

# Hausdorff Distance under Translation for Points, Disks, and Balls\*

Pankaj K. Agarwal<sup>†</sup>   Sariel Har-Peled<sup>‡</sup>   Micha Sharir<sup>§</sup>   Yusu Wang<sup>¶</sup>

July 9, 2002

## Abstract

Let  $\mathcal{A}$  and  $\mathcal{B}$  be two sets of balls in  $\mathbb{R}^d$ ,  $d = 2, 3$ . We measure similarity between  $\mathcal{A}$  and  $\mathcal{B}$  by computing the minimum Hausdorff distance between  $\mathcal{A} + t$  and  $\mathcal{B}$ , where the minimum is taken either over all vectors  $t \in \mathbb{R}^d$  or over the vectors  $t$  such that  $\mathcal{A} + t$  and  $\mathcal{B}$  do not intersect. These problems arise in measuring similarity between the shapes of two proteins. We propose a number of exact and approximation algorithms for these problems. Since Hausdorff distance is sensitive to out-liers, we also propose efficient approximation algorithms for computing the minimum root-mean-square (rms) Hausdorff distance, under translation, between two point sets.

## 1 Introduction

The problem of shape matching in two and three dimensions arises in a variety of applications, including computer graphics, computer vision, pattern recognition, computer aided design, and molecular biology [AG99, HMWN02, SL02]. For example, proteins with similar shapes are likely to have similar functionalities, therefore classifying proteins (or their fragments) based on their shapes is important. Similarly, the proclivity of two proteins binding with each other also depends on their shapes, so shape matching is central to the so-called *docking* problem in molecular biology [HMWN02].

---

\*P.A. is supported by by NSF grants ITR-333-1050, EIA-9870724, EIA-997287, and CCR-02-04118, and by a grant from the U.S.-Israeli Binational Science Foundation. S.H. is supported by NSF CAREER award CR-0132901. M.S. is supported by NSF Grants CCR-97-32101 and CCR-00-98246, by a grant from the Israel Science Fund (for a Center of Excellence in Geometric Computing), by the Hermann Minkowski-MINERVA Center for Geometry at Tel Aviv University, and by a grant from the U.S.-Israeli Binational Science Foundation. Y.W. is supported by NSF grants ITR-333-1050 and CCR-02-04118.

<sup>†</sup>Department of Computer Science, Duke University, Durham, NC 27708-0129, U.S.A. Email: [pankaj@cs.duke.edu](mailto:pankaj@cs.duke.edu).

<sup>‡</sup>Department of Computer Science, DCL 2111; University of Illinois; 1304 West Springfield Ave.; Urbana, IL 61801; USA. Email: [sariel@cs.uiuc.edu](mailto:sariel@cs.uiuc.edu).

<sup>§</sup>School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel; and Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA. Email: [sharir@math.tau.ac.il](mailto:sharir@math.tau.ac.il).

<sup>¶</sup>Department of Computer Science, Duke University, Durham, NC 27708-0129, U.S.A. Email: [wys@cs.duke.edu](mailto:wys@cs.duke.edu).

Informally, the shape-matching problem can be described as follows: given a distance measure between two sets of objects in  $\mathbb{R}^2$  and  $\mathbb{R}^3$ , determine a transformation that minimizes the distance between them. Although in many applications the “best” transformation can be any rigid motion, there are constraints on transformations in some applications. For example, when matching the pieces of a puzzle, it is important that no two pieces overlap each other. Another example is the aforementioned docking problem, where two molecules bind together to form a compound. Strong complementarity exists at the interface of the docking, while two proteins remain separate due to inter-atomic forces [HMWN02, NLWN95]. See Figure 1 for an example.

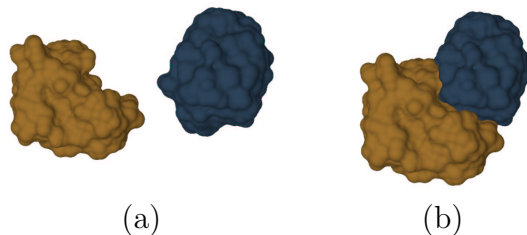


Figure 1: (a) Two proteins. (b) Binding of the proteins in (a); one protein complements the other.

Several distance measures, depending on the input objects and applications, have been proposed. One widely studied distance measure is the Hausdorff distance measure [AG99], originally proposed for point sets. In this paper we study shape matching, with or without constraints, for sets of points and balls, under Hausdorff distance and its variants. Although we are primarily interested in balls because of molecular-biology applications, for a molecule is typically modeled as a set of balls (each modeling an atom of the molecule), any two or three-dimensional shape can be approximated by a finite union of balls [AK00, AK01].

**Problem statement.** Let  $A$  and  $B$  be two (possibly infinite) sets of geometric objects (e.g., points, balls, simplices) in  $\mathbb{R}^d$ , and let  $d : A \times B \rightarrow \mathbb{R}$  be a distance function between the objects in  $A$  and  $B$ . For  $a \in A$ , we define  $\delta(a, B) = \inf_{b \in B} d(a, b)$ . Similarly, we can define  $\delta(A, b)$ , for  $b \in B$ . The *directional Hausdorff distance* between  $A$  and  $B$  is defined as

$$h(A, B) = \sup_{a \in A} \delta(a, B)$$

and the *Hausdorff distance* between  $A$  and  $B$  is defined as

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

In order to measure similarity between  $A$  and  $B$ , we compute the minimum value of Hausdorff distance over all translates of  $B$  within a given set  $T$ . That is, let  $T \subseteq \mathbb{R}^d$  be a set of translation vectors. We define

$$\sigma(A, B; T) = \inf_{t \in T} H(A + t, B).$$

In our applications,  $T$  will either be the entire  $\mathbb{R}^d$  or the set of *collision-free* placements of  $B$  at which it does not intersect any ball of  $A$ . As mentioned above, the collision-free matching between objects is useful for applications in which the goal is to locate a transformation, so that the shape of one object best complements that of the other. We will use  $\sigma(A, B)$  to denote  $\sigma(A, B; \mathbb{R}^d)$ .

Our definition of (directional) Hausdorff distance is slightly different than the one typically used in the literature [AG99], where one considers the set of points lying in the union of objects in  $A$  and  $B$ , respectively. That is, let  $U_A$  (resp.  $U_B$ ) be the union of the objects in  $A$  (resp.  $B$ ). Then they compute  $H(U_A, U_B)$  and  $\sigma(U_A, U_B)$ . Abusing the notation a little, we will use  $h_U(A, B)$  to denote  $h(U_A, U_B)$ . Similarly, define  $H_U(A, B)$  and  $\sigma_U(A, B; T)$ .

A drawback of directional Hausdorff distance (and thus of Hausdorff distance) is its sensitivity to outliers in  $A$ . One possible approach to circumvent this problem is to use “partial matching” [CDEK99, HKR93], but then one has to determine how many of the objects in  $A$  should be matched to  $B$ . Another possible approach is to use the root-mean-square (rms, for brevity) Hausdorff distance between  $A$  and  $B$ , namely, define

$$\tilde{h}(A, B) = \frac{\int_A \delta^2(a, B) dA}{\int_A dA}$$

and

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

Finally, define  $\tilde{\sigma}(A, B; T) = \inf_{t \in T} \tilde{\sigma}(A + t, B)$ .

In this paper we develop algorithms for computing  $\sigma_U(A, B; T)$  and  $\sigma(A, B; T)$  for balls, and for computing  $\tilde{\sigma}(A, B; T)$  for sets of points.

**Previous results.** It is beyond the scope of this paper to discuss all the results on shape matching. We refer the reader to [BJ85, CD86, HMWN02, SL02] and references therein for a sample of known results. Here we summarize the known results on shape matching using the Hausdorff distance measure.

Most of the early work on computing Hausdorff distance focused on point sets. Let  $\mathcal{A}$  and  $\mathcal{B}$  be two families of  $m$  and  $n$  points, respectively, in  $\mathbb{R}^d$ .  $H(\mathcal{A}, \mathcal{B})$  in  $\mathbb{R}^2$  can be computed in time  $O((n + m) \log nm)$  using Voronoi diagram [ABB95], and in  $\mathbb{R}^3$ , it can be computed in time  $O(n^{4/3+\varepsilon})$  using the data structure by Agarwal and Matoušek [AM93]. Huttenlocher *et al.* [HKS93] showed that  $\sigma(\mathcal{A}, \mathcal{B})$  can be computed in time  $O(nm(n + m)\alpha(nm) \log(n + m))$  in  $\mathbb{R}^2$  and in  $O((nm)^2(n + m)^{1+\varepsilon})$  time in  $\mathbb{R}^3$ , where  $\varepsilon > 0$  is arbitrarily small. For higher dimensions, Chew *et al.* [CDEK99] presented an algorithm for computing  $\sigma(\mathcal{A}, \mathcal{B})$  with running time  $O(n^{\lceil 3d/2 \rceil + 1} \log^3 n)$  in  $\mathbb{R}^d$ . The minimum Hausdorff distance between  $\mathcal{A}$  and  $\mathcal{B}$  under rigid motion transformation in  $\mathbb{R}^2$  can be computed in  $O((m + n)^6 \log nm)$  time [HKK92].

Faster approximation algorithms for computing  $\sigma(\mathcal{A}, \mathcal{B})$  were first proposed by Goodrich *et al.* [GMO94]. Aichholzer *et al.* [AAR97] proposed a framework of approximation algorithm by the so-called reference points. See [CS98, IMV99, IV00] for other approximation algorithms. The problem of partial matching problem under Hausdorff distance was studied in [CS98, CDEK99, HKR93, IMV99]. Indyk *et al.* [IMV99] approximated the partial matching under rigid motions in time  $O(\Delta n^2)$  in  $\mathbb{R}^2$  and  $O(\Delta^3 n^2)$  in  $\mathbb{R}^3$ , where  $\Delta$  is the maximum of

the diameters of the two point sets. Actually, their algorithm can be extended to minimize the rms Hausdorff distance.

Algorithms for computing  $H_U(\mathcal{A}, \mathcal{B})$  and  $\sigma_U(\mathcal{A}, \mathcal{B})$  when  $\mathcal{A}$  and  $\mathcal{B}$  are sets of  $m$  and  $n$  segments in the plane were developed in [AST94, ABB95, Ata83]. Agarwal *et al.* [AST94] showed that  $\sigma_U(\mathcal{A}, \mathcal{B})$  can be computed in time  $O((mn)^2 \log^3(mn))$ . If we also allow rotations to minimize the Hausdorff distance, Chew *et al.* [CGH<sup>+</sup>97] developed an  $O((mn)^3 \log^2(mn))$ -time algorithm. Very little is known for computing  $\sigma(\mathcal{A}, \mathcal{B})$  in higher dimensions, where  $\mathcal{A}$  and  $\mathcal{B}$  are simplices or other geometric shapes.

**Our results.** This paper consists of three parts. The first two parts consider Hausdorff distance for balls, and the third part considers rms Hausdorff distance for point sets. Let  $D(c, r)$  denote the ball in  $\mathbb{R}^d$  of radius  $r$  centered at  $c$ . Let  $\mathcal{A} = \{A_1, \dots, A_m\}$  and  $\mathcal{B} = \{B_1, \dots, B_n\}$  be two families of balls in  $\mathbb{R}^d$ , where  $A_i = D(a_i, \rho_i)$  and  $B_j = D(b_j, r_j)$ . Let  $\mathcal{F}$  be the set of all translation vectors  $t$  so that  $\mathcal{A} + t$  does not intersect any ball of  $\mathcal{B}$ .

Section 2 considers the problem of Hausdorff distance between sets  $\mathcal{A}$  and  $\mathcal{B}$ . For  $A_i \in \mathcal{A}$  and  $B_j \in \mathcal{B}$ , let  $d(A_i, B_j) = d(a_i, b_j) - \rho_i - r_j$ , i.e., we regard  $A_i$  and  $B_j$  as weighted points while measuring the distance. In Section 2 we describe an  $O(mn(m+n) \log^3 mn)$ -time algorithm for computing  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in  $\mathbb{R}^2$  and an  $O(m^2 n^2 (m+n) \log^3 mn)$ -time algorithm in  $\mathbb{R}^3$ . Our algorithm can also compute  $\sigma(\mathcal{A}, \mathcal{B})$ . We use this function for computing distances between two balls because measuring similarities between molecules using the weighted distance between the centers of their atoms is common in molecular biology [HMWN02]. Furthermore, it also enables us to develop an algorithm for the (collision-free) partial-matching problem under Hausdorff distance, which we show can be solved in the same asymptotic time complexity as computing  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$ .

Section 3 considers the problem of computing  $\sigma_U(\mathcal{A}, \mathcal{B})$  and  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$ , i.e., it computes the Hausdorff distance between the points lying in the union of  $\mathcal{A}$  and the union of  $\mathcal{B}$ , minimized over all translates of  $\mathcal{A}$  in  $\mathbb{R}^d$  or in  $\mathcal{F}$ . We first describe an  $O(mn(m+n) \log^3 mn)$ -time algorithm for computing  $\sigma_U(\mathcal{A}, \mathcal{B})$  in  $\mathbb{R}^2$ , which relies on a number of geometric properties of the union of disks. The same approach can be extended to compute  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$  within the same asymptotic time complexity. A straightforward extension of our algorithm to  $\mathbb{R}^3$  is rather inefficient, so we propose approximation algorithms for  $\sigma_U(\mathcal{A}, \mathcal{B})$  and  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$ . In particular, given a parameter  $\varepsilon > 0$ , we compute a translation  $t$  in  $O(\frac{(m+n)}{\varepsilon^2} \log^3(m+n))$  time in  $\mathbb{R}^2$  and in  $O(\frac{(m^2+n^2)}{\varepsilon^3} \log^2(m+n))$  time in  $\mathbb{R}^3$  such that  $H_U(\mathcal{A} + t, \mathcal{B}) \leq (1 + \varepsilon)\sigma_U(\mathcal{A}, \mathcal{B})$ . We also present a pseudo-approximation algorithm for computing  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$ . For a given parameter  $\varepsilon > 0$ , the algorithm computes a region  $X$ , an  $\varepsilon$ -approximation of  $\mathcal{F}$  (which will be defined formally in Section 3). It then returns a placement  $t \in X$  such that  $H_U(\mathcal{A} + t, \mathcal{B}; X) \leq (1 + \varepsilon)\sigma_U(\mathcal{A}, \mathcal{B}; X)$ , in time  $O(\frac{(m^2+n^2)}{\varepsilon^3} \log^2 mn)$  in  $\mathbb{R}^3$ . This variant of approximation makes sense in applications where the data is noisy and shallow penetrations between objects are allowed, as is the case in the docking problem [HMWN02].

Finally, let  $\mathcal{A}$  and  $\mathcal{B}$  be two sets of points in  $\mathbb{R}^d$ . Section 4 describes an  $O((mn/\varepsilon^d) \log(mn/\varepsilon))$ -time algorithm for computing  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ . Although the running time of our algorithm is only slightly better than that of Indyk *et al.* [IMV99], our algorithm is simpler and more direct. In fact, we solve a more general problem, which is interesting in its own right. Given a family  $P_1, \dots, P_l$  of point sets in  $\mathbb{R}^d$ , with a total of  $N$  points, we construct a decomposition of  $\mathbb{R}^d$

into  $O(N/\varepsilon^d)$  cells, which is an  $\varepsilon$ -approximation of the Voronoi diagrams of each  $P_i$ , in the sense defined in [AM02]. Given a semigroup operation  $+$ , we can preprocess this decomposition in  $O((N/\varepsilon^d) \log(N/\varepsilon))$  time so that for a query point  $q$ , an  $(1 + \varepsilon)$ -approximation of  $\sum_{i=1}^l \delta^2(q, P_i)$  can be computed in  $O(\log(N/\varepsilon))$  time.

## 2 Collision-Free Hausdorff Distance between Sets of Balls

Let  $\mathcal{A} = \{A_1, \dots, A_m\}$  and  $\mathcal{B} = \{B_1, \dots, B_n\}$  be two sets of balls in  $\mathbb{R}^d$ ,  $d = 2, 3$ . For  $A_i \in \mathcal{A}$  and  $B_j \in \mathcal{B}$ , we define  $d(A_i, B_j) = d(a_i, b_j) - \rho_i - r_j$ . Let  $\mathcal{F}$  be the set of placements  $t$  of  $\mathcal{A}$  such that  $\mathcal{A} + t$  does not intersect any ball of  $\mathcal{B}$ . In this section, we describe an exact algorithm for computing  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$ , and then show that it can be extended to partial matching.

**Computing  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$ .** As now common in geometric optimization, we first present an algorithm for the decision problem, namely, given a parameter  $\delta > 0$ , determine whether  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F}) \leq \delta$ . We then use the parametric-searching technique [AST94, Meg83] to compute  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$ . Given  $\delta \geq 0$ , for  $1 \leq i \leq m$ , let  $V_i \subseteq \mathbb{R}^d$  be the set of vectors  $t \in \mathbb{R}^d$  such that

(V1)  $A_i + t$  does not intersect the interior of any ball of  $\mathcal{B}$ ;

(V2)  $\min_{1 \leq j \leq n} d(A_i + t, B_j) \leq \delta$ .

Let  $D_{ij}^- = D(b_j - a_i, \rho_i + r_j)$ , and  $D_{ij}^+ = D(b_j - a_i, \rho_i + r_j + \delta)$ . Then  $U_i^+ = \bigcup_{j \leq n} D_{ij}^+$  is the set of vectors that satisfy (V2), and the interior of  $U_i^- = \bigcup_{j \leq n} D_{ij}^-$  violates (V1). It is obvious that  $V_i = cl(U_i^+ \setminus U_i^-)$ . See Figure 2 for an illustration. Let

$$V(\mathcal{A}, \mathcal{B}) = \bigcap_{1 \leq i \leq m} V_i = cl \left( \left( \bigcap_i U_i^+ \right) \setminus \left( \bigcup_i U_i^- \right) \right).$$

By definition,  $V(\mathcal{A}, \mathcal{B}) \subseteq \mathcal{F}$  and for all  $t \in V(\mathcal{A}, \mathcal{B})$ ,  $h(\mathcal{A} + t, \mathcal{B}) \leq \delta$ . Similarly, we define  $V(\mathcal{B}, \mathcal{A}) \subseteq \mathcal{F}$  to be the set of all vectors  $t$  such that  $h(\mathcal{B}, \mathcal{A} + t) \leq \delta$ . Obviously  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F}) \leq \delta$  if and only if  $V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A}) \neq \emptyset$ .

**Lemma 2.1** *The combinatorial complexity of  $V(\mathcal{A}, \mathcal{B})$  is  $O(m^2n)$  in  $\mathbb{R}^2$ .*

*Proof:* If an edge of  $\partial V(\mathcal{A}, \mathcal{B})$  is not adjacent to any vertex, then it is the entire circle bounding a disk of  $D_{ij}^+$  or  $D_{ij}^-$ . There are  $O(mn)$  such disks, so it suffices to bound the number of vertices in  $V(\mathcal{A}, \mathcal{B})$ .

Let  $v$  be a vertex of  $V(\mathcal{A}, \mathcal{B})$ ;  $v$  is either a vertex of  $V_i$ , for some  $1 \leq i \leq m$ , or an intersection point of an edge in  $V_i$  and an edge in  $V_k$ . In the latter case,  $v \in V_i \cap V_k$ , for  $1 \leq i < k \leq m$ , i.e.,  $v \in (U_i^+ \cap U_k^+) \setminus (U_i^- \cup U_k^-)$ . In other words,  $v$  is a vertex of  $U_i^+ \cap U_k^+$ ,  $U_i^+ \setminus U_k^-$ , or  $U_i^- \cup U_k^-$ , for  $1 \leq i, k \leq m$ . It can be checked that a vertex of  $U_i^+ \cap U_k^+$  (resp.  $U_i^+ \setminus U_k^-$ ) that lies on both  $\partial U_i^+$  and  $\partial U_k^+$  (resp.  $\partial U_k^-$ ) is also a vertex of  $U_i^+ \cup U_k^+$  (resp.  $U_i^+ \cup U_k^-$ ). Therefore, every vertex in  $V(\mathcal{A}, \mathcal{B})$  is a vertex of  $U_i^+ \cup U_k^+$ ,  $U_i^+ \cup U_k^-$ , or  $U_i^- \cup U_k^-$ ,

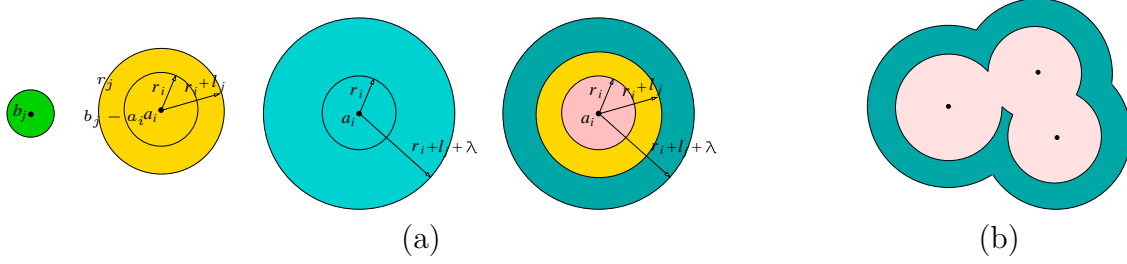


Figure 2: (a) Inner, middle and outer disks are  $B_j - a_i$ ,  $D_{ij}^-$ , and  $D_{ij}^+$ , respectively; (b) an example of  $V_i$  (dark region), which is the difference between  $U_i^+$  (the whole union) and  $U_i^-$  (inner light region).

for some  $1 \leq i \leq k \leq m$ . Since each  $U_i^+$  is the union of a set of  $n$  disks, each of  $U_i^- \cup U_k^+$ ,  $U_i^+ \cup U_k^-$ ,  $U_i^- \cup U_k^-$  is the union of a set of  $2n$  disks and thus has  $O(n)$  vertices [KLPS86]. Hence,  $V(\mathcal{A}, \mathcal{B})$  has  $O(m^2n)$  vertices. ■

**Remark:** The above argument not only bounds the complexity of  $V(\mathcal{A}, \mathcal{B}) = \bigcap_{i=1}^m V_i$ , but also bounds the complexity of the arrangement of  $\mathcal{V} = \{V_1, V_2, \dots, V_m\}$ . Indeed, any intersection point of  $\partial V_i$  and  $\partial V_k$  lies on the boundary of  $\partial(V_i \cap V_k)$ , and we have argued that  $V_i \cap V_k$  has  $O(n)$  vertices. Hence, the entire arrangement has  $O(m^2n)$  vertices.

Similarly, we can prove that  $V(\mathcal{B}, \mathcal{A})$  has  $O(n^2m)$  vertices. Following the same argument as in the proof of the above lemma, we can show the following:

**Corollary 2.2** *The combinatorial complexity of  $V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A})$  is  $O(mn(n + m))$ .*

We can compute  $V(\mathcal{A}, \mathcal{B})$ ,  $V(\mathcal{B}, \mathcal{A})$ , and their intersection in time  $O((m + n)nm \log nm)$ , using a divide-and-conquer approach combined with a sweepline approach. We omit the straightforward details. Finally, as in [AST94], using parametric search technique, we can compute  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in  $O(mn(m + n) \log^3 mn)$  time.

Next, we prove the complexity of  $V(\mathcal{A}, \mathcal{B})$  in  $\mathbb{R}^3$ .

**Lemma 2.3** *The combinatorial complexity of  $V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A})$  in  $\mathbb{R}^3$  is  $O(n^2m^2(m + n))$ .*

*Proof:* It suffices to prove that the combinatorial complexity of  $V(\mathcal{A}, \mathcal{B})$  is  $O(m^3n^2)$ . The number of faces or edges of  $V(\mathcal{A}, \mathcal{B})$  that do not contain any vertex is  $O(n^2m^2)$  since they are defined by at most two balls in a family of  $2mn$  balls. We therefore focus on the number of vertices in  $V(\mathcal{A}, \mathcal{B})$ . As in the proof of Lemma 2.1, we can argue that every vertex of  $V(\mathcal{A}, \mathcal{B})$  is a vertex of  $X_i \cup X_j \cup X_k$ , where  $X_i$  (or  $X_j, X_k$ ) is  $U_i^+$  or  $U_i^-$ . Since the union of  $r$  balls in  $\mathbb{R}^3$  has  $O(r^2)$  vertices,  $X_i \cup X_j \cup X_k$  has  $O(n^2)$  vertices, thereby implying that  $V(\mathcal{A}, \mathcal{B})$  has  $O(m^3n^2)$  vertices. Similarly,  $V(\mathcal{B}, \mathcal{A})$  has  $O(n^3m^2)$  vertices and  $V(\mathcal{B}, \mathcal{A}) \cap V(\mathcal{A}, \mathcal{B})$  has  $O(m^2n^2(m + n))$  vertices. ■

Let  $\Gamma = \{\gamma_1, \dots, \gamma_\ell\}$ , for  $\ell = O(mn)$ , be the sets of spheres that bound  $V(\mathcal{A}, \mathcal{B})$  and  $V(\mathcal{B}, \mathcal{A})$ . To decide whether  $V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A}) \neq \emptyset$ , it suffices to check whether  $M_i =$

$V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A}) \cap \gamma_i$  is nonempty for any  $1 \leq i \leq \ell$ . The same argument as in Lemma 2.1 implies that the complexity of  $M_i$  is  $O(mn(m+n))$ . Furthermore, we can compute  $M_i$  in time  $O(mn(m+n) \log mn)$ , by the same divide-and-conquer approach as computing  $V(\mathcal{A}, \mathcal{B}) \cap V(\mathcal{B}, \mathcal{A})$  in  $\mathbb{R}^2$ . Testing whether  $M_i \neq \emptyset$  for all  $i \leq \ell$ , we can determine in  $O(m^2 n^2 (m+n) \log mn)$  time whether  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F}) \leq \delta$ . Again, the optimization problem can be solved by the parametric search technique [AST94] with an extra  $\log^2 mn$  factor. Putting everything together, we have the following theorem.

**Theorem 2.4** *Given two families  $\mathcal{A}$  and  $\mathcal{B}$  of  $m$  and  $n$  disks (or balls), we can compute  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in time  $O(mn(m+n) \log^3 mn)$  in  $\mathbb{R}^2$  and in time  $O(m^2 n^2 (m+n) \log^3 mn)$  in  $\mathbb{R}^3$ .*

**Partial matching.** In many practical applications, we are interested in computing a partial matching between  $\mathcal{A}$  and  $\mathcal{B}$ . For example, in the docking problem, if there is no prior knowledge of where two molecules bind, the goal then is to find the maximal matched substructures without causing collision. See Figure 1 for an illustration. As in [IMV99], we define the partial collision-free Hausdorff distance problem as follows.

Given an integer  $k$ , we define  $h_k(\mathcal{A}, \mathcal{B})$  to be the  $k^{\text{th}}$  largest value in the set  $\{\delta(a, \mathcal{B}) \mid a \in \mathcal{A}\}$ , i.e., we discard the  $(k-1)$  balls of  $\mathcal{A}$  that are farthest from  $\mathcal{B}$ . Note that  $h(\mathcal{A}, \mathcal{B}) = h_1(\mathcal{A}, \mathcal{B})$ . We define  $H_k(\mathcal{A}, \mathcal{B})$ ,  $\sigma_k(\mathcal{A}, \mathcal{B}; T)$  as before. The above algorithm can be extended to compute  $\sigma_k(\mathcal{A}, \mathcal{B}; \mathcal{F})$  as follows.

Let  $\mathcal{V} = \{V_1, V_2, \dots, V_m\}$  be as defined above, and let  $\Xi(\mathcal{V})$  be the arrangement of  $\mathcal{V}$ . For each cell  $\Delta \in \Xi(\mathcal{V})$ , let  $\chi(\Delta)$  be the number of  $V_i$ 's that contain  $\Delta$ . For any point  $t$  in a cell  $\Delta$  with  $\chi(\Delta) > (m-k)$ ,  $h_k(\mathcal{A}+t, \mathcal{B}) \leq \delta$ . Hence, we compute  $\Xi(\mathcal{V})$  and  $\chi(\Delta)$  for each cell  $\Delta \in \Xi(\mathcal{V})$ , and then discard all the cells  $\Delta$  for which  $\chi(\Delta) \leq (m-k)$ . The remaining cells form the set  $T_1 = \{t \mid h_k(\mathcal{A}+t, \mathcal{B}) \leq \delta\}$ . By Remark following Lemma 2.1,  $\Xi$  has  $O(m^2 n)$  vertices, and it can be computed in  $O(m^2 n \log mn)$  time. Therefore,  $T_1$  can be computed in  $O(m^2 n \log mn)$  time. Similarly, we can compute  $T_2 = \{t \mid h_k(\mathcal{B}, \mathcal{A}+t) \leq \delta\}$  in  $O(mn^2 \log mn)$  time, and we can determine in  $O(mn(m+n) \log mn)$  time whether  $T_1 \cap T_2 \neq \emptyset$ . Similar results hold for the three-dimensional case. Putting everything together, we obtain the following.

**Theorem 2.5** *Given two families  $\mathcal{A}$  and  $\mathcal{B}$  of  $m$  and  $n$  balls and an integer  $k$ , we can compute  $\sigma_k(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in  $O(mn(m+n) \log^3 mn)$  time in  $\mathbb{R}^2$  and in  $O(m^2 n^2 (m+n) \log^3 mn)$  time in  $\mathbb{R}^3$ .*

### 3 Hausdorff Distance between Unions of Balls

In Section 3.1 we describe the algorithm for computing  $\sigma_U(\mathcal{A}, \mathcal{B}; T)$  in  $\mathbb{R}^2$ , and in Section 3.2, we present approximation algorithms in  $\mathbb{R}^2$  and  $\mathbb{R}^3$ .

#### 3.1 The exact 2D algorithm

Let  $\mathcal{A} = \{A_1, \dots, A_m\}$  and  $\mathcal{B} = \{B_1, \dots, B_n\}$  be two sets of disks in the plane. Let  $U_{\mathcal{A}}$  (resp.  $U_{\mathcal{B}}$ ) be the union of disks in  $\mathcal{A}$  (resp.  $\mathcal{B}$ ). As in Section 2 we focus on the decision problem.

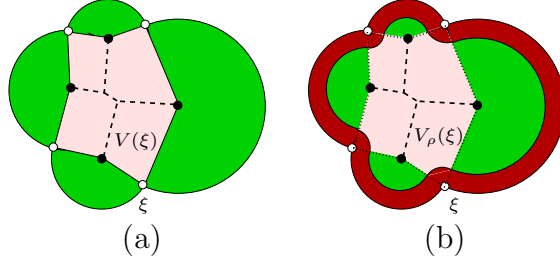


Figure 3: (a) Medial axis of a union of four disks centered at solid points — Voronoi diagram decomposes the union into 8 cells; (b) shrinking each Voronoi cell by  $D(\xi, \rho)$ .

For any point  $p$ , we have

$$\begin{aligned} \min_{q \in U_{\mathcal{B}}} d(p, q) &= \min_{j=1, \dots, n} d(p, B_j) \\ &= \min_{j=1, \dots, n} \max\{d(p, b_j) - r_j, 0\}. \end{aligned}$$

This value is greater than  $\delta$  if and only if  $\min_{j=1, \dots, n} d(p, b_j) - (r_j + \delta) > 0$ . In other words,  $h_U(\mathcal{A} + t, \mathcal{B}) > \delta$  if and only if there exists a point  $p \in U_{\mathcal{A}}$  such that  $p + t \notin U_{\mathcal{B}}(\delta)$ , where  $U_{\mathcal{B}}(\delta) = \bigcup_{j=1}^n D(b_j, r_j + \delta)$  is the union of the disks in  $\mathcal{B}$  expanded by  $\delta$ . Let  $B_j(\delta) = D(b_j, r_j + \delta)$ .

Let  $T_1 = \{t \mid h_U(\mathcal{A} + t, \mathcal{B}) \leq \delta\}$ .  $T_1$  is the set of all translations  $t$  such that  $U_{\mathcal{A}} + t \subseteq U_{\mathcal{B}}(\delta)$ . Our decision procedure thus computes the set  $T_1$ , and also computes the analogously defined set  $T_2 = \{t \mid h_U(\mathcal{B}, \mathcal{A} + t) \leq \delta\}$ , and then tests whether  $T_1 \cap T_2 \neq \emptyset$ . To understand the structure of  $T_1$ , we first study the case in which  $\mathcal{A}$  consists of just one disk  $A$ , with center  $a$  and radius  $\rho$ . For simplicity of notation, we denote  $U_{\mathcal{B}}(\delta)$  by  $U$ . Let  $\Gamma$  denote the set of circular arcs that constitute  $\partial U$ . By the result of [KLPS86],  $|\Gamma| \leq 6n - 12$ . Let  $Q$  denote the set of vertices of  $\partial U$ , which are the endpoints of the arcs of  $\Gamma$ . Clearly,  $|Q| \leq |\Gamma|$ .

Consider the medial-axis of  $\partial U$ , i.e., Voronoi diagram  $\text{Vor}(Q \cup \Gamma)$  of the boundary features of  $U$ , clipped to within  $U$ . This is a decomposition of  $U$  into cells, so that, for each  $\xi \in Q \cup \Gamma$ , the cell  $V(\xi)$  of  $\xi$  is the set of points  $x \in U$  such that  $d(x, \xi) \leq d(x, \xi')$ , for all  $\xi' \in Q \cup \Gamma$ . See Figure 3 (a) for an illustration.

The diagram has the following structure. For each  $\gamma \in \Gamma$ , let  $W(\gamma)$  denote the circular sector spanned by  $\gamma$  within the disk  $B_j(\delta)$  whose boundary contains  $\gamma$ . Let  $U' = U \setminus \bigcup_{\gamma \in \Gamma} W(\gamma)$ . The following lemma was observed in [AK01, AM97].

**Lemma 3.1** (a) For each  $\gamma \in \Gamma$ , we have  $V(\gamma) = W(\gamma)$ .  
(b) For each  $\xi \in Q$ , we have  $V(\xi) = U' \cap V'(\xi)$ , where  $V'(\xi)$  is the Voronoi cell of  $\xi$  in the Voronoi diagram  $\text{Vor}(Q)$  of  $Q$  alone. Moreover,  $V(\xi)$  is a convex polygon.

Lemma 3.1 implies that  $\text{Vor}(Q \cup \Gamma)$  yields a convex decomposition of  $U$  of linear size.

Returning to the study of the structure of  $T_1$ , by definition,  $A + t \subseteq U$  if and only if  $d(a + t, \xi) \geq \rho$ , where  $\xi$  is the feature of  $Q \cup \Gamma$ . This implies that the set  $T_1(A)$  of all translations  $t$  of  $A$  for which  $A + t \subseteq U$  is given by

$$T_1(A) = \bigcup_{\xi \in Q \cup \Gamma} V_\rho(\xi) - a,$$

where

$$V_\rho(\xi) = \{x \in V(\xi) \mid d(x, \xi) \geq \rho\}.$$

For  $\gamma \in \Gamma$ ,  $V_\rho(\gamma)$  is the sector obtained from  $W(\gamma)$  by shrinking it by distance  $\rho$  towards its center. For  $\xi \in Q$ ,  $V_\rho(\xi) = V(\xi) \setminus D(\xi, \rho)$ . See Figure 3 (b) for an illustration.

Now return to the original case in which  $A$  has  $m$  disks; we obtain

$$T_1 = \bigcap_{i=1}^m T_1(A_i) = \bigcap_{i=1}^m \bigcup_{\xi \in Q \cup \Gamma} (V_{\rho_i}(\xi) - a_i).$$

Note that each region  $T_1(A_i)$  is bounded by  $O(n)$  circular arcs, some of which are *convex* (those bounding shrunk sectors), and some are *concave* (those bounding shrunk Voronoi cells of vertices).

**Lemma 3.2** *For each pair of disks  $A_i, A_j \in \mathcal{A}$ , the complexity of  $T_1(A_i) \cap T_1(A_j)$  is  $O(n)$ .*

*Proof:* Clearly,  $T_1(A_i) \cap T_1(A_j)$  is also bounded by circular arcs, whose endpoints are either vertices of  $T_1(A_i)$  or  $T_1(A_j)$ , or are intersection points between an arc of  $\partial T_1(A_i)$  and an arc of  $\partial T_1(A_j)$ . It suffices to estimate the number of vertices of the latter kind. We bound separately the number of intersections of two convex arcs, one from each boundary, of two concave arcs, and of a convex arc with a concave arc.

*Intersections between two convex arcs:* Consider the set  $\mathcal{B}_{ij}$  of the  $2n$  disks

$$\{D(b_k - a_i, r_k + \delta - \rho_i), D(b_k - a_j, r_k + \delta - \rho_j)\}_{k=1}^n.$$

Let  $U_{ij}$  be the union of disks in  $\mathcal{B}_{ij}$ . We claim that any intersection point under consideration lies on  $\partial U_{ij}$ . Let  $\gamma, \gamma'$  be two arcs in  $\Gamma$  such that the arc  $\beta$  bounding  $V_{\rho_i}(\gamma) - a_i$  intersects the arc  $\beta'$  bounding  $V_{\rho_j}(\gamma') - a_j$  at a point  $t$ . Suppose that  $\gamma$  bounds  $B_k(\delta)$  and  $\gamma'$  bounds  $B_\ell(\delta)$ . Note that  $\beta$  bounds  $D(b_k - a_i, r_k + \delta - \rho_i)$  and  $\beta'$  bounds  $D(b_\ell - a_j, r_\ell + \delta - \rho_j)$ . Hence,  $t \in U_{ij}$ . If  $t$  does not lie on  $\partial U_{ij}$ , it is contained in the interior of some disk, say,  $D(b_s - a_i, r_s + \delta - \rho_i)$ . Then, by definition,  $A_i + t$  is fully contained in the interior of  $B_s(\delta)$ , which is easily seen to imply that  $t$  does not lie on  $\partial T_1(A_i)$ . This contradiction establishes the claim.

It thus follows, using the bound of [KLPS86], that the number of intersections under consideration is at most  $6 \cdot 2n - 12 = O(n)$ .

Using a similar but somewhat more involved argument, we show that there are  $O(n)$  intersection points between two concave arcs and between a convex and a concave arc. This completes the proof of the lemma.  $\blacksquare$

As a corollary, we obtain the following result.

**Lemma 3.3** *The complexity of  $T_1$  is  $O(m^2n)$ , and it can be computed in  $O(m^2n \log nm)$  time.*

Similarly, the set  $T_2$  can be computed in time  $O(mn^2 \log mn)$ . Finally, determining whether  $T_1 \cap T_2 \neq \emptyset$  can be accomplished by plane sweep, in time  $O(mn(m+n) \log mn)$ . Using parametric search as in [AST94], we can compute  $\sigma_U(\mathcal{A}, \mathcal{B})$  in  $O(mn(m+n) \log^3 mn)$  time.

To compute  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$ , we follow the same approach as computing  $\sigma(\mathcal{A}, \mathcal{B}; \mathcal{F})$  described in last Section. We omit the description here. Combined with similar argument as above, we can show that:

**Theorem 3.4** *Given two families  $\mathcal{A}$  and  $\mathcal{B}$  of  $m$  and  $n$  disks, we can compute both  $\sigma_U(\mathcal{A}, \mathcal{B})$  and  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in time  $O(mn(m+n)\log^3 mn)$ .*

## 3.2 Approximation algorithms

Since no good bound is known on the complexity of the medial axis of the union of  $n$  balls in  $\mathbb{R}^3$ , the naïve extension of previous exact algorithm to  $\mathbb{R}^3$  is quite inefficient. We therefore study approximation algorithms in this section. The first problem is as follows: given parameter  $\varepsilon > 0$ , compute a translation  $t$  of  $\mathcal{A}$  such that  $H_U(\mathcal{A}+t, \mathcal{B}) \leq (1+\varepsilon)\sigma_U(\mathcal{A}, \mathcal{B})$ , i.e.,  $H_U(\mathcal{A}+t, \mathcal{B})$  is an  $(1+\varepsilon)$ -approximation of  $\sigma_U(\mathcal{A}, \mathcal{B})$ .

Our approximation algorithm for  $\sigma_U(\mathcal{A}, \mathcal{B})$  follows the same approach as the one exploited in [AAR97, ABB95]. Let  $r(\mathcal{A})$  and  $r(\mathcal{B})$  denote the bottom left point, called the reference point, of the axis-parallel bounding box of  $U_{\mathcal{A}}$  and  $U_{\mathcal{B}}$ , respectively. Set  $\tau = r(\mathcal{B}) - r(\mathcal{A})$ . It is shown in [ABB95] that in  $\mathbb{R}^d$ ,

$$H_U(\mathcal{A} + \tau, \mathcal{B}) \leq (1 + \sqrt{d})\sigma_U(\mathcal{A}, \mathcal{B}).$$

Computing  $\tau$  takes  $O(m+n)$  time. We compute  $H_U(\mathcal{A} + \tau, \mathcal{B})$  using the parametric search technique. That is, given a parameter  $\delta$ , we first decide whether  $H_U(\mathcal{A} + \tau, \mathcal{B}) \leq \delta$ . As earlier, let  $U_{\mathcal{A}}(\delta) = \bigcup_i D(a_i, \rho_i + \delta)$  and  $U_{\mathcal{B}}(\delta) = \bigcup_j D(b_j, r_j + \delta)$ . Note that  $H_U(\mathcal{A} + \tau, \mathcal{B}) \leq \delta$  if and only if  $U_{\mathcal{A}} + \tau \subseteq U_{\mathcal{B}}(\delta)$  and  $U_{\mathcal{B}} \subseteq U_{\mathcal{A}}(\delta) + \tau$ . In order to test whether  $U_{\mathcal{A}} + \tau \subseteq U_{\mathcal{B}}(\delta)$ , we compute  $(U_{\mathcal{A}} + \tau) \cup U_{\mathcal{B}}(\delta)$ , the union of balls in  $A + \tau$  and  $B(\delta)$ , and check whether any ball of  $A$  appears on the boundary. If the answer is no, then  $U_{\mathcal{A}} + \tau \subseteq U_{\mathcal{B}}(\delta)$ . Similarly, we test whether  $U_{\mathcal{B}} \subseteq U_{\mathcal{A}}(\delta) + \tau$ . The total time spent is  $O((m+n)\log(m+n))$  in  $\mathbb{R}^2$  and  $O(m^2 + n^2)$  in  $\mathbb{R}^3$ . In order to compute an  $(1+\varepsilon)$ -approximation, we use the a standard trick of putting a grid in the neighborhood of  $r(\mathcal{B})$ . Omitting the details, we conclude the following.

**Theorem 3.5** *An  $(1+\varepsilon)$ -approximation of  $\sigma_U(\mathcal{A}, \mathcal{B})$  can be computed in  $O(\frac{1}{\varepsilon^2}(n+m)\log^3 nm)$  in  $\mathbb{R}^2$  and in  $O(\frac{1}{\varepsilon^3}(n^2 + m^2)\log^2 nm)$  in  $\mathbb{R}^3$ .*

Next we consider the problem of approximating  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$  in  $\mathbb{R}^3$ . Currently we do not have an efficient  $(1+\varepsilon)$ -approximation algorithm for computing  $\sigma_U(\mathcal{A}, \mathcal{B}; \mathcal{F})$ , so we present a pseudo-approximation algorithm.

Let  $\mathcal{K} = U_{\mathcal{B}} \oplus (-U_{\mathcal{A}})$ , where  $\oplus$  is the Minkowski sum, be the set of all placements of  $\mathcal{A}$  at which  $U_{\mathcal{A}}$  intersects  $U_{\mathcal{B}}$ .  $\mathcal{K} = \bigcup_{i,j} D(b_j - a_i, \rho_i + r_j)$ . Therefore  $\mathcal{F} = cl(\mathbb{R}^d \setminus \mathcal{K})$ . For a parameter  $\varepsilon \geq 0$ , let

$$\mathcal{K}(\varepsilon) = \bigcup_{i,j} D(b_j - a_i, (1 - \varepsilon)(\rho_i + r_j)),$$

and  $\mathcal{F}(\varepsilon) = cl(\mathbb{R}^d \setminus \mathcal{K}(\varepsilon))$ . We call a region  $X \subseteq \mathbb{R}^3$   $\varepsilon$ -free if  $\mathcal{F} \subseteq X \subseteq \mathcal{F}(\varepsilon)$ . This notion of approximating  $\mathcal{F}$  is motivated by some applications in which the data is noisy. For example, each atom in a protein is a “fuzzy” ball instead of a hard ball [HMWN02]. We

can model this fuzziness by allowing the boundary of any atom  $D(b, r)$  to lie in the annulus  $D(b, r) \setminus D(b, (1 - \varepsilon)r)$  for some  $\varepsilon > 0$ , and thus allowing the atoms of two molecules to penetrate a little in the desired placement.

Although  $\mathcal{F}$  can have large complexity, we show that an  $\varepsilon$ -free region  $X$  of small complexity can be computed. We thus compute  $X$  and a placement  $t \in X$  such that  $H_U(\mathcal{A} + t, \mathcal{B}) \leq (1 + \varepsilon)\sigma_U(\mathcal{A}, \mathcal{B}; X)$ .

**Lemma 3.6** *An  $\varepsilon$ -free region  $X$  of size  $O(mn/\varepsilon^3)$  can be computed in time  $O((mn/\varepsilon^3) \log(nm/\varepsilon))$  time.*

*Proof:* Let  $D = \{D(b_j - a_i, \rho_i + r_j) \mid 1 \leq i \leq m, 1 \leq j \leq n\}$ . We insert each ball  $D_{ij} \in D$  into an oct-tree  $T$ . An oct-tree, a bounding-box hierarchy, is an 8-way tree each of whose node  $v$  is associated with a cube  $C_v$ . The cubes associated with the children of  $v$  are obtained by dividing  $C_v$  into eight congruent cubes, each of size half of  $C_v$ .

In order to insert  $D_{ij}$ , we visit  $T$  in a top-down manner. Suppose we are at a node  $v$ . If  $C_v \subseteq D_{ij}$ , we mark  $v$  black and stop. If  $C_v \cap D_{ij} \neq \emptyset$  and the size of  $C_v$  is at least  $\varepsilon(\rho_i + r_j)/2$ , then we recursively visit the children of  $v$ . If all eight children of a node are marked black, we mark  $v$  black. Let  $V = \{v_1, v_2, \dots, v_k\}$  be the set of highest marked nodes i.e. each  $v_i$  is marked black but none of its ancestors is black. It can be shown that each  $D_{ij}$  marks at most  $O(1/\varepsilon^3)$  nodes black, so  $|V| = O(mn/\varepsilon^3)$ . The whole construction can be done in  $O((mn/\varepsilon^3) \log(mn/\varepsilon))$  time. We show that  $\mathcal{K}(\varepsilon) \subseteq \bigcup_{v \in V} C_v \subseteq \mathcal{K}$ . Set  $X = cl(\mathbb{R}^3 \setminus \bigcup_{v \in V} C_v)$ . We omit the details of computing the boundary representation of  $X$ . ■

Next, we show how to approximate  $\sigma_U(\mathcal{A}, \mathcal{B}; X)$ . Let  $\tau = r(\mathcal{B}) - r(\mathcal{A})$  be as defined above.

**Lemma 3.7** *Let  $t^* \in X$  be the closest point of  $\tau$  in  $X$ . Then  $H_U(\mathcal{A} + t^*, \mathcal{B}) \leq (1 + 2\sqrt{3})\sigma_U(\mathcal{A}, \mathcal{B}; X)$ .*

*Proof:* Let  $\delta^* = H_U(\mathcal{A} + t^*, \mathcal{B})$ , let  $\hat{\delta} = \sigma_U(\mathcal{A}, \mathcal{B}; X)$ , and let  $\hat{t} \in X$  be a placement s.t.  $H_U(\mathcal{A} + \hat{t}, \mathcal{B}) = \hat{\delta}$ .

$$\|\hat{t} - \tau\| = \|\hat{t} - r(\mathcal{B}) + r(\mathcal{A})\| = d(r(\mathcal{A}) + \hat{t}, r(\mathcal{B})) \leq \sqrt{3}\hat{\delta},$$

where the last inequality follows from an argument in [ABB95]. On the other hand,

$$\begin{aligned} \delta^* &\leq \hat{\delta} + \|\hat{t} - t^*\| \\ &\leq \hat{\delta} + \|\hat{t} - \tau\| + \|\tau - t^*\| \\ &\leq \hat{\delta} + 2\|\tau - \hat{t}\| \\ &\leq \hat{\delta} + 2\sqrt{3}\hat{\delta} = (1 + 2\sqrt{3})\sigma_U(\mathcal{A}, \mathcal{B}; X). \end{aligned}$$

■  
The closest point of  $\tau$  in  $X$  can be computed in  $O(mn/\varepsilon^3)$  time. Since we can compute  $H_U(\mathcal{A} + t^*, \mathcal{B})$  in  $O((n^2 + m^2) \log^2 nm)$  time, as described above, we can approximate  $\sigma_U(\mathcal{A}, \mathcal{B}; X)$  in  $O((n^2 + m^2) \log^2 mn)$  time. Next, we can again draw a grid around  $t^*$  and compute an  $(1 + \varepsilon)$ -approximation of  $\sigma_U(\mathcal{A}, \mathcal{B}; X)$ . Omitting the details, we conclude the following.

**Theorem 3.8** *Given  $\mathcal{A}, \mathcal{B}$ , in  $\mathbb{R}^3$  and  $\varepsilon > 0$ , we can compute an  $\varepsilon$ -free region  $X \subseteq \mathbb{R}^3$  and a placement  $t \in X$  of  $\mathcal{A}$  s.t.  $H_U(\mathcal{A} + t, \mathcal{B}) \leq (1 + \varepsilon)\sigma_U(\mathcal{A}, \mathcal{B}; X)$  in  $O(\frac{n^2+m^2}{\varepsilon^3} \log^2 nm)$  time.*

## 4 RMS Hausdorff Distance between Points

Let  $\mathcal{A} = \{a_1, \dots, a_m\}$  and  $\mathcal{B} = \{b_1, \dots, b_n\}$  be two sets of points in  $\mathbb{R}^d$ . We wish to compute  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ , the minimum rms Hausdorff distance between  $\mathcal{A}$  and  $\mathcal{B}$  under translation. For  $1 \leq i \leq m$ , let

$$f_i(t) = \delta^2(t, \mathcal{B} - a_i) = \min_{1 \leq j \leq n} d^2(t, b_j - a_i).$$

Each  $f_i$  is induced by the Voronoi diagram of the point set  $\mathcal{B} - a_i$ . We define

$$\begin{aligned} F_1(t) &= m \times \tilde{h}(\mathcal{A} + t, \mathcal{B}) = \sum_{i=1}^m \delta^2(a_i + t, \mathcal{B}) \\ &= \sum_{i=1}^m \delta^2(t, \mathcal{B} - a_i) = \sum_{i=1}^m f_i(t). \end{aligned}$$

By overlaying the Voronoi diagrams  $\text{Vor}(\mathcal{B} - a_i)$ ,  $1 \leq i \leq m$ , we can construct a subdivision of  $\mathbb{R}^d$  so that for any  $t \in \mathbb{R}^d$ ,  $F_1(t)$  can be computed quickly. Similarly we can construct a subdivision of  $\mathbb{R}^d$  for computing  $F_2(t) = n\tilde{h}(\mathcal{B}, \mathcal{A} + t)$  and overlay these two subdivisions to compute  $\tilde{\sigma}(\mathcal{A}, \mathcal{B}) = \min_t \max\{F_1(t)/m, F_2(t)/n\}$ . However, the size of these overlays is  $(mn)^{O(d)}$ . We therefore present an  $(1 + \varepsilon)$ -approximation algorithm for computing  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ , which runs in time  $O(\frac{mn}{\varepsilon^d} \log \frac{mn}{\varepsilon})$ .

We will describe an algorithm that given a family  $P_1, \dots, P_l$  of point sets in  $\mathbb{R}^d$  and a parameter  $\varepsilon > 0$ , preprocesses the points in time  $O((N/\varepsilon^d) \log(N/\varepsilon))$ ,  $N = \sum_i |P_i|$ , into a data structure, so that for a query point  $q$ , it returns in  $O(\log(N/\varepsilon))$  time a value  $\hat{F}(q)$ , with the property

$$\sum_{i=1}^l \delta^2(q, P_i) \leq \hat{F}(q) \leq (1 + \varepsilon) \sum_{i=1}^l \delta^2(q, P_i).$$

**Computing  $\hat{F}$ .** Our data structure is based on a recent result by Arya and Malamatos [AM02]. Given a set  $P$  of  $n$  points and a parameter  $\varepsilon > 0$ , they construct a partition  $\Xi$  of  $\mathbb{R}^d$  into  $O(n/\varepsilon^d)$  cells — each cell  $\Delta \in \Xi$  is the region lying between two nested hypercubes (the inner hypercube may be empty) and is associated with a point  $\phi(\Delta) \in P$  so that for any point  $q \in \Delta$ ,  $d(q, \phi(\Delta)) \leq (1 + \varepsilon)\delta(q, P)$ . Their structure is basically a compressed quad tree  $T$  [Sam89], built on a given hypercube  $C$  that contains  $P$ .  $\Xi$  and  $T$  can be constructed in  $O((n/\varepsilon^d) \log(n/\varepsilon))$  time, and the cell of  $\Xi$  containing a query point can be located in  $O(\log(n/\varepsilon))$  time.

Let  $C$  be a hypercube containing  $\bigcup_{i=1}^l P_i$ . We construct the above compressed quad tree  $T_i$  for  $P_i$ , and let  $\Xi_i$  be the resulting subdivision. We then merge  $T_1, \dots, T_l$  into a single compressed quad tree  $T$  [Sam89] and thus overlay  $\Xi_1, \dots, \Xi_l$ . Since all  $T_i$ 's are built using the same initial hypercube  $C$ , any two cells of  $\Xi_i$  and  $\Xi_j$  are either disjoint or one of them is contained in the other. Therefore,  $T_1, \dots, T_l$  can be merged without creating any additional

nodes. Let  $\Xi$  be the resulting overlay of  $\Xi_1, \dots, \Xi_l$ ;  $\Xi$  is a refinement of each  $\Xi_i$ . It can be shown that  $|\Xi| = O(N/\varepsilon^d)$ . For each cell  $\Delta \in \Xi$ , if  $\Delta$  lies inside the cell  $\Delta_i$  of  $\Xi_i$ , we store the function  $\hat{F}_\Delta(t) = \sum_{i=1}^l d^2(t, \phi(\Delta_i))$ , which is the equation of a parabola and can be stored using  $O(1)$  space. Since the merged tree  $T$  is also a compressed quad tree, the cell of  $\Xi$  containing a point can be computed in  $O(\log(N/\varepsilon))$  time. Omitting all the details, we conclude the following.

**Theorem 4.1** *Given a family  $P_1, \dots, P_l$  of point sets in  $\mathbb{R}^d$ , with a total of  $N$  points, and a parameter  $\varepsilon > 0$ , we can compute in  $O((N/\varepsilon^d) \log(N/\varepsilon))$  time a subdivision of  $\mathbb{R}^d$  of size  $O(N/\varepsilon^d)$  so that for any point  $q \in \mathbb{R}^d$ ,  $\hat{F}(q)$  can be computed in  $O(\log(N/\varepsilon))$  time.*

We mention two extensions of this structure. For each cell  $\Delta \in \Xi$ , if we store  $\phi(\Delta_i)$ , where  $\Delta \subseteq \Delta_i \in \Xi_i$ , we need  $O(Nl/\varepsilon^d)$  space. But by storing the points carefully and using persistence, we can construct a data structure of size  $O(N/\varepsilon^d)$  so that for any query point  $q \in \mathbb{R}^d$ , we can return in  $O(\log(N/\varepsilon))$  time  $\mu_1, \dots, \mu_l$ , such that  $\delta(q, P_i) \leq \mu_i \leq (1+\varepsilon)\delta(q, P_i)$  for every  $i$ .

We can also modify the data structure so that it ignores distances that exceeds a prespecified threshold; this is sometimes used in matching protein structures [HMWN02]. It is also straightforward to modify the data structure so that it reports the number of points that have distances below this threshold. This can be considered as a variant of partial matching problem under summed measure.

**Computing  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ .** Let  $P_i = \mathcal{B} - a_i$  for  $1 \leq i \leq m$ . We construct the decomposition  $\Xi_1$  and the data structure, as described in Theorem 4.1, for  $P_1, \dots, P_m$ . For any point  $t \in \mathbb{R}^d$ , it can compute a value  $\hat{F}_1(t)$  such that

$$F_1(t) \leq \hat{F}_1(t) \leq (1 + \varepsilon)F_1(t).$$

Similarly we construct a subdivision  $\Xi_2$  on  $Q_1, \dots, Q_n$ , where  $Q_j = b_j - \mathcal{A}$ , that approximates  $F_2(t)$ . By overlaying  $\Xi_1$  and  $\Xi_2$  and searching over all cells of the overlay, we can compute a placement  $t \in \mathbb{R}^d$  such that  $\tilde{H}(\mathcal{A} + t, \mathcal{B}) \leq (1 + \varepsilon)\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ . Hence, we obtain the following.

**Theorem 4.2** *Given two sets  $\mathcal{A}$  and  $\mathcal{B}$  of  $m$  and  $n$  points in  $\mathbb{R}^d$  and a parameter  $\varepsilon > 0$ , we can compute a translation vector  $t$  in  $O((mn/\varepsilon^d) \log(mn/\varepsilon))$  time so that  $\tilde{H}(\mathcal{A} + t, \mathcal{B}) \leq (1 + \varepsilon)\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ .*

Finally, we briefly describe a simple randomized algorithm to approximate  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ . Let  $t^*$  be the optimal translation, i.e.,  $\tilde{H}(\mathcal{A} + t^*, \mathcal{B}) = \tilde{\sigma}(\mathcal{A}, \mathcal{B})$ . Using Markov's inequality, we can prove the following.

**Lemma 4.3** *For a random point  $a_k$  from  $\mathcal{A}$ ,  $\delta(a_k + t^*, \mathcal{B}) \leq 2\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ , with probability greater than  $1/2$ .*

Choose a random point  $a_k \in \mathcal{A}$ . Let  $t_j = b_j - a_k$  and  $\delta_j = \tilde{H}(\mathcal{A} + t_j, \mathcal{B})$ , for  $1 \leq j \leq n$ . Following the same argument as in Lemma 3.7 and using Lemma 4.3, we can show that  $\min_j \delta_j$  is a constant-factor approximation of  $\tilde{\sigma}(\mathcal{A}, \mathcal{B})$ , with probability greater than  $1/2$ . Computing  $\delta_j$  exactly is expensive in  $\mathbb{R}^d$ , therefore we compute an approximate value of

$\delta_j$ , for  $1 \leq j \leq n$ , in time  $O((m+n) \log mn)$ , by performing approximate nearest-neighbor queries [AM02]. We can improve the algorithm to compute an  $(1 + \varepsilon)$ -approximation of  $\tilde{\sigma}(A, B)$  using the same technique as in Section 3. We thus obtain the following result.

**Theorem 4.4** *Given two sets  $A$  and  $B$  of  $m$  and  $n$  points in  $\mathbb{R}^d$  and a parameter  $\varepsilon > 0$ , we can compute in time  $O((nm/\varepsilon^d) \log nm)$  a translation  $t$ , such that with probability greater than  $1/2$ ,  $\tilde{H}(A, B + t) \leq (1 + \varepsilon)\tilde{\sigma}(A, B)$ .*

## References

- [AAR97] Oswin Aichholzer, Helmut Alt, and Günter Rote. Matching shapes with a reference point. *International Journal on Computational Geometry and Applications*, 7:349–363, 1997.
- [ABB95] Helmut Alt, Bernd Behrends, and Johannes Blömer. Approximate matching of polygonal shapes. *Ann. Math. Artif. Intell.*, 13:251–266, 1995.
- [AG99] H. Alt and L. Guibas. Discrete geometric shapes: Matching, Interpolation, and Approximation. *Handbook of Computational Geometry (J.-R. Sack and J. Urrutia eds)*, 1999.
- [AK00] N. Amenta and R. Kolluri. Accurate and efficient unions of balls. In *Proceedings of the 16th Annual ACM Symposium on Computational Geometry*, pages 119–128, 2000.
- [AK01] N. Amenta and R. Kolluri. The medial axis of a union of balls. *Computational Geometry: Theory and Applications*, 20:25–37, 2001.
- [AM93] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22:540–570, 1993.
- [AM97] D. Attali and A. Montanvert. Computing and simplifying 2D and 3D continuous skeletons. *Computer Vision and Image Understanding*, 67(3):261–273, 1997.
- [AM02] S. Arya and T. Malamatos. Linear-size approximate voronoi diagrams. In *Proc. 13th ACM-SIAM Symp. on Discrete Algorithms*, pages 147–155, 2002.
- [AST94] P. K. Agarwal, M. Sharir, and S. Toledo. Applications of parametric searching in geometric optimization. *J. Algorithms*, 17:292–318, 1994.
- [Ata83] M. J. Atallah. A linear time algorithm for the Hausdorff distance between convex polygons. *Inform. Process. Lett.*, 17:207–209, 1983.
- [BJ85] P. J. Besl and R. C. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17(1):75–154, 1985.
- [CD86] R. T. Chin and C. R. Dyer. Model-based recognition in robot vision. *ACM Computing Surveys*, 18(1):67–108, 1986.

- [CDEK99] L. P. Chew, D. Dor, A. Efrat, and K. Kedem. Geometric pattern matching in  $d$ -dimensional space. *Discrete and Computational Geometry*, 21:257–274, 1999.
- [CGH<sup>+</sup>97] L. P. Chew, M. T. Goodrich, D. P. Huttenlocher, K. Kedem, J. M. Kleinberg, and D. Dravets. Geometric pattern matching under euclidean motion. *Comput. Geom. Theory Appl.*, 7:113–124, 1997.
- [CS98] D. Cardoze and L. Schulman. Pattern matching for spatial point sets. In *Proc. 39th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 156–165, 1998.
- [GMO94] M. T. Goodrich, J. S. 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.
- [HKK92] D. P. Huttenlocher, K. Kedem, and J. M. Kleinberg. On dynamic Voronoi diagrams and the minimum Hausdorff distance for point sets under Euclidean motion in the plane. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 110–120, 1992.
- [HKR93] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.
- [HKS93] D. P. Huttenlocher, K. Kedem, and M. Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete Comput. Geom.*, 9:267–291, 1993.
- [HMWN02] I. Halperin, B. Ma, H. Wolfson, and R. Nussinov. Principles of docking: An overview of search algorithms and a guide to scoring functions. *Proteins: Structure, Function, and Genetics*, 47:409–443, 2002.
- [IMV99] P. Indyk, R. Motwani, and S. Venkatasubramanian. Geometric matching under noise: Combinatorial bounds and algorithms. In *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms*, pages 457–465, 1999.
- [IV00] P. Indyk and S. Venkatasubramanian. Approximate congruence in nearly-linear time. In *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, pages 354–360, 2000.
- [KLPS86] K. Kedem, R. Livne, J. Pach, and Micha Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Comput. Geom.*, 1:59–71, 1986.
- [Meg83] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30:852–865, 1983.
- [NLWN95] R. Norel, S. L. Lin, H. Wolfson, and R. Nussinov. Molecular surface complementarity at protein-protein interfaces: the critical role played by surface normals at well placed, sparse points in docking. *J. Mol. Biol.*, 252:263–273, 1995.

- [Sam89] H. Samet. *Spatial Data Structures: Quadtrees, Octrees, and Other Hierarchical Methods*. Addison-Wesley, Reading, MA, 1989.
- [SL02] S. Seeger and X. Laboureaux. Feature extraction and registration: An overview. *Principles of 3D Image Analysis and Synthesis*, pages 153–166, 2002.