

# Maximizing the Area of Overlap of two Unions of Disks under Rigid Motion

Mark de Berg<sup>1,\*</sup>, Sergio Cabello<sup>2,\*\*</sup>, Panos Giannopoulos<sup>3,\*\*\*</sup>, Christian Knauer<sup>4</sup>, René van Oostrum<sup>3</sup>, and Remco C. Veltkamp<sup>3</sup>

<sup>1</sup> Faculty of Mathematics and Computing Science, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands. [m.t.d.berg@TUE.nl](mailto:m.t.d.berg@TUE.nl)

<sup>2</sup> IMFM, Department of Mathematics, Jadranska 19, SI-1000 Ljubljana, Slovenia. [sergio.cabello@imfm.uni-lj.si](mailto:sergio.cabello@imfm.uni-lj.si)

<sup>3</sup> Institute of Information and Computing Sciences, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, The Netherlands.

[{panos, rene, Remco.Veltkamp}@cs.uu.nl](mailto:{panos, rene, Remco.Veltkamp}@cs.uu.nl)

<sup>4</sup> Institut für Informatik, Freie Universität Berlin, Takustraße 9, D-14195 Berlin, Germany. [Christian.Knauer@inf.fu-berlin.de](mailto:Christian.Knauer@inf.fu-berlin.de)

**Abstract.** Let  $A$  and  $B$  be two sets of  $n$  resp.  $m$  disjoint unit disks in the plane, with  $m \geq n$ . We consider the problem of finding a translation or rigid motion of  $A$  that maximizes the total area of overlap with  $B$ . The function describing the area of overlap is quite complex, even for combinatorially equivalent translations and, hence, we turn our attention to approximation algorithms. We give deterministic  $(1 - \epsilon)$ -approximation algorithms for translations and for rigid motions, which run in  $O((nm/\epsilon^2) \log(m/\epsilon))$  and  $O((n^2m^2/\epsilon^3) \log m)$  time, respectively. For rigid motions, we can also compute a  $(1 - \epsilon)$ -approximation in  $O((m^2 n^{4/3} \Delta^{1/3}/\epsilon^3) \log n \log m)$  time, where  $\Delta$  is the diameter of set  $A$ . Under the condition that the maximum area of overlap is at least a constant fraction of the area of  $A$ , we give a probabilistic  $(1 - \epsilon)$ -approximation algorithm for rigid motions that runs in  $O((m^2/\epsilon^4) \log(m/\epsilon) \log^2 m)$  time. Our results generalize to the case where  $A$  and  $B$  consist of possibly intersecting disks of different radii, provided that (i) the ratio of the radii of any two disks in  $A \cup B$  is bounded, and (ii) within each set, the maximum number of disks with a non-empty intersection is bounded.

## 1 Introduction

Shape matching is a fundamental problem in computational geometry with applications in computer vision: given two shapes  $A$  and  $B$  and a distance measure,

---

\* Research supported by the Netherlands' Organisation for Scientific Research under project no. 639.023.301.

\*\* Research conducted at IICS, Utrecht University, The Netherlands, and partially supported by the Cornelis Lely Stichting.

\*\*\* Research partially supported by the Dutch Technology Foundation STW under project no. UIF 5055, and a Marie Curie Fellowship of the EC programme 'Combinatorics, Geometry, and Computation', FU Berlin, HPMT-CT-2001-00282.

one wants to determine a transformation of  $A$  — such as a translation or a rigid motion — that minimizes its distance to  $B$ . Typical problems include: matching point sets with respect to the Hausdorff or bottleneck distance and matching polygons with respect to the Hausdorff or Fréchet distance between their boundaries; see Alt and Guibas [4] for a survey. The *area of overlap* of two polygons is less sensitive to noise than the Hausdorff or Fréchet distance between their boundaries [8, 3] and therefore more appropriate for certain applications.

Mount et al. [11] showed that the function of the area of overlap of two simple polygons, with  $n$  and  $m$  vertices, under translation is continuous and has  $O((nm)^2)$  pieces, with each piece being a polynomial of degree at most two. A representation of the function can be computed in  $O((nm)^2)$  time. No algorithm is known that computes the translation that maximizes the area of overlap and does not compute the complete representation of the overlap function. One of the open problems mentioned by Mount et al. was to give efficient matching algorithms for objects with curved boundaries. De Berg et al. [8], gave an  $O((n + m) \log(n + m))$  algorithm for determining the optimal translation for *convex* polygons, while Alt et al. [3] gave a constant-factor approximation of the minimum *area of symmetric difference* of two convex shapes.

We study the following problem: given two sets  $A$  and  $B$  of disks in the plane, we would like to find a rigid motion that maximizes the area of overlap. Our main goal is to match two shapes, each being expressed as a union of disks; thus the overlap we want to maximize is the overlap between the two unions (which is not the same as the sum of overlaps of the individual disks). In the most general setting we assume the following: (i) the largest disk is only a constant times larger than the smallest one, and (ii) any disk in  $A$  intersects only a constant number of other disks in  $A$ , and the same holds for  $B$ .

Since any two- or three-dimensional shape can be efficiently approximated by a finite union of disks or balls—see, for example, the works by O’Rourke and Badler [12] and Amenta and Kolluri [5]—our algorithms can be used to match a variety of shapes. Ranjan and Fournier [13] also used the union of disks or spheres representation to interpolate between two shapes. The assumptions (i) and (ii) above will often be satisfied when disks or balls are used to approximate objects, although the constant in assumption (i) may become large when the approximated objects have fine details. Moreover, both assumptions make perfect sense in molecular modelling with the hard sphere model [10]. Under this model the radii range of the spheres is fairly restricted and no center of a sphere can be inside another sphere; a simple packing argument shows that the latter implies assumption (ii). A related problem with applications in protein shape matching was examined by Agarwal et al. [1], who gave algorithms for minimizing the Hausdorff distance between two unions of discs or balls under translations.

Another application comes from weighted point-set matching. Consider a two- or three-dimensional shape that is reduced to a set of descriptive feature points, each of them weighted relatively to its importance. For example, a curve or a contour can be reduced to a set of points of high curvature. We can assign to each point a ball centered at it with radius relative to its weight. Thus, the

two shapes are represented as unions of balls and a possible measure of their similarity is the area of overlap of the two unions.

Recently, Cheong et al. [6] gave an almost linear, probabilistic approximation algorithm that computes the maximum area of overlap under translations up to an absolute error with high probability. When the maximum overlap is at least a constant fraction of the area of one of the two sets, the absolute error is in fact a relative error. This is usually good enough for shape matching, since if two shapes are quite dissimilar we usually do not care about how bad the match exactly is. A direct application of the technique of Cheong et al. to rigid motions gives an  $O((m^2/\epsilon^6) \log(m/\epsilon) \log^3 m)$  time algorithm that requires the computation of intersection points of algebraic curves, which is not very practical.

Our contributions are the following. First, we show in Section 2 that the maximum number of combinatorially distinct translations of  $A$  with respect to  $B$  can be as high as  $\Theta(n^2 m)$ . When rotations are considered as well, the complexity is  $O(n^3 m^2)$ . Moreover, the function describing the area of overlap is quite complex, even for combinatorially equivalent placements. Therefore, the focus of our paper is on approximation algorithms. Next, we give a lower bound on the maximum area of overlap under translations, expressed in the number of pairs of disks that contribute to that area. This is a vital ingredient of almost all our algorithms.

In the remaining sections, we present our approximation algorithms. For the sake of clarity we describe the algorithms for the case of disjoint unit disks. It is not hard to adapt the algorithms to sets of disks satisfying assumptions (i) and (ii) above. Due to lack of space, the necessary changes for the latter, and several other proofs, are omitted; these can be found in the full version of this paper. For any  $\epsilon > 0$ , our algorithms can compute a  $(1 - \epsilon)$ -approximation of the optimum overlap. For translations we give an algorithm that runs in  $O((nm/\epsilon^2) \log(m/\epsilon))$  time. This is worse than the algorithm of Cheong et al., but our algorithm is deterministic and our error is always relative, even when the optimum is small. It also forms an ingredient to our algorithm for rigid motions, which runs in  $O((n^2 m^2/\epsilon^3) \log m)$  time. If  $\Delta$  is the diameter of set  $A$ —recall that we are dealing with unit disks—the running time of the latter becomes  $O((m^2 n^{4/3} \Delta^{1/3}/\epsilon^3) \log n \log m)$ , which yields an improvement when  $\Delta = o(n^2/\log^3 n)$ . Note that in many applications the union will be connected, which implies that the diameter will be  $O(n)$ . If the area of overlap is a constant fraction of the area of the union of  $A$ , we can get a probabilistic algorithm for rigid motions that runs in  $O((m^2/\epsilon^4) \log(m/\epsilon) \log^2 m)$  time and succeeds with high probability.

## 2 Basic properties of the overlap function

Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$ , be two sets of disjoint unit disks in the plane, with  $n \leq m$ . We consider the disks to be closed. Both  $A$  and  $B$  lie in the same two-dimensional coordinate space, which we call the *work space*;

their initial position is denoted simply by  $A$  and  $B$ . We consider  $B$  to be fixed, while  $A$  can be translated and/or rotated relative to  $B$ .

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

Let  $\text{Int}(C), V(C)$  be, respectively, the interior and area of a compact set  $C \in \mathbb{R}^2$ , and let  $\mathcal{V}_{ij}(\vec{t}, \theta) = V(A_i(\vec{t}, \theta) \cap B_j)$ . The *area of overlap* of  $A(\vec{t}, \theta)$  and  $B$ , as  $\vec{t}, \theta$  vary, is a function  $\mathcal{V} : \mathcal{I} \rightarrow \mathbb{R}$  with  $\mathcal{V}(\vec{t}, \theta) = V((\bigcup A(\vec{t}, \theta)) \cap (\bigcup B))$ . Thus the problem that we are studying can be stated as follows: Given two sets  $A, B$ , defined as above, compute a rigid motion  $I_{\vec{t}_{opt}, \theta_{opt}}$  that maximizes  $\mathcal{V}(\vec{t}, \theta)$ .

Let  $d_{ij}(\vec{t}, \theta)$  be the Euclidean distance between the centers of  $A_i(\vec{t}, \theta)$  and  $B_j$ . Also, let  $r_i$  be the Euclidean distance of  $A_i$ 's center to the origin. For simplicity, we write  $\mathcal{V}(\vec{t}), \mathcal{V}_{ij}(\vec{t}), d_{ij}(\vec{t})$  when  $\theta$  is fixed and  $\mathcal{V}(\theta), \mathcal{V}_{ij}(\theta), d_{ij}(\theta)$  when  $\vec{t}$  is fixed. The Minkowski sum of two planar sets  $A$  and  $B$ , denoted by  $A \oplus B$ , is the set  $\{p_1 + p_2 : p_1 \in A, p_2 \in B\}$ . Similarly the Minkowski difference  $A \ominus B$  is the set  $\{p_1 - p_2 : p_1 \in A, p_2 \in B\}$ .

**Theorem 1.** *Let  $A$  be a set of  $n$  disjoint unit disks in the plane, and  $B$  a set of  $m$  disjoint unit disks, with  $n \leq m$ . The maximum number of combinatorially distinct placements of  $A$  with respect to  $B$  is  $\Theta(n^2 m)$  under translations, and  $O(n^3 m^2)$  under rigid motions.*

This theorem implies that explicitly computing the subdivision of the configuration space into cells with combinatorially equivalent placements is highly expensive. Moreover, the computation for rigid motions can cause non-trivial numerical problems since it involves algebraic equations of degree six [4]. Finally, the optimization problem in a cell of this decomposition is far from easy: one has to maximize a function consisting of a linear number of terms. Therefore we turn our attention to approximation algorithms. The following theorem, which gives a lower bound on the maximum area of overlap, will be instrumental in obtaining a relative error.

**Theorem 2.** *Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$  be two sets of disjoint unit disks in the plane. Let  $\vec{t}_{opt}$  be the translation that maximizes the area of overlap  $\mathcal{V}(\vec{t})$  of  $A(\vec{t})$  and  $B$  over all possible translations  $\vec{t}$  of set  $A$ . If  $k_{opt}$  is the number of overlapping pairs  $A_i(\vec{t}_{opt}), B_j$ , then  $\mathcal{V}(\vec{t}_{opt})$  is  $\Theta(k_{opt})$ .*

*Proof.* First, note that  $\mathcal{V}(\vec{t}_{opt}) \leq k_{opt}\pi$ . Since we are considering only translations, the configuration space is two-dimensional. For each pair  $A_i(\vec{t}_{opt}), B_j$  for which  $A_i(\vec{t}_{opt}) \cap B_j \neq \emptyset$ , we draw, in configuration space, the region of translations  $\mathcal{K}_{ij}$  that bring the center of  $A_i(\vec{t}_{opt})$  into  $B_j$ ; see Figure 1. Such a

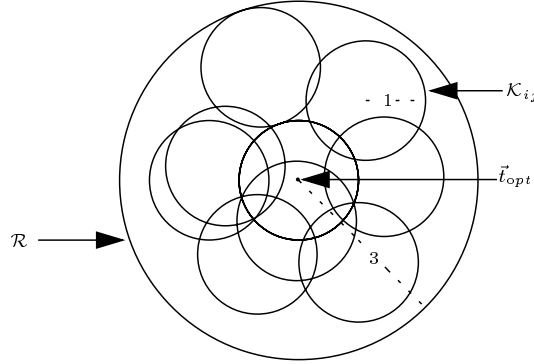


Fig. 1. All disks  $\mathcal{K}_{ij}$  are confined within a disk  $\mathcal{R}$  of area  $9\pi$ .

region is a unit disk that is centered at a distance at most 2 from  $\vec{t}_{opt}$ . Thus, all regions  $\mathcal{K}_{ij}$  are fully contained in a disk  $\mathcal{R}$ , centered at  $\vec{t}_{opt}$ , of radius 3. By a simple volume argument, there must be a point  $\vec{t}_\# \in \mathcal{R}$  (which represents a translation) that is covered by at least  $k_{opt}/9$  disks  $\mathcal{K}_{ij}$ . Each of the corresponding pairs  $A_i(\vec{t}_\#), B_j$  has an overlap of at least  $2\pi/3 - \sqrt{3}/2$ . Thus,  $\mathcal{V}(\vec{t}_{opt}) \geq \mathcal{V}(\vec{t}_\#) \geq (2\pi/27 - \sqrt{3}/18)k_{opt}$ .  $\square$

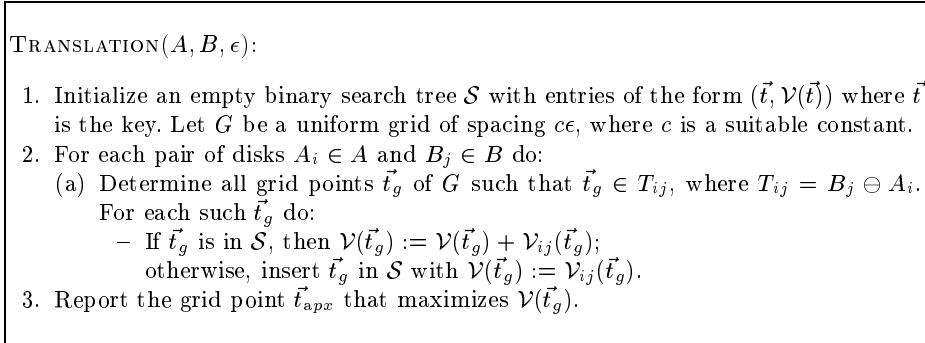
### 3 A $(1 - \epsilon)$ -approximation algorithm for translations

Theorem 2 suggests the following simple approximation algorithm: compute the arrangement of the regions  $\mathcal{K}_{ij}$ , with  $i = 1, \dots, n$  and  $j = 1, \dots, m$ , and pick any point of maximum depth. Such a point corresponds to a translation  $\vec{t}$  that gives a constant-factor approximation. An analysis similar to that of Theorem 2 leads to a factor of  $(2/3 - \sqrt{3}/2\pi) \approx 0.39$ . It is possible to do much better, however.

We proceed, first with a deterministic  $(1 - \epsilon)$ -approximation algorithm. Since both  $A$  and  $B$  consist of disjoint disks, we have  $\mathcal{V}(\vec{t}) = \sum_{A_i \in A, B_j \in B} \mathcal{V}_{ij}(\vec{t})$ . The algorithm is based on sampling the configuration space by using a uniform grid. This is possible due to the following lemma that implies that, in terms of absolute error, it is not too bad if we choose a translation which is close to the optimal one.

**Lemma 1.** *Let  $k$  be the number of overlapping pairs  $A_i(\vec{t}, \theta), B_j$  for some  $\vec{t} \in \mathbb{R}^2, \theta \in [0, 2\pi)$ . For any given  $\delta > 0$  and any  $\vec{t}' \in \mathbb{R}^2$  for which  $|\vec{t} - \vec{t}'| = O(\delta)$ , we have  $\mathcal{V}(\vec{t}', \theta) = \mathcal{V}(\vec{t}, \theta) - O(k\delta)$ .*

Instead of computing  $\mathcal{V}(\vec{t})$  for every grid translation directly, we use a simple voting scheme that speeds up the algorithm by a linear factor. Algorithm TRANSLATION is given in Figure 2.



**Fig. 2.** Algorithm TRANSLATION( $A, B, \epsilon$ ).

**Theorem 3.** *Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$  be two sets of disjoint unit disks in the plane. Let  $\vec{t}_{opt}$  be the translation that maximizes  $\mathcal{V}(\vec{t})$ . Then, for any given  $\epsilon > 0$ , TRANSLATION( $A, B, \epsilon$ ) computes a translation  $\vec{t}_{apx}$ , for which  $\mathcal{V}(\vec{t}_{apx}) \geq (1 - \epsilon)\mathcal{V}(\vec{t}_{opt})$ , in  $O((mn/\epsilon^2) \log(m/\epsilon))$  time.*

*Proof.* It follows from Theorem 2 that there must be at least one pair with significant overlap in an optimal translation. This implies that the grid point closest to the optimum must have a pair of disks overlapping, and so the algorithm checks at least one grid translation  $\vec{t}_g$  for which  $|\vec{t}_{opt} - \vec{t}_g| = O(\epsilon)$ . Let  $k_{opt}$  be the number of overlapping pairs  $A_i(\vec{t}_{opt}), B_j$ . According to Lemma 1, by setting  $\theta = 0$ , we have that  $\mathcal{V}(\vec{t}_{opt}) - \mathcal{V}(\vec{t}_{apx}) \leq \mathcal{V}(\vec{t}_{opt}) - \mathcal{V}(\vec{t}_g) = O(k_{opt}\epsilon)$ . By Theorem 2, we have  $\mathcal{V}(\vec{t}_{opt}) = \Theta(k_{opt})$ , and the approximation bound follows.

The algorithm considers  $O(1/\epsilon^2)$  grid translations  $\vec{t}_g$  per pair of disks. Each translation is handled in  $O(\log(nm/\epsilon^2))$  time. Thus, the total running time is  $O((nm/\epsilon^2) \log(nm/\epsilon^2)) = O((nm/\epsilon^2) \log(m/\epsilon))$ .  $\square$

## 4 The rotational case

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

Observe that this problem has a one-dimensional configuration space: the angle of rotation. Consider the function  $\mathcal{V} : [0, 2\pi) \rightarrow \mathbb{R}$  with  $\mathcal{V}(\theta) := V((\bigcup A(\theta)) \cap (\bigcup B)) = \sum_{A_i \in A, B_j \in B} \mathcal{V}_{ij}(\theta)$ . For now, our objective is to guarantee an absolute error on  $\mathcal{V}$  rather than a relative one. We start with a result that bounds the difference in overlap for two relatively similar rotations. Recall that  $r_i$  is the distance of  $A_i$ 's center to the origin.

ROTATION( $A, B, \delta$ ):

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

**Fig. 3.** Algorithm ROTATION( $A, B, \delta$ ).

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

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

**Lemma 3.** *For any  $A_i, B_j$  with  $r_i > 0$ , and any given  $\delta > 0$ , we have  $|R_{ij}| / (\delta / (2r_i)) = O(1/\delta)$ .*

This lemma implies that we have to consider only  $O(1/\delta)$  sample rotations per pair of disks. Thus we need to check  $O(nm/\delta)$  rotations in total. It seems that we would have to compute all overlaps at every rotation from scratch, but here Lemma 2 comes to the rescue: in between two consecutive rotations  $\theta, \theta'$  defined for a given pair  $A_i, B_j$  there may be many other rotations, but if we conservatively estimate the overlap of  $A_i, B_j$  as the minimum overlap at  $\theta$  and

$\theta'$ , we do not lose too much. In Figure 3, algorithm ROTATION is described in more detail; the value  $\tilde{\mathcal{V}}(\theta)$  is the conservative estimate of  $\mathcal{V}(\theta)$ , as just explained.

**Lemma 4.** *Let  $\theta_{opt}$  be a rotation that maximizes  $\mathcal{V}(\theta)$  and let  $k_{opt}$  be the number of overlapping pairs  $A_i(\theta_{opt}), B_j$ . For any given  $\delta > 0$ , the rotation  $\theta_{apx}$  reported by ROTATION( $A, B, \delta$ ) satisfies  $\mathcal{V}(\theta_{opt}) - \mathcal{V}(\theta_{apx}) = O(k_{opt}\delta)$  and can be computed in  $O((mn/\delta) \log m)$  time.*

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

Any rigid motion can be described as a translation followed by a rotation around the origin. This is used in algorithm RIGIDMOTION described in Figure 4, which combines the algorithms for translations and for rotations to obtain an  $(1 - \epsilon)$ -approximation for rigid motions.

RIGIDMOTION( $A, B, \epsilon$ ):

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

**Fig. 4.** Algorithm RIGIDMOTION( $A, B, \epsilon$ ).

**Theorem 4.** *Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$ , with  $n \leq m$ , be two sets of disjoint unit disks in the plane. Let  $I_{\vec{t}_{opt}, \theta_{opt}}$  be a rigid motion that maximizes  $\mathcal{V}(\vec{t}, \theta)$ . Then, for any given  $\epsilon > 0$ , RIGIDMOTION( $A, B, \epsilon$ ) computes a rigid motion  $I_{\vec{t}_{apx}, \theta_{apx}}$  such that  $\mathcal{V}(\vec{t}_{apx}, \theta_{apx}) \geq (1 - \epsilon)\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$  in  $O((n^2m^2/\epsilon^3) \log m)$  time.*

*Proof.* We will show that  $\mathcal{V}(\vec{t}_{apx}, \theta_{apx})$  approximates  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$  up to an absolute error. To convert the absolute error into a relative error, and hence show the algorithm's correctness, we use again Theorem 2.

Let  $A_{opt}$  be the set of disks in  $A$  that participate in the optimal solution and let  $|A_{opt}| = \bar{k}_{opt}$ . Since the 'kissing' number of unit open disks is six, we have that  $k_{opt} < 6\bar{k}_{opt}$ , where  $k_{opt}$  is the number of overlapping pairs in the optimal solution. Next, imagine that RIGIDMOTION( $A_{opt}, B, \epsilon$ ) is run instead

of  $\text{RIGIDMOTION}(A, B, \epsilon)$ . Of course, an optimal rigid motion for  $A_{opt}$  is an optimal rigid motion for  $A$  and the error we make by applying a non-optimal rigid motion to  $A_{opt}$  bounds the error we make when applying the same rigid motion to  $A$ .

Consider a disk  $A_i \in A_{opt}$  and an intersecting pair  $A_i(\vec{t}_{opt}, \theta_{opt}), B_j$ . Since, at some stage, the algorithm will use  $B_j$ 's center as the center of rotation, and  $I_{\vec{t}_{opt}, \theta_{opt}} = R_{\theta_{opt}} \circ T_{\vec{t}_{opt}}$ , we have that  $A_i(\vec{t}_{opt}) \cap B_j \neq \emptyset$  if and only if  $A_i(\vec{t}_{opt}, \theta_{opt}) \cap B_j \neq \emptyset$ . Hence, we have that  $\vec{t}_{opt} \in T_{ij}$  and the algorithm will consider some grid translation  $\vec{t}_g \in T_{ij} = B_j \ominus A_i$ , for which  $|\vec{t}_{opt} - \vec{t}_g| = O(\epsilon)$ . By Lemma 1 we have  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt}) - \mathcal{V}(\vec{t}_g, \theta_{opt}) = O(k_{opt}\epsilon) = O(\bar{k}_{opt}\epsilon)$ .

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

Now, using the fact that  $\mathcal{V}(\vec{t}_g, \theta_{apx}^g) \leq \mathcal{V}(\vec{t}_{apx}, \theta_{apx})$  and that  $\bar{k}_{opt} \leq k_{opt}$ , and putting it all together we get  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt}) - \mathcal{V}(\vec{t}_{apx}, \theta_{apx}) = O(k_{opt}\epsilon)$ . Since the optimal rigid motion can be also defined as a rotation followed by some translation, Theorem 2 holds for  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$  as well. Thus,  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt}) = \Theta(k_{opt})$  and the approximation bound follows.

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

## 5.1 An improvement for sets with small diameter

We can modify the algorithm such that its running time depends on the diameter  $\Delta$  of set  $A$ . The main idea is to convert our algorithm into one that is sensitive to the number of pairs of disks in  $A$  and  $B$  that have approximately the same distance, and then use the combinatorial bounds by Gavrilov et al. [9]. This, and a careful implementation of  $\text{ROTATION}$ , allows us to improve the analysis of the running time of  $\text{RIGIDMOTION}$  for small values of  $\Delta$ . In many applications it is reasonable to assume bounds of the type  $\Delta = O(n)$  [9], and therefore the result below is relevant; in this case, this result shows that we can compute a  $(1 - \epsilon)$ -approximation in  $O((m^2n^{5/3}/\epsilon^3) \log n \log m)$  time.

**Theorem 5.** *Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$ ,  $n \leq m$  be two sets of disjoint unit disks in the plane. Let  $\Delta$  be the diameter of  $A$ , and let  $I_{\vec{t}_{opt}, \theta_{opt}}$  be the rigid motion maximizing  $\mathcal{V}(\vec{t}, \theta)$ . For any  $\epsilon > 0$ , we can find a rigid motion  $I_{\vec{t}_{apx}, \theta_{apx}}$  such that  $\mathcal{V}(\vec{t}_{apx}, \theta_{apx}) \geq (1 - \epsilon)\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$  in  $O((m^2n^{4/3}\Delta^{1/3}/\epsilon^3) \log n \log m)$  time.*

## 6 A Monte Carlo algorithm for rigid motions

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

The algorithm is simple and follows the two-step framework of Section 5 in which an approximation of the best translation is followed by an approximation of the best rotation. However, now, the first step is a combination of grid sampling of the space of translations and random sampling of set  $A$ . This random sampling is based on the observation that the deterministic algorithm of Section 5 will compute a  $(1 - \epsilon)$ -approximation  $k_{opt}$  times, where  $k_{opt}$  is the number of pairs of overlapping disks in an optimal solution. Intuitively, the larger this number is, the quicker such a pair will be tried out in the first step. Similar observations were made by Akutsu et al. [2] who gave exact Monte Carlo algorithms for the largest common point set problem.

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

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

**Lemma 5.** *Let  $\theta_{opt}$  be the rotation that maximizes  $\mathcal{V}(\theta)$ . For any given  $\epsilon > 0$ , let  $S$  be a uniform random sample of points in  $A$  with  $|S| \geq c_1 \frac{\log m}{\epsilon^2}$  where  $c_1$  is an appropriate constant. A vertex  $\theta_{apx}$  of  $\Theta_B(S)$  of maximum depth satisfies  $\mathcal{V}(\theta_{opt}) - \mathcal{V}(\theta_{apx}) \leq \epsilon V(A)$  with probability at least  $1 - 1/m^6$ .*

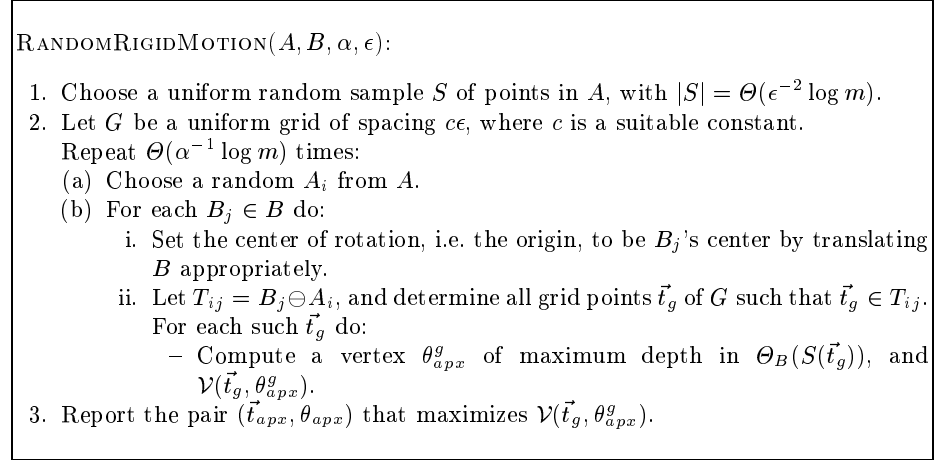
*Proof.* The proof is very similar to the proofs of Lemma 4.1 and Lemma 4.2 by Cheong et al. [6]. □

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

We could apply the idea above directly to rigid motions and compute the arrangement of all regions  $W(s)$  with respect to rigid motions of  $S$ . Lemma 5 holds for this arrangement, and a vertex of maximum depth gives an absolute error on  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$ . This arrangement has  $O(|S|^3 m^2) = O((m^2/\epsilon^6) \log^3 m)$  vertices [7] that correspond — in workspace — to combinations of triples of points in  $S$  and triples of disks in  $B$  such that each point lies on the boundary of a disk. All such possible combinations can be easily found in  $O((m^2/\epsilon^6) \log(m/\epsilon) \log^3 m)$  time. However, computing the actual rigid motion for any such combination is not trivial, as already explained in section 2. This complication is avoided by applying

the technique to rotations only, thus computing a one-dimensional arrangement instead.

*Rigid motions.* Since we assume that  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt}) \geq \alpha V(A)$ , for some given constant  $0 < \alpha \leq 1$ , we have that  $k_{opt} \geq \alpha n$ . Based also on the fact that the number of disks in  $A$  that participate in an optimal solution is at least  $k_{opt}/6$ , we can easily prove that the probability that  $\Theta(\alpha^{-1} \log m)$  uniform random draws of disks from  $A$  will all fail to give a disk participating in an optimal solution is at most  $1/m^6$ . Algorithm RANDOMRIGIDMOTION is given in Figure 5.



**Fig. 5.** Algorithm RANDOMRIGIDMOTION( $A, B, \alpha, \epsilon$ ).

**Theorem 6.** Let  $A = \{A_1, \dots, A_n\}$  and  $B = \{B_1, \dots, B_m\}$ , be two sets of disjoint unit disks in the plane and  $I_{\vec{t}_{opt}, \theta_{opt}}$  be a rigid motion that maximizes  $\mathcal{V}(\vec{t}, \theta)$ . Assume that  $\mathcal{V}(\vec{t}_{opt}, \theta_{opt}) \geq \alpha V(A)$ , for some given constant  $0 < \alpha \leq 1$ . For any given  $\epsilon > 0$ , RANDOMRIGIDMOTION( $A, B, \alpha, \epsilon$ ) computes a rigid motion  $I_{\vec{t}_{apx}, \theta_{apx}}$  such that  $\mathcal{V}(\vec{t}_{apx}, \theta_{apx}) \geq (1 - \epsilon)\mathcal{V}(\vec{t}_{opt}, \theta_{opt})$  in  $O((m^2/\epsilon^4) \log(m/\epsilon) \log^2 m)$  time. The algorithm succeeds with probability at least  $1 - 2/m^6$ .

## 7 Concluding remarks

We have presented approximation algorithms for the maximum area of overlap of two sets of disks in the plane. Theorem 2 on the lower bound on the maximum area of overlap generalizes to three dimensions in a straightforward way. The approximation algorithm for translations generalizes as well, in the following way: the arrangement of  $n$  spheres (under the assumptions (i) and (ii) of Section 1) has  $O(n)$  complexity and can be computed in  $O(n \log n)$  time [10]. In addition,

there exists a decomposition of this arrangement into  $O(n)$  simple cells that can be computed in  $O(n \log n)$  time [10]. By using these cells in the voting scheme, the running time of the algorithm is  $O((mn/\epsilon^3) \log(mn/\epsilon))$ .

Although our algorithms for rigid motions generalize to 3D, their running times increase dramatically. It would be worthwhile to study this case in detail, refine our ideas and give more efficient algorithms.

## References

1. P. K. Agarwal, S. Har-Peled, M. Sharir, and Y. Wang. Hausdorff distance under translation for points, disks, and balls. In *Proceedings of the 19th Annual ACM Symposium on Computational Geometry*, pages 282–291, 2003.
2. T. Akutsu, H. Tamaki, and T. Tokuyama. Distribution of distances and triangles in a point set and algorithms for computing the largest common point sets. *Discrete and Computational Geometry*, 20(3):307–331, 1998.
3. H. Alt, U. Fuchs, G. Rote, and G. Weber. Matching convex shapes with respect to the symmetric difference. *Algorithmica*, 21:89–103, 1998.
4. H. Alt and L. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation. In J.R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 121–153. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 1999.
5. 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.
6. O. Cheong, A. Efrat, and S. Har-Peled. On finding a guard that sees most and a shop that sells most. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms*, pages 1098–1107, 2004.
7. P. Chew, M. Goodrich, D. P. Huttenlocher, K. Kedem, J. M. Kleinberg, and D. Kravets. Geometric pattern matching under euclidean motion. *Computational Geometry: Theory and Applications*, 7:113–124, 1997.
8. M. de Berg, O. Devillers, M. van Kreveld, O. Schwarzkopf, and M. Teillaud. Computing the maximum overlap of two convex polygons under translations. *Theory Comput. Systems*, 31:613–628, 1998.
9. M. Gavrillov, P. Indyk, R. Motwani, and S. Venkatasubramanian. Combinatorial and experimental methods for approximate point pattern matching. *Algorithmica*, 38(2):59–90, 2003.
10. D. Halperin and M. H. Overmars. Spheres, molecules, and hidden surface removal. *Computational Geometry: Theory and Applications*, 11(2):83–102, 1998.
11. D. M. Mount, R. Silverman, and A. Y. Wu. On the area of overlap of translated polygons. *Computer Vision and Image Understanding*, 64:53–61, 1996.
12. J. O’Rourke and N. Badler. Decomposition of three dimensional objects into spheres. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 1(3):295–305, July 1979.
13. V. Ranjan and A. Fournier. Matching and interpolation of shapes using unions of circles. *Computer Graphics Forum*, 15(3):129–142, August 1996.