'$ , then there exists no solution of value  $d$ .

6 The correctness of Observation 2 is easy to see. Suppose the Hausdorff distance was smaller than  
7 or equal  $d$ , then every point of  $P$  would lie at most  $d$  away from a placement of a point  $\tilde{q}$  and thus  
8 lie within  $\tilde{Q}'$ .

9 So, we assume  $P$  is covered by  $\tilde{Q}'$ . We can test this easily, and immediately return false if this is  
10 not the case. Now, we compute the arrangement  $\mathcal{A}$  of  $\tilde{Q}'$ , i.e. the partition of the plane formed by  
11 the boundary of the expanded discs. This arrangement has a quadratic complexity in the worst  
12 case. Fig. 12(a) shows an example of the arrangement. If  $I \subseteq \{1, \dots, n\}$  is a certain set of indices,  
13 denote by  $\mathcal{A}_I$  the cell of the arrangement in the intersection of all discs  $\{\tilde{q}'_i \mid i \in I\}$ , but not inside  
14 any other disc. (Of course, most of these cells do not exist, since there is only a quadratic number  
15 of cells.) Each cell of this arrangement contains a subset of  $P$ ; we define  $P_I$  to be the set of points  
16 of  $P$  inside  $\mathcal{A}_I$ . Fig. 12(b) shows an example.

17 **Observation 3:** Let  $I$  be a set of indices. In the optimal solution  $Q \in \tilde{Q}$ , all the points of  $P_I$  are  
18 covered by circles of radius  $d$  around the points in  $\{q_i \mid i \in I\}$ .

19 **Proof:** Since the optimal solution has Hausdorff distance  $h(P, Q) \leq d$ , we know that each point  
20  $p \in P$  is covered by some circle of radius  $d$  around a point  $q \in Q$ . Now assume that  $p \in P_I$ . Then  
21 we know that  $|pq| \leq d$ , and  $q \in \tilde{q}$ , therefore  $p \in \tilde{q}'$ . So, by definition of  $\mathcal{A}_I$ ,  $q$  must be  $q_i$  for some  
22  $i \in I$ .  $\square$

23 This observation suggests we can solve the problem somehow separately in each cell of  $\mathcal{A}$ . For a  
24 given cell  $\mathcal{A}_I$ , the optimal solution uses  $k_I \leq |I|$  circles of radius  $d$  (centred around points of  $Q$ )

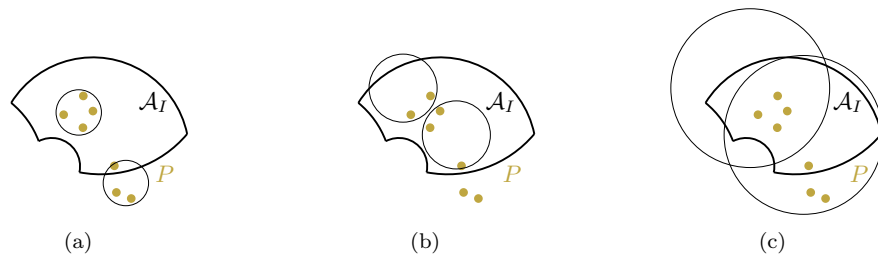


Fig. 13: (a) A cell  $\mathcal{A}_I$  of the arrangement, and a set of two circles of radius  $d$  covering  $P_I$  in the optimal solution. (b) A different set of at most two circles of radius  $cd$  covering  $P_I$ , as produced by the subroutine. (c) The enlarged circles with radius  $(c+2)d$  also cover all points outside  $\mathcal{A}_I$  that could be covered by the circles of the optimal solution.

1 to cover the points in  $P_I$ . Now, we could compute such a set of circles (most likely a different  
 2 set) by applying the  $c$ -approximation algorithm for geometric  $k$ -centre  $O(\log k_I)$  times in binary  
 3 search-like fashion on the value of  $k$ , until we find the smallest number  $k'_I$  for which the algorithm  
 4 returns a solution of radius smaller than  $cd$ . Note that the approximation guarantee implies that  
 5  $k'_I \leq k_I$ . This would provide us a set  $C_I$  of  $k'_I \leq k_I$  circles of radius  $cd$  that cover  $P_I$ . However,  
 6 there is a problem with this approach. The solutions are not independent: it is possible that a  
 7 certain circle of the optimal solution covers points from two different cells of the arrangement.  
 8 This means we may have constructed more than  $n$  circles.

9 So, what we do instead is this. We process the cells of the arrangement in any order. For the first  
 10 cell  $\mathcal{A}_I$ , we compute a set  $C_I$  of at most  $k_I$  circles of radius  $cd$  that cover  $P_I$ . Now, we grow our  
 11 circles until they have radius  $(c+2)d$ . This ensures that any points of  $P$  outside  $\mathcal{A}_I$  that were  
 12 covered by discs of the optimal solution that were covering any points of  $P_I$ , are now also covered  
 13 by  $C_I$ . Fig. 13 illustrates this.

14 A second complication comes from the fact that we required the centres of the circles to be in  
 15  $\tilde{Q}$ , not just in  $\tilde{Q}'$ . In order to ensure this, we simply move the circle centres to the closest point  
 16 in their discs, moving them by at most  $d$ . Since the circles are enlarged by  $2d$ , the moved and  
 17 enlarged circles will still cover all points of  $P$  that were covered by the original circles. Fig. 14  
 18 shows this case. Furthermore, this case does not interfere with the case described above, because  
 19 a circle cannot at the same time be close to the boundary of  $\mathcal{A}_I$  and far enough away from it not  
 20 to cover a point that is covered by a circle that also covers a point from a neighbouring cell.

21 For each next cell, we only consider those points that have not been covered yet, and otherwise  
 22 proceed in the same way.

23 This procedure results in a set  $C$  of at most  $n$  circles, composed of a set  $C_I$  for each cell  $\mathcal{A}_I$   
 24 of the arrangement. This set has the property that each  $C_I$  contains no more circles than the  
 25 corresponding set in the optimal solution. This implies that there exists a matching between  $C$   
 26 and  $\tilde{Q}'$  in the graph that has an edge between circle  $o$  and disc  $\tilde{q}'_i$  if  $c$  is in a set  $C_I$  where  $i \in I$ .  
 27 Clearly, this means that the centre of  $o$  is inside  $\tilde{q}'_i$ . Since an optimal matching exists, we can also  
 28 compute one efficiently (although it may be a different one).

29 For each value  $d$ , we spend  $O(n^2)$  time to compute  $\mathcal{A}$ , and  $\log k_I \cdot T(k_I, |P_I|)$  time per cell to  
 30 solve the geometric  $k$ -centre problem. If  $k$  is the largest value of  $k_I$  over all  $I$ , then a crude upper  
 31 bound for this is  $n^2 T(k, m)$ . As seen in the previous section, a matching can be computed in  
 32  $O(mn + m\sqrt{m})$  time [8]. So, we spend  $O(n^2 \log k \cdot T(k, m) + mn + m\sqrt{m})$  time in total on the  
 33 decision algorithm.

34 We can summarise:

35 **Lemma 3:** Let a value  $d$  and an algorithm that can compute a  $c$ -approximation to the geometric



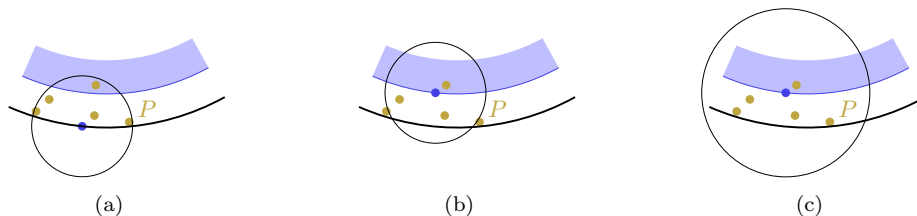


Fig. 14: (a) A circle of radius  $cd$  covers a number of points of  $P_I$  inside a certain cell  $\mathcal{A}_I$  of the arrangement. The centre  $q$  of the circle lies inside  $\mathcal{A}_I$ , but not inside the region  $\tilde{q}$ . (b) The point  $q$  has been moved into  $\tilde{q}$ , but now some points of  $P_I$  that were covered are no longer covered. (c) The enlarged circle of radius  $(c+2)d$  covers the points again.

1  $k$ -centre problem in time  $T(k, n)$  be given. We can compute, in  $O(n^2 \log k \cdot T(k, m) + mn + m\sqrt{m})$   
 2 time, either a solution with distance  $d'$  that is at most  $(c+2)$  times larger than  $d$ , or decide that  
 3 no solution of distance  $d$  exists.

4 **Optimisation algorithm.** We now describe how to use the decision algorithm to obtain an opti-  
 5 misation algorithm. First, we compute the set of possible values of the Hausdorff distance using  
 6 algorithm CANDIDATES. Then we do a binary search on the value of  $d$ , by picking the middle of  
 7 the remaining candidates and recursing up if the decision algorithm returns “no” or down if it  
 8 returns a solution. This procedure will result in two consecutive candidate values  $d_1$  and  $d_2$ , such  
 9 that  $d_1$  returns no solution but  $d_2$  does. Let  $d'$  be the solution returned by the decision algorithm  
 10 run on  $d_2$ , and let  $d^*$  be the optimal solution. Then we know that  $d' \leq (c+2)d_2$ . We also know  
 11 that  $d^* > d_1$ , and since the next candidate is  $d_2$ , this means  $d^* \geq d_2$ . Therefore,  $d' \leq (c+2)d^*$ .  
 12 Since obviously also  $d^* \leq d'$ , we conclude that  $d'$  is a  $(c+2)$  approximation of  $d^*$ .

13 As for the running time, we first execute CANDIDATES in  $O(m^3 + m^2n)$  time. This results in a  
 14 set of  $O(m^3 + m^2n)$  candidates; doing a binary search on them we execute the decision algorithm  
 15 described above  $O(\log(m+n))$  times, for a total of  $O((n^2 \log k \cdot T(k, m) + mn + m\sqrt{m}) \log m + n)$   
 16 time. Some terms are dominated by the computation of the algorithms, which results in the  
 17 following theorem:

18 **Theorem 7:** Let  $P$  denote a precise point set consisting of  $m$  points and  $\tilde{Q}$  denote an imprecise  
 19 point set consisting of  $n$  discs. Given a  $c$ -approximation to the geometric  $k$ -covering problem that  
 20 runs in  $T(k, m)$  time, we can compute a  $(c+2)$ -approximation to the tight lower bound of  $h(P, \tilde{Q})$   
 21 in  $O(m^3 + m^2n + mn \log(m+n) + n^2 \log(m+n) \log k \cdot T(k, m))$  time, where  $k \leq n$  is an internal  
 22 parameter of the optimal solution.

## 23 5.5 Algorithm CENTREPOINTS

24 A trivial algorithm is to place all points in  $Q$  in the centres of their discs. This algorithm is a  
 25  $c$ -approximation if the smallest possible Hausdorff distance is at least  $r_{\max}/(c-1)$ , where  $r_{\max}$   
 26 denotes the radius of the largest disc in  $\tilde{Q}$ .

27 **Lemma 4:** Let  $P$  denote a precise point set consisting of  $m$  points and  $\tilde{Q}$  denote an imprecise point  
 28 set consisting of  $n$  discs. We can compute a  $c$ -approximation to the tight lower bound of  $h(P, \tilde{Q})$   
 29 if the Hausdorff distance is at least  $r_{\max}/(c-1)$ , where  $r_{\max}$  denotes the radius of the largest disc  
 30 in  $\tilde{Q}$ .



Fig. 15: (a) Four disjoint discs of radius  $r$  of  $\tilde{Q}$  can be laid out such that the arrangement of enlarged circles of radius  $\frac{1}{2}r$  have a common intersection. (b) With five discs, this is not possible anymore.

## 1 5.6 Putting the algorithms together

2 When  $P$  is precise and  $\tilde{Q}$  is imprecise, we note that by Theorem 7 Algorithm GROWNDISCS  
 3 immediately presents a 4-approximation for the case when the discs may have different radii and  
 4 overlap, which we obtain by plugging in a 2-approximation algorithm for geometric  $k$ -covering  
 5 that runs in  $O(m \log k)$  time [4]. (Note that although the authors also present a solution for  
 6 approximating the value of  $k$  for a fixed radius that achieves the much better approximation  
 7 factor of  $(1 + \varepsilon)$ , we cannot use that solution since we require a guarantee on the radius, not the  
 8 number of disks.)

9 Corollary 2: Let  $P$  denote a precise point set consisting of  $m$  points and  $\tilde{Q}$  denote an imprecise  
 10 point set consisting of  $n$  discs. We can compute a 4-approximation to the tight lower bound of  
 11  $h(P, \tilde{Q})$  in  $O(m^3 + m^2n + mn^2 \log(m+n) \log^2 n)$  time in the worst case.

12 We can improve this algorithm by first testing whether the tight lower bound is smaller than  
 13  $r_{\min}(\sqrt{5} - 2\sqrt{3} - 1)/2$  where  $r_{\min}$  denotes the radius of the smallest disc in  $\tilde{Q}$  using Algorithm  
 14 INDEPENDENTSETS, without increasing the asymptotic running time. If it is, then we can actually  
 15 compute the exact solution.

16 Furthermore, when the discs are disjoint and all have the same size, we can improve this result to  
 17 a 3-approximation by combining the algorithms GROWNDISCS and CENTREPOINTS:

18 Theorem 8: Let  $P$  denote a precise point set consisting of  $m$  points and  $\tilde{Q}$  denote an imprecise  
 19 point set consisting of  $n$  *disjoint* discs of the same radius. The tight lower bound for  $h(P, \tilde{Q})$  is  
 20 3-approximable in time  $O(m^3 + m^2n + n \log^2 n)$ .

21 First we test whether the lower bound on the Hausdorff distance is at least  $r/2$  by applying  
 22 CENTREPOINTS and checking whether the resulting distance is at least  $3/2r$ . If it is, we are done.  
 23 Otherwise, note that each cell of  $\mathcal{A}$  is a subset of the intersection of  $k \leq 4$  discs, because  $\tilde{Q}$ 's  
 24 discs are disjoint and the Hausdorff distance is less than  $r/2$ , see Fig. 15(a) and 15(b). Therefore,  
 25 by Theorem 7 we can obtain a 3-approximation from Algorithm GROWNDISCS by plugging in an  
 26 exact algorithm to solve the geometric 4-covering problem.

27 We can solve the geometric 4-covering problem exactly by computing the arrangement of discs of  
 28 radius  $d$  around the points to be covered. The arrangement has quadratic complexity. We need  
 29 to find out whether there is a set of four cells such that every disc of the arrangement contains  
 30 at least one of the cells. There are  $\binom{O(m^2)}{4} \in O(m^8)$  such combinations to test, and by keeping  
 31 track of which discs are already taken care of each can be tested in constant time. So, using this  
 32 algorithm, we have a  $1 + 2 = 3$ -approximation to the original problem for disjoint unit discs. The  
 33 total running time now becomes  $O(n^2 m^8 \log(m+n))$ .

## 6 Conclusions and Future Work

We study computing tight lower and upper bounds on the directed Hausdorff distance between two point sets, when at least one of the sets has imprecision. We give 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 do not have any constructive results for the lower bound.

All our results hold for the directed Hausdorff distance. A 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 maximum of the two directed distances. However, computing lower bounds seems to be more complicated, because 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 underlying metrics other than the Euclidean metric, similarity measures other than the Hausdorff distance, or, as is common in shape matching, allowing some transformation of the point sets.

## Acknowledgments

This research was initiated during the Computational Geometry Workshop on Imprecise Data, which was held in December 2008 in Sydney and partly supported by NICTA. The authors thank the other participants of the workshop for helpful discussions and for providing a stimulating working environment, and also the anonymous reviewers of an earlier version of this paper for their suggestions and comments. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. M. Löffler was supported by the Netherlands Organisation for Scientific Research (NWO) under the project GOGO, and by the U.S. Office of Naval Research under grant N00014-08-1-1015. C. Knauer and M. Scherfenberg were supported by the German Research Foundation (DFG), grant AL 253/5-2.

## References

- [1] H.-K. Ahn, C. Knauer, M. Scherfenberg, L. Schlipf, and A. Vigneron. Computing the discrete Fréchet distance with imprecise input. In O. Cheong, K.-Y. Chwa, and K. Park, editors, *Algorithms and Computation*, volume 6507 of *Lecture Notes in Computer Science*, pages 422–433. Springer Berlin / Heidelberg, 2010.
- [2] H. Alt, B. Behrends, and J. Blömer. Approximate matching of polygonal shapes. *Annals of Mathematics and Artificial Intelligence*, 13(3-4):251–265, 1995.
- [3] H. Alt and L. Guibas. *Handbook on Computational Geometry*, chapter Discrete Geometric Shapes: Matching, Interpolation, and Approximation - A Survey, pages 251–265. 1995.
- [4] T. Feder and D. Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 434–444, New York, NY, USA, 1988. ACM.
- [5] S. J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.
- [6] M. T. Goodrich and J. Snoeyink. Stabbing parallel segments with a convex polygon. *Computer Vision, Graphics, and Image Processing*, 49(2):152–170, 1990.
- [7] L. J. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: building robust algorithms from imprecise computations. In *Proceedings of the fifth Annual Symposium on Computational Geometry*, SCG '89, pages 208–217, New York, NY, USA, 1989. ACM.

- 
- 1 [8] J. E. Hopcroft and R. M. Karp. An  $n^{3/2}$  algorithm for maximum matching in bipartite graphs. *SIAM*  
2 *Journal on Computing*, 2(4):225–231, 1973.
- 3 [9] D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982.
- 4 [10] M. Löffler and M. van Kreveld. Largest bounding box, smallest diameter, and related problems on  
5 imprecise points. *Computational Geometry: Theory and Applications*, 43:419–433, 2010.
- 6 [11] A. Mukhopadhyay, E. Greene, and S. V. Rao. On intersecting a set of isothetic line segments with  
7 a convex polygon of minimum area. *International Journal of Computational Geometry and Applica-*  
8 *tions*, 19(6):557–577, 2009.
- 9 [12] A. Mukhopadhyay, C. Kumar, E. Greene, and B. Bhattacharya. On intersecting a set of parallel  
10 line segments with a convex polygon of minimum area. *Information Processing Letters*, 105(2):58–64,  
11 2008.
- 12 [13] T. Nagai and N. Tokura. Tight error bounds of geometric problems on convex objects with imprecise  
13 coordinates. In *Revised Papers from the Japanese Conference on Discrete and Computational*  
14 *Geometry*, JCDCG '00, pages 252–263, London, UK, 2001. Springer-Verlag.