# Congruence Testing of Point Sets in 4-Space*

**Heuna Kim[1] and Günter Rote[2]**

1 **Institut für Informatik, Freie Universität Berlin, Berlin, Germany**
heunak@mi.fu-berlin.de

2 **Institut für Informatik, Freie Universität Berlin, Berlin, Germany**
rote@inf.fu-berlin.de

## Abstract

We give a deterministic $O(n \log n)$-time algorithm to decide if two $n$-point sets in 4-dimensional Euclidean space are the same up to rotations and translations. It has been conjectured that $O(n \log n)$ algorithms should exist for any fixed dimension. The best algorithms in $d$-space so far are a deterministic algorithm by Brass and Knauer (2000) [6] and a randomized Monte Carlo algorithm by Akutsu (1998) [1]. In 4-space, they take time $O(n^2 \log n)$ and $O(n^{3/2} \log n)$, respectively. Our algorithm exploits the geometric structure and the properties of 4-dimensional space, such as angles between planes, packing numbers, Hopf fibrations, and Plücker coordinates.

**1998 ACM Subject Classification** I.3.5 Computational Geometry and Object Modeling

**Keywords and phrases** Congruence Testing Algorithm, Symmetry, Computational Geometry

**Digital Object Identifier** 10.4230/LIPIcs.SoCG.2016.48

## 1 Introduction

Given two $n$-point sets $A, B \subset \mathbb{R}^4$, we want to test whether they are *congruent*, i. e., whether there exists an orthogonal matrix $R$ and a translation vector $t$ such that $RA + t = \{ Ra + t \mid a \in A \}$ equals $B$. Our main result is an optimal algorithm for this task.

▶ **Theorem 1.** *We can decide if two $n$-point sets $A$ and $B$ in 4-space are congruent in $O(n \log n)$ time and $O(n)$ space.*

**The Computational Model.** We use the Real Random-Access Machine (Real-RAM) model, which is common in Computational Geometry. We use square roots and basic operations from linear algebra and analytic geometry, such as orthonormal bases in 4 dimensions and eigenvectors of $2 \times 2$-matrices. We assume that we can compute exact results in constant time. Sines and cosines and their inverses are also used, but these operations can be eliminated in favor of purely algebraic operations. In order to test exact congruence, it is inevitable to use exact arithmetic with real numbers. If the model is restricted to rational arithmetic, the problem would be severely constrained. For example, five-fold symmetry is impossible in any dimension. The interesting and difficult inputs for congruence testing are the symmetric cases, and these instances appear only with irrational coordinates.

* This research was supported by the Deutsche Forschungsgemeinschaft within the research training group *Methods for Discrete Structures* (GRK 1408).

© Heuna Kim and Günter Rote;
licensed under Creative Commons License CC-BY



**Pruning, Condensing, and Dimension Reduction.** The conventional way of *pruning* is to classify points of $A$ according to some invariant that is simply computable and to keep only the smallest class. For example, after we prune points by the distance to the origin, we obtain points on a sphere. We generalize this principle to a more general method: We try to *condense* a finite set $A$ to a nonempty set $A' = F(A)$ of smaller size, not necessarily a subset of $A$. This should be done in an equivariant way, that is, $F(R \cdot A) = R \cdot F(A)$ for any rotation matrix $R$. For example, when we have a perfect matching of the points $A$ that has been constructed by geometric criteria, we can replace the vertex set by the midpoints of the edges. Condensing can be repeated until it gets stuck, without affecting the time and space complexity, provided that we guarantee a size reduction by a factor $c < 1$.

At first glance, condensing looks dangerous because it *throws away information*. It could introduce new symmetries, and it might happen that the condensed sets are congruent, whereas the original sets are not. Therefore, the condensed set should be kept only temporarily. The prime goal of iterative condensing steps is to reduce point sets eventually to small enough sets $A''$ and $B''$ so that we can afford *dimension reduction*.

The standard dimension reduction procedure, which we call *1+3 dimension reduction*, is as follows: Choose $a_0 \in A''$ arbitrarily and successively match it to each $b \in B''$. We fix an arbitrary rotation $R$ such that $Ra_0 = b$. Then the axis $\ell$ through $a_0 = b$ is fixed, and the problem becomes a three-dimensional problem of finding a rotation in the subspace $\ell^\perp$ perpendicular to $\ell$. At this stage, we go back to the original sets $A$ and $B$ and project them onto $\ell^\perp$. To each projected point, we attach the signed distance from $\ell^\perp$ as a label. We then look for 3-dimensional congruences that respect this labeling information. We extend this to a new technique, called *2+2 dimension reduction*, for the case when the image of a two-dimensional plane is known, see Section 7.

Pruning and condensing are very powerful and versatile methods, because we can use them with any criterion or construction that one might think of (provided that it is not too hard to compute). They allow us to concentrate on the cases where condensing makes no progress, and these cases are highly structured and symmetric. The difficulty is to pick the right pruning criteria, and to decide how to proceed when pruning and condensing gets stuck.

*Condensing and Pruning Convention.* We will mostly describe the condensing steps only for the set $A$, but, whenever we apply any condensing on $A$, we will apply the same condensing to $B$ in parallel. If $A$ and $B$ are congruent, $B$ will undergo exactly the same sequence of steps as $A$. If at any point, a difference manifests itself, $A$ and $B$ are not congruent, and we can terminate. In the case of pruning, we choose the class of the smallest cardinality. To ensure that the results of $A$ and $B$ are identical, we use some lexicographic rule for tie-breaking. We will just say that we "prune by criterion X" without explicitly mentioning a tie-breaking rule.

**Previous Work and Significance of the Results.** Congruence is a fundamental geometric problem with a long history. A lower bound of $\Omega(n \log n)$ holds already in one dimension [3, 4]. Optimal algorithms with a running time of $O(n \log n)$ were developed in two dimensions [12, 3] (first in 1976), and in three dimensions [14, 4, 2]. It is believed by many researchers in the field that the problem is solvable in $O(n \log n)$ time in any *fixed* dimension. However, so far the best algorithms in general dimension $d$ are exponential in $d$. The first approach in $d$-space by Alt, Mehlhorn, Wagener, and Welzl [2] reduces the $d$-dimensional problem to $n$ instances in dimension $d-1$, leading to a running time of $O(n^{d-2} \log n)$. Matoušek (mentioned in [1]) improved this dimension reduction approach to an $O(n^{\lfloor d/2 \rfloor} \log n)$ algorithm by matching two points at a time. He used *closest pairs*, that is, pairs of points that achieve the minimum

distance. The number of closest pairs is $O(n)$ because each point can belong to at most $O(1)$ closest pairs, by a packing argument. Brass and Knauer [6] extended this idea to triplets and obtained an $O(n^{\lceil d/3 \rceil} \log n)$ algorithm. The extension to $k$-tuples with $k > 3$ is difficult, because a special treatment is required when $k$-tuples degenerate. Akutsu [1] gave a randomized Monte Carlo algorithm that uses Matoušek's idea. Its running time, not explicitly stated in [1], is $O(n^{\lfloor d/2 \rfloor/2} \log n)$ for $d \geq 6$ and $O(n^{3/2} \log n)$ for $d = 4, 5$. These are the best algorithms for general dimensions $d$ to date.

**The new algorithm.** The techniques of closest pairs, pruning, and dimension reduction have been used for congruence testing before. We extend these ideas and apply them in a novel way. Our algorithm uses closest pairs not just as a convenient tool for matching, but it extracts helices around great circles on 3-sphere from the structure of the closest-pair graph.

Our algorithm uses 4-dimensional geometric tools: We measure angles between planes in 4-space and use Plücker coordinates, representing planes as antipodal points on the 5-sphere. As a result of symmetries, we will also encounter beautiful mathematical structures. In particular, the structure of Hopf bundles organizes the special case of isoclinic planes in a pleasant way. We will need the classification of Coxeter groups for special cases.

We have not yet succeeded in designing an $O(n \log n)$-time algorithm for an arbitrary fixed dimension. Our new algorithm for the four-dimensional case is already quite complicated and involves nontrivial mathematical properties of the 3-sphere, and therefore it is not straightforward to extend it. We view it as a major step towards a better understanding of the problem that would eventually lead to better solutions for higher dimensions.

The detailed analysis and background of the algorithm is given in the full version [11].

**Preliminaries.** In the problem, the translation vector $t$ can be eliminated by translating $A$ and $B$ so that their centroids become the origin. Furthermore, we only consider orthogonal matrices $R$ of determinant $+1$ (direct congruences). If we want to allow orthogonal matrices of determinant $-1$ (mirror congruence), we repeat the algorithm with a reflected copy of $B$.

For packing arguments, we denote by $K_d$ the kissing number on $\mathbb{S}^d$: the maximum number of equal interior-disjoint balls on the $d$-dimensional sphere $\mathbb{S}^d$ that can simultaneously touch a ball of the same size. We use the known bounds $K_2 = 5$, $K_3 = 12$, and $40 \leq K_5 \leq 44$.

We will declare the problem to be trivially solvable if the closest-pair distance $\delta$ is large, i.e., $\delta > \delta_0 := 0.0005$. This implies that the input size $|A|$ is bounded by $n_0 < 3.016 \times 10^{11}$, and hence, by 1+3 dimension reduction, the problem can be reduced to at most $n_0$ instances of 3-dimensional congruence testing, taking $O(n \log n)$ time overall.

## 2 Overview of the Algorithm

We first give an overview of our algorithm, omitting details and some special cases. Figure 1 gives a schematic view. Our goal is to apply condensing repetitively until we can apply one of the two dimension reduction techniques to the original input point sets: 1+3 dimension reduction or 2+2 dimension reduction. We can afford 1+3 dimension reduction, as described in Section 1, as soon as the input has been reduced to small size, i.e., if $|A| \leq n_0$.

Similarly, if we find a set $\mathcal{P}$ of equivariant great circles such that $1 \leq |\mathcal{P}| \leq 829$, we can trigger 2+2 dimension reduction. By applying 2+2 dimension reduction, the problem boils down to congruence testing of labeled points on a 2-dimensional torus under translations. This can be solved in time $O(n \log n)$, see Section 7.

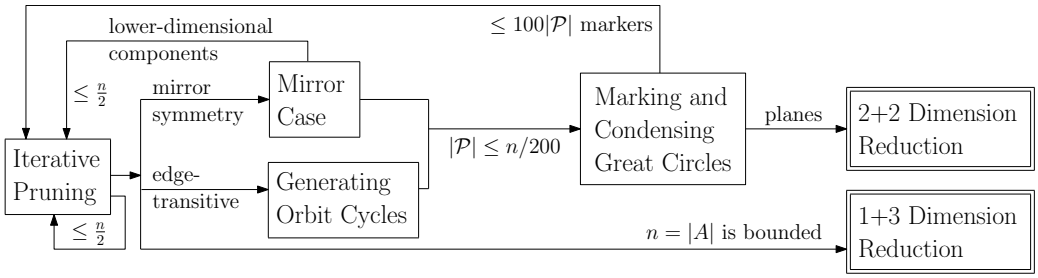


**Figure 1** The general flow of the algorithm.

The algorithm begins with pruning by distance from the origin. We may thus assume without loss of generality that the resulting set $A'$ lies on the unit 3-sphere $\mathbb{S}^3 \subset \mathbb{R}^4$.

In iterative pruning (Section 3), we first check if the minimum distance $\delta$ between points of $A'$ is bigger than the threshold $\delta_0$. If yes, we conclude that $|A'| \leq n_0$ and trigger 1+3 dimension reduction. Otherwise, we construct the closest-pair graph $G$ on $A'$. We then prune the edges of $G$ by the congruence type of their edge figures. An edge figure consists of two adjacent vertices and all their neighbors. We apply pruning with other criteria to the set $A'$ until the resulting set $A_0$ cannot be further reduced. Then all edge neighborhoods in the graph $G$ are congruent. This allows us to find either *orbit cycles* in $G$, or a subset of $A_0$ with mirror symmetry. An orbit cycle is a cyclic path $a_1a_2 \dots a_\ell a_1$ with a rotation $R$ such that $Ra_i = a_{(i \bmod \ell)+1}$; in other words, it is the orbit of the point $a_1$ under a rotation $R$.

Mirror symmetry swaps the two endpoints for each edge of $G$ and maps the whole neighborhood of the edge onto itself. If the point set $A_0$ has mirror symmetry, then by the classical classification of discrete reflection groups, $A_0$ must be related to one of the regular four-dimensional polyhedra, or $A_0$ (and $G$) is the Cartesian product of two regular polygons in orthogonal planes. The former case can be excluded, since $\delta < \delta_0$, and in the latter case, we can proceed to Section 6 for marking and condensing great circles by letting $\mathcal{P}$ as a set of circumscribing great circles of regular polygons, see Section 5.

Let us look at the case when we have found orbit cycles (Section 4). Geometrically, an orbit cycle lies on a helix around a great circle $C$. We associate this great circle $C$ to each orbit cycle and get a collection $\mathcal{P}$ of great circles on $\mathbb{S}^3$. We treat these great circles as objects in their own right, and we construct the closest-pair graph on $\mathcal{P}$. For this, we use the Plücker embedding of the corresponding planes into $\mathbb{S}^5/\mathbb{Z}_2 = \mathbb{RP}^5$, mapping each plane to a pair of antipodal points on the 5-sphere $\mathbb{S}^5$. Then we construct the closest-pair graph of planes with respect to the Euclidean distance in the Plücker embedding.

For each closest pair $(C, D)$ in $\mathcal{P}$, we look at the projection of $C$ to the plane containing $D$. Generically, this is an ellipse, and the major axis of this ellipse marks two points on $C$. The set of all markers replaces the set $A_0$. This completes a successful condensing step, and we restart and continue as before. If all projected "ellipses" turn out to be circles, we will be able to find a subfamily of great circles in a special position: they must be part of a *Hopf bundle*. The circles of a Hopf bundle can be mapped one-to-one to points on the 2-sphere. We can thus use pruning and condensing procedures for a 2-sphere in three dimensions. This yields a small set $\mathcal{P}$ of at most 12 great circles. Then we can apply the 2+2 dimension reduction technique. The details are actually more complicated, since we might have a phase in which $\mathcal{P}$ is successively condensed, see Section 6.

This concludes the summary of the algorithm. The steps of the algorithm involve several different operations: We need closest-pair graphs in 4 and 6 dimensions. The closest-pair

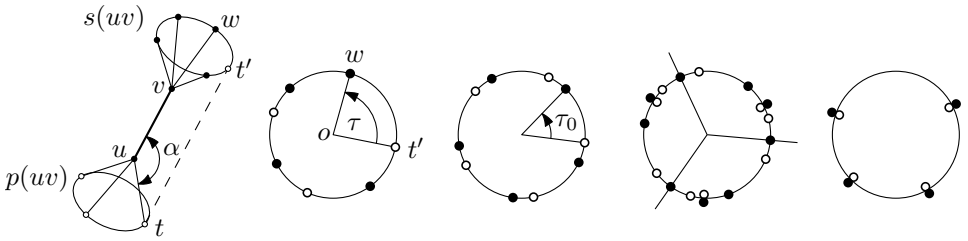


**Figure 2** (a) Predecessors and successors of $uv$ at angle $\alpha$; $t'$ is the reflected copy of $t$ on the successor circle. (b) The corresponding predecessor-successor figure, and the torsion angle $\tau(tuvw)$. Predecessors are drawn white and successors black. (c) An edge-transitive predecessor-successor figure with torsion angle $\tau_0$. (d) Canonical axes. (e) Mirror symmetry.

graph of $k$ points can be calculated in $O(k \log k)$ time in any fixed dimension, by a classical divide-and-conquer approach [5]. We also need Voronoi regions in two dimensions and convex hulls in three dimensions. Finally, we need to sort lists of numbers lexicographically. In summary, we will be able to reduce the size of the current point set $A'$ by a constant factor less than $1/2$, in $O(|A'| \log |A'|)$ time, until dimension reduction is possible.

## 3 Iterative Pruning Based on the Closest-Pair Graph: Algorithm C

After we construct the closest-pair graph $G$ for a point set $A$ on the 3-sphere, we try to condense it. We first explain the notion of a *predecessor-successor figure* in $G$.

We have a directed edge $uv$ of $G$ together with a set of *predecessor edges* $p(uv)$ incident to $u$ and a set of *successor edges* $s(uv)$ incident to $v$, as in Fig. 2a. All edges have the same length, their endpoints lie on the 3-sphere $\mathbb{S}^3$, and all predecessor and successor edges form the same angle $\alpha$ with $uv$. Then the endpoints of these edges lie on two circles. If we reflect the predecessor circle at the bisecting hyperplane of $uv$, it comes to lie on the successor circle. This results in one circle with a succinct representation of the geometric situation. see Fig. 2a. We refer to the endpoints of the predecessor and successor edges as *predecessors* and *successors*.

The following algorithm will successively prune $A$ until the closest distance $\delta$ is greater than $\delta_0$, or it will exit to either Algorithm M or Algorithm O.

**Algorithm C (Prune the closest-pair graph).** We are given a set $A \subset \mathbb{R}^4$ of $n$ points, equidistant from the origin. This set may already be the result of some previous condensing steps. Without loss of generality, we may assume that $A$ lies on the unit sphere $\mathbb{S}^3$.

**C1.** [Well-separated points?] Compute the closest distance $\delta$ between points of $A$. If $\delta > \delta_0 = 0.0005$, apply 1+3 dimension reduction. ($|A|$ is bounded by a constant.)

**C2.** [Construct the closest-pair graph.] Construct the closest-pair graph $G$, and initialize the directed graph $D$ with two opposite arcs for every edge of $G$. (This takes $O(n \log n)$ time, and the degrees in $G$ are bounded by $K_3 = 12$.)

**C3.** [Prune by degree.] If the indegrees and outdegrees in $D$ are not all the same, prune the vertices by degree, and return to C1, with the smallest class $A'$ taking the role of $A$. (Otherwise, we now enter a loop in which we try to prune arcs from $D$.)

**C4.** [Prune by directed edge figure.] The *directed edge figure* of an arc $uv \in D$ consists of $uv$ together with all arcs of $D$ out of $v$ and all arcs of $D$ that lead into $u$. If the directed

edge figures are not all congruent, *prune the arcs of D* by congruence type of directed edge figures, and return to C3. (Here we apply the pruning principle not to points but to arcs and replace the edge set $D$ by a smaller subset $D'$. Since the degrees are bounded, we can compare two directed edge figures in constant time.)

**C5.** [Mirror symmetry?] (Now all arcs have the same directed edge figure.) If the direct edge figure of $uv$ is symmetric with respect to the bisecting hyperplane of $uv$, proceed to Algorithm R (Section 5).

**C6.** [Choose an angle $\alpha$ with no mirror symmetry.] Pick an angle $\alpha$ for which the predecessors and successors are not completely symmetric, that is, the predecessor-successor figure does not look like Figure 2e.

**C7.** [Initialize Successors.] For every arc $uv \in D$, set $s(uv) := \{vw : vw \in D, \angle uvw = \alpha\}$. (The size of $s(uv)$ is bounded by $K_2 = 5$. We will now enter an inner loop in which we try to prune arcs from the sets $s(uv)$.)

**C8.** [Update predecessors.] Define the predecessor edges by $p(uv) := \{\, tu : uv \in s(tu) \,\}$. Build the predecessor-successor figure for each arc, as explained in the text above.

**C9.** [Prune by predecessor-successor figures.] If there are arcs whose predecessor-successor figures are not congruent, prune the arcs of $D$ accordingly, and return to C3.

**C10.** [Check regularity.] (Now all arcs have the same predecessor-successor figure. Each figure must contain the same number $k$ of predecessors and successors, since the total number of predecessors in all figures must balance the total number of successors.) If the predecessor-successor figure consists of two regular $k$-gons, proceed to Algorithm O for generating orbit cycles, see Section 4. (We call this the *edge-transitive* case. See Figure 2c.)

**C11.** [Prune successors by canonical axes.] Prune $s(uv)$ to a proper nonempty subset by computing *canonical axes* as explained below, and return to C8.

The canonical-axes construction in Step C11 is a well-known procedure for detecting the rotational symmetries in a planar point configuration [12, 3]. We encode the points on a circle as a cyclic string alternating between $k$ angular distances and $k$ color labels. By standard string-processing techniques, we find the lexicographically smallest cyclic reordering of this string. The starting point of this string, together with the cyclic shifts which yield the same string, gives rise to a set of $p$ equidistant rays starting from the origin (the *canonical axes*) as in Fig. 2d. These axes have the same $p$ rotational symmetries as the original configuration. We know that $p < k$ because the maximally symmetric case of two regular $k$-gons (the *edge-transitive* case shown in Fig. 2c) has been excluded in Step C10. The loop from C8–C11 maintains the following loop invariant on entry to Step C10:

*There is a position of a successor that is not occupied by a predecessor.* (1)

For the reduction in Step C11, we rotate the canonical axes counterclockwise until they hit the first position of type (1). The successors that are intersected by the canonical axes form a nonempty proper subset $s' \subseteq s(uv)$. We thus replace $s(uv)$ for each edge $uv$ by $s'$ and return to Step C8. By construction, we have made sure that (1) still holds. After pruning all successor sets, the predecessor sets are reduced accordingly in Step C8, but this cannot invalidate (1). (The invariant (1) holds on first entry to the loop because of Step C6.)

The algorithm has three nested loops (indicated by indentation) and works its way up to higher and higher orders of regularity. After C3, all vertices have the same degree. After C4, we know that all *pairs* of adjacent vertices look the same. If we exit to Algorithm O in Step C10, wee will see that certain chains of *four* points can be found everywhere.

There is the *global loop* that leads back to C1 after each successful pruning of vertices by degree. Since the size of $A$ is reduced to less than a half, we need not count the iterations of this loop. In addition, there is an outer loop that resumes working at C3 after pruning the edges of $D$, and an inner loop that starts at C8 and is repeated whenever the successor set $s(uv)$ is pruned. In these loops, we maintain that $D \neq \emptyset$ and $s(uv) \neq \emptyset$. In Step C3, if we have removed at least one edge from $D$, we will either be able to prune by degree, or the degree of all vertices has gone down by at least one. Since the degree is initially bounded by 12, Step C3 can be visited at most 12 times before exiting to C1. Similarly, Step C11 removes at least one element of $s(uv)$, so this loop is repeated at most 5 times before there is an exit to the outer loop in step C9. The most time-consuming step is the construction of the closest-pair graph in Step C2. All other operations take $O(n)$ time, not counting the exits to Algorithms M and O. Thus, the overall time is $O(|A| \log |A|)$.

## 4 Generating Orbit-Cycles: Algorithm O

We now describe how, in the edge-transitive case, the algorithm produces a set $\mathcal{P}$ of at most $|A|/200$ great circles. All predecessor-successor figures look like Fig. 2c. The *torsion angle* $\tau(tuvw)$ between a predecessor edge $tu \in p(uv)$ and a successor edge $vw \in s(uv)$ is the oriented angle $\angle(t'ow)$ in the predecessor-successor figure of $uv$. We define $\tau_0$ as the smallest counterclockwise torsion angle that appears in the predecessor-successor figure. Let $t_0u_0v_0w_0$ be a fixed quadruple with this torsion angle.

▶ **Lemma 2.**

1. *For every triple $a_1a_2a_3$ with $a_2a_3 \in s(a_1a_2)$, there is a unique cyclic sequence $a_1a_2 \dots a_\ell$ such that $a_ia_{i+1}a_{i+2}a_{i+3}$ is congruent to $t_0u_0v_0w_0$ for all $i$. (Indices are taken modulo $\ell$.)*
2. *Moreover, there is a unique rotation matrix $R$ such that $a_{i+1} = Ra_i$. In other words, $a_1a_2 \dots a_\ell$ is the orbit of $a_1$ under the rotation $R$.*
3. *The points $a_1a_2 \dots a_\ell$ do not lie on a circle.*

**Proof.** We give only a sketch; the full proof is written out in [11, Lemma 17]. It is a straightforward sequence of arguments, once we have established that the three points $t_0, u_0, v_0$ cannot lie on a great circle, and therefore a rotation is uniquely determined by the images of these three points. If the points $t_0, u_0, v_0$ would lie on a great circle, this would imply a mirror symmetry in the predecessor-successor figure, contradicting the filtering steps C5 and C6 which have led to the invariant (1). ◀

We call the cyclic paths that are constructed in Lemma 2 *orbit cycles*. The following lemma gives a bound on the number of orbit cycles in terms of $|A|$ when the closest-pair distance is small enough.

▶ **Lemma 3.** *The number of orbit cycles is at most $|A|/200$ provided that the closest distance $\delta$ is smaller than $\delta_0 = 0.0005$.*

**Proof.** We have $a_{i+1} = R_{\varphi\psi}a_i$ for all $i$, with a rotation matrix that can be written as

$$R_{\varphi\psi} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 & 0 \\ \sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & \cos\psi & -\sin\psi \\ 0 & 0 & \sin\psi & \cos\psi \end{pmatrix} \tag{2}$$

in an appropriate basis $x_1y_1x_2y_2$. We cannot have $\varphi = 0$ or $\psi = 0$, because otherwise the orbit would form a regular polygon $a_1a_2a_3a_4 \dots a_\ell$ in a plane, contradicting Lemma 2.3.

Thus, we know that $\varphi, \psi \neq 0$. To get a closed loop, we must have $|\varphi|, |\psi| \geq 2\pi/\ell$. If the projection of $a_i$ to the $x_1y_1$-plane has norm $r_1$ and the projection to the $x_2y_2$-plane has norm $r_2$ with $r_1^2 + r_2^2 = 1$, then the squared distance is

$$\delta^2 = \|a_{i+1} - a_i\|^2 = (2r_1 \sin \tfrac{|\varphi|}{2})^2 + (2r_2 \sin \tfrac{|\psi|}{2})^2 \geq 4 \sin^2(\pi/\ell).$$

Thus, if $\delta \leq 2\sin(\pi/12000) \approx 0.000523$, it is guaranteed that $\ell \geq 12000$, that is, every orbit cycle contains at least 12000 points. On the other hand, each point $u \in A$ belongs to a bounded number of orbit cycles: it has at most $K_3 = 12$ incoming arcs $tu$, and each each arc $tu$ has at most $K_2 = 5$ successor edges $uv \in s(tu)$. The triple $tuv$ specifies a unique orbit cycle, and thus there are at most $12 \times 5 = 60$ orbit cycles through $u$. The total number of orbit cycles is therefore at most $|A| \cdot 60/12000 = |A|/200$. ◀

For each orbit cycle, we can find an appropriate rotation matrix $R_{\varphi,\psi}$. If $\varphi = \pm\psi$, then the orbit of any point under this rotation lies on a great circle, contradicting Lemma 2.3. Thus, we get a unique invariant plane that rotates by the smaller angle in itself. We replace each orbit cycle by the great circle in this invariant plane, yielding the desired set $\mathcal{P}$ of great circles with $|\mathcal{P}| \leq |A|/200$.

## 5 The Mirror Case: Algorithm R

**Algorithm R (Treat mirror-symmetric closest-pair graphs).** We are given a nonempty directed subgraph $D$ of the closest-pair graph on a point set $A \subset \mathbb{S}^3$ with closest-pair distance $\delta \leq \delta_0$. All directed edges in $D$ have equal edge-figures and exhibit perfect mirror-symmetry, as established in Steps C4 and C5 of Algorithm C. Algorithm R will produce, in an equivariant way, either

**(a)** a set $A'$ of at most $|A|/2$ points, or
**(b)** a set $\mathcal{P}$ of at most $|A|/200$ great circles.

**R1.** [Make $D$ undirected.] Construct the undirected version of $D$ and call it $G$.
**R2.** [Check for eccentric centers of mass.] Compute the center of mass of each connected component of $G$. If these centers are not in the origin, return the set $A'$ of these centers.
**R3.** [Two-dimensional components?] If each component is a regular polygon with center at the origin, return the set $\mathcal{P}$ of circumcircles of each polygon.
**R4.** [Three-dimensional components?] If each component spans a 3-dimensional hyperplane $H$, replace the component by two antipodal points perpendicular to $H$. Return the set $A'$ of these points.
**R5.** [Toroidal grid.] Now each component of $D$ is the product $P \times Q$ of a regular $p$-gon $P$ and a regular $q$-gon $Q$ in orthogonal planes, with $p, q \geq 3$. Pick a vertex $u$ from each component. There are four incident edges, and the plane spanned by two incident edges is orthogonal to the plane spanned by the other two edges. Represent the component of $G$ by these two orthogonal planes shifted to the origin. Return the set $\mathcal{P}$ of great circles in these planes (two per component).

The algorithm takes linear time. We still need to show that these cases are exhaustive. After we make $D$ undirected in Step R1, for every edge $uv$ of $G$, the bisecting hyperplane of $uv$ acts as a mirror that exchanges $u$ and $v$ together with their neighborhoods. Since the reflected mirror hyperplanes act again as mirrors, it follows that every component of the graph is the orbit of some point $u_0$ under the group generated by reflections perpendicular to the edges incident to $u_0$. Thus, the incident edges of a single point determine the geometry of the whole component. The graph may have several components, all of which are congruent.

The discrete groups generated by reflections (finite Coxeter groups) have been classified in any dimension, cf. [7, Table IV on p. 297]. In four dimensions, they were first enumerated by Édouard Goursat in 1899: There are five *irreducible groups*, which are the symmetry groups of the five regular 4-dimensional polytopes, and the *reducible* groups, which are a direct product of lower-dimensional reflection groups. There are infinitely-many reducible groups of the form $D_2^p \times D_2^q$, where $D_2^p$ is the dihedral group of order $2p$, the symmetry group of the regular $p$-gon. These groups are treated in Step R5. Except this infinite family, there are only eight other groups, which can be excluded by the following lemma.

▶ **Lemma 4.** *If every component of $G$ is the orbit of a point under a four-dimensional symmetry group which is not $D_2^p \times D_2^q$, then the minimum distance $\delta$ is at least* 0.07*, and therefore bigger than $\delta_0$.*

**Proof.** We analyze each of the eight groups case by case. The arrangement of all mirror hyperplanes of a reflection group cuts the 3-sphere into equal cells, which can be taken as the *fundamental regions* of the group. (These fundamental regions are not necessarily equal to the Voronoi regions of the point set; the Voronoi regions are usually cut into smaller cells by mirrors passing through the centers of the regions.) It is known that the fundamental region of a reflection group is a spherical *simplex*, by a theorem of Cartan (1928), see [7, Theorem 11.23]. Thus, in 4-space, the fundamental region is a spherical tetrahedron $T$. We must now consider all possibilities how the orbit of a point $u_0 \in T$ might give rise to a component of $G$. We have to place $u_0$ on some facets $F$ of $T$ and equidistant from the remaining facets of $T$. If the point is not chosen equidistant from the other faces, the closest-pair graph would not contain edges that generate all four mirrors. The minimum distance is achieved when $u_0$ lies in the interior of $T$ and $F = \emptyset$. This value is approximately 0.0782. See [11, Section 10.1] for more details. ◀

We go through the steps one by one to check if the algorithm achieves the claimed results. Step R2 treats the case when a (lower-dimensional) component does not go through the origin. This includes the cases when $G$ is a matching or a union of "small" regular polygons. Every component contains at least 2 points, and thus $|A'| \le |A|/2$. Steps R3 and R4 treat the two- and three-dimensional components that are centered at the origin. (The one-dimensional case of a matching cannot be centered at the origin, because then we would have $\delta = 2$.) If we have a regular $k$-gon inscribed in a great circle (Step R3), from $\delta < \delta_0$, $k \ge 12000$. Thus $|C| \le |A|/12000 \le |A|/200$. A three-dimensional component (Step R4) contains at least four points and is reduced to two antipodal points. Again we have $|A'| \le |A|/2$.

Steps R2–R4 have treated all cases of Coxeter groups which are not full-dimensional, and Lemma 4 excludes all full-dimensional groups which are not of the form $D_2^p \times D_2^q$. Thus, in Step R5, each component of $D$ is the product $P \times Q$ of a regular $p$-gon $P$ and a regular $q$-gon $Q$ in orthogonal planes, with $p, q \ge 3$. Such a component forms a toroidal $p \times q$ grid. Each vertex has four neighbors. The two polygons $P$ and $Q$ have equal side lengths $\delta$, because otherwise the four neighbors would not be part of the closest-pair graph $G$. The two incident edges of a vertex $u$ that come from $P$ are orthogonal to the two edges that come from $Q$, so we can distinguish the two edge classes. (The case when all four edges are perpendicular is the 4-cube with $p = q = 4$ and with reflection group $D_2^4 \times D_2^4$, which has 16 vertices and minimum distance $\delta = 1$ and is therefore excluded.) All copies of $P$ in the grid lie in parallel planes, and so do the copies of $Q$. Accordingly, Step R5 represents each connected component by two orthogonal planes through the origin. We need to show that the component is large. As $P \times Q$ lies on the unit 3-sphere, the circumradii $r_P$ and $r_Q$ of the two polygons satisfies $r_P^2 + r_Q^2 = 1$. Thus, the larger circumradius, let us say $r_P$, is at least $1/\sqrt{2}$. Since the

closest-pair distance is $\delta = 2r_P \sin\frac{\pi}{p} \leq \delta_0 = 0.0005$, we get $\sqrt{2}\sin\frac{\pi}{p} \leq \delta_0$, which implies $p \geq 8886$. Each component contains $pq \geq p$ points and is reduced to two circles. Thus, the algorithm achieves the claimed reduction.

## 6 Marking and Condensing of Great Circles: Algorithm M

We have extracted a set $\mathcal{P}$ of at most $|A|/200$ great circles from the point set $A$, either from the mirror case (Algorithm R in Section 5) or from orbit cycles (Algorithm O in Section 4). Algorithm M obtains a small set $A'$ of *marker points* on each circle so that we can resume Algorithm C, or it exits to Algorithm T for 2+2 dimension reduction (Section 7). For this purpose, we look at pairs of circles $C, D \in \mathcal{P}$ and the angle between the 2-planes $P, Q$ in which they lie. The angle of two planes $P, Q$ in 4-space consists of *two* numbers $\alpha, \beta$, with $0 \leq \alpha, \beta \leq \pi/2$ [13]. One way to define the angle is by the orthogonal projection $C'$ of the unit circle $C$ in plane $P$ onto $Q$. This projection is an ellipse, and its axes are $\cos\alpha$ and $\cos\beta$. If $\alpha \neq \beta$, we can use the major axis of the projection ellipse $C'$ to mark the two points of $D$ on this axis, which are closest points on $D$ to $C$. We similarly mark two points on $C$.

Doing this for all pairs of circles would lead to a quadratic blowup. Therefore we construct the closest-pair graph on the *set of circles*. We use Plücker coordinates to map great circles of $\mathbb{S}^3$ to points on $\mathbb{S}^5$. We can then compute the closest-pair graph in six dimensions, and every circle has at most $K_5$ closest neighbors.

For a plane $P$ spanned by two vectors $u, v$, Plücker coordinates are the sixtuples of the $2 \times 2$ determinants of the $2 \times 4$ matrix with rows $u$ and $v$. Plücker coordinates are usually regarded as points of projective 5-space: they are unique up to scaling. We thus normalize them, and represent every plane as a pair of antipodal points on the sphere $\mathbb{S}^5$. We define the *Plücker distance* between two planes as the smallest Euclidean distance among any two of the four representative points in $\mathbb{R}^6$. The following lemma shows that this distance is geometrically meaningful and does not depend on the choice of a coordinate system.

▶ **Lemma 5.**
1. *Two planes $P, Q$ with angles $\alpha, \beta$ have Plücker distance $\sqrt{2(1 - \cos\alpha\cos\beta)}$.*
2. *A circle can have at most $K_5$ closest neighbors on the Plücker sphere $\mathbb{S}^5$.*

**Proof.** Property 1 is a lengthy calculation [11, Lemma 4]. Property 2 is straightforward. ◀

The above approach fails for planes with $\alpha = \beta$. Such planes or circles are called *isoclinic*. They come in two variations, left-isoclinic and right-isoclinic, which are distinguished as follows: Let $v_1, v_2$ be a basis of $P$, and let $v_1', v_2'$ be the projection of $v_1, v_2$ to $Q$. Then $P$ and $Q$ are right-isoclinic if the vectors $v_1, v_2, v_1', v_2'$ form a positively oriented basis of $\mathbb{R}^4$. This classification as left or right is called the *chirality* of a pair of isoclinic planes. When $\alpha = \beta = \pi/2$ (completely orthogonal planes) or $\alpha = \beta = 0$ (identical planes), the two planes are both left-isoclinic and right-isoclinic.

Isoclinic pairs manifest a very rigid structure on a sphere. The following proposition summarizes some beautiful properties of such pairs. They are formulated for right-isoclinic circles, but they hold equally with left and right exchanged.

▶ **Proposition 6.**
1. *The relation of being right-isoclinic is transitive (as well as reflexive and symmetric). An equivalence class is called a* right Hopf bundle.
2. *For each right Hopf bundle, there is a* right Hopf map *$h$ that maps the circles of this bundle to points on $\mathbb{S}^2$.*

3. *By this map, two isoclinic circles with angle $\alpha, \alpha$ (and with Euclidean distance $\sqrt{2}\sin\alpha$ on the Plücker sphere $\mathbb{S}^5$ ) are mapped to points at angular distance $2\alpha$ on $\mathbb{S}^2$.*
4. *A circle can have at most $K_2 = 5$ closest neighbors on the Plücker sphere $\mathbb{S}^5$ that are right-isoclinic.*

**Proof Sketch.** Transitivity (Property 1) and the preservation of distances (Property 3) can be established by calculations, see [11, Corollary 11 and Lemma 12]. The right Hopf map in Property 2 is obtained as follows: Choose a positively oriented coordinate system $(x_1, y_1, x_2, y_2)$ for which some circle $C_0$ of the bundle lies in the $x_1y_1$-plane. Then the map $h\colon \mathbb{S}^3 \to \mathbb{S}^2$ defined by $h(x_1, y_1, x_2, y_2) = \big(2(x_1y_2 - y_1x_2),\ 2(x_1x_2 + y_1y_2),\ 1 - 2(x_2^2 + y_2^2)\big)$ maps all points on a circle of the bundle to the same point on $\mathbb{S}^2$. A different choice of $C_0$ would lead to a different map, but by Property 3, the images are related by an isometry of $\mathbb{S}^2$. Property 4 is a direct consequence of Properties 2 and 3. ◀

We need an auxiliary Algorithm K, defined later, for condensing on the 2-sphere:

▶ **Lemma 7.** *Algorithm K condenses a nonempty set $F$ of points on the 2-sphere $\mathbb{S}^2$ in an equivariant way to a nonempty set $F'$ of at most $\min\{12, |F|\}$ representative points on $\mathbb{S}^2$, in $O(|F| \log |F|)$ time. These points are either the vertices of a regular tetrahedron, a regular octahedron, a regular icosahedron, a single point, or a pair of antipodal points.*

**Algorithm M (Mark and condense great circles).** Given a set $\mathcal{P}$ of great circles on $\mathbb{S}^3$, this algorithm will produce, in an equivariant way, either a nonempty set $A'$ of at most $100|\mathcal{P}|$ points on $\mathbb{S}^3$, or a nonempty set $\mathcal{P}'$ of at most 829 great circles.

The algorithm will update the set $\mathcal{P}$ and maintain an equivalence relation $\sim$ on $\mathcal{P}$ such that all circles in the same equivalence class belong to a common (left or right) Hopf bundle. The common chirality of all bundles is indicated by a variable *chirality* $\in \{None, Left, Right\}$, where *None* is chosen when the equivalence relation is trivial and all classes are singletons. The size of the equivalence classes is bounded by 12.

**M1.** [Initialize.] Let every circle $C \in \mathcal{P}$ form a singleton equivalence class.
**M2.** [Prune by size.] If equivalence classes are of different sizes, choose the size that occurs least frequently. Throw away all equivalence classes that are not of this size, together with the circles they contain.
**M3.** [Few circles?] If $|\mathcal{P}| \le 829$, return the set $\mathcal{P}$.
**M4.** [Singletons?] If all equivalence classes are singletons, set *chirality* := *None*.
**M5.** [Construct closest pairs.] Represent each circle $C \in \mathcal{P}$ by two antipodal points on $\mathbb{S}^5$, using Plücker coordinates. Construct the closest-pair graph $H$ on $\mathcal{P}$ with respect to the distances on $\mathbb{S}^5$.
**M6.** [Classify edges.] Partition the edges of $H$ into $E_L \cup E_R \cup E_N$, representing pairs of circles that are left-isoclinic, right-isoclinic, and not isoclinic.
**M7.** [Use non-isoclinic edges.] If $E_N \neq \emptyset$, let $\mathcal{N} := E_N$ and go to Step M12.
**M8.** [Find non-isoclinic pairs from equivalent circles.] If $E_L \neq \emptyset$ and *chirality* = *Right*, set $\mathcal{N} := \big\{\{C', D\} \mid \{C, D\} \in E_L,\ C' \text{ is closest to } C \text{ among the circles } C' \sim C\ \big\}$, and go to Step M12.
**M9.** (Same as M8, with left and right exchanged.)
**M10.** [Merge classes.] If $E_L \neq \emptyset$ and *chirality* $\in \{Left, None\}$, merge equivalence classes that contain circles that are connected in $E_L$. Use Algorithm K to condense each resulting class $F$ to a set $F'$ of at most 12 *representative* circles. Set $\mathcal{P}$ to the set of all representatives

circles, and put them into the same equivalence class if and only if they come from the same set $F'$. Set *chirality* := *Left*, and return to M2.

**M11.** (Same as M10, with left and right exchanged. Since all possibilities are exhausted, the only remaining possibility in this step is to merge classes and return to M2.)

**M12.** [Mark points on circles.] (Each pair in $\mathcal{N}$ is now a non-isoclinic pair of circles.) For each pair of circles $\{C, D\} \in \mathcal{N}$, mark the two points on $C$ that are closest to $D$, and likewise, mark two points on $D$. Return the set $A'$ of all marked points.

The crucial properties for the correctness and the running time are formulated in a lemma, implying that the algorithm produces indeed at most $100|\mathcal{P}|$ points, 4 for each pair in $\mathcal{N}$.

▶ **Lemma 8.**

1. *After the algorithm returns to Step M2 from step M10 or M11, the number of equivalence classes is reduced at least by half.*
2. *Algorithm M terminates in $O(|\mathcal{P}| \log |\mathcal{P}|)$ time.*
3. *In Step M12, $\mathcal{N}$ is a nonempty set of at most $25|\mathcal{P}|$ non-isoclinic pairs.*

**Proof.**

1. The only possibility for the algorithm to stall is that the edges of $H$ are *within* classes, and no merging takes place in Step M10 or M11. Each class must be one of the five configurations listed in Lemma 7, and the smallest possible angular distance $\delta_{\text{ico}} \approx 1.107$ occurs for two adjacent vertices of the icosahedron. Translating this to the distance on $\mathbb{S}^5$ by Lemma 3, we get that the closest-pair distance is at least $\delta_{\min} \approx 0.7435$, and a volume packing argument on $\mathbb{S}^5$ yields that there can be at most 829 circles with this distance, see [11, Section 11.3]. Then the algorithm exits in Step M3.
2. The most expensive step is the construction of the closest-pair graph in Step M5, which takes $O(|\mathcal{P}| \log |\mathcal{P}|)$ time. The algorithm contains one loop, when returning from M10 or M11 to M2. Since the number of equivalence classes decreases geometrically and each class contains at most 12 circles, the overall time is also bounded by $O(|\mathcal{P}| \log |\mathcal{P}|)$.
3. When $\mathcal{N}$ is constructed in Step M7, there can be at most $22|\mathcal{P}|$ such pairs, because the degree in $H$ is bounded by $K_5 \leq 44$. In Step M8, the constructed set $\mathcal{N}$ is nonempty: Every pair $\{C, D\} \in E_L$ produces at least one element of $\mathcal{N}$, since all equivalence classes contain at least two elements, by M2 and M4. When a pair $\{C', D\}$ in $\mathcal{N}$ is formed, we have a left-isoclinic pair $\{C, D\}$ and a right-isoclinic pair $\{C, C'\}$. If $\{C', D\}$ were also isoclinic, we would get a contradiction to transitivity (Lemma 1). Each circle $C \in \mathcal{P}$ gives rise to at most $5 \cdot 5 = 25$ pairs, since a circle can have at most 5 isoclinic neighbors in $H$ by Lemma 4. ◀

**Constructing a Canonical Set on the 2-Sphere: Algorithm K.** This algorithm is a variant of a standard 3-dimensional congruence testing algorithm [4]. It condenses a set $F \subset \mathbb{S}^2$ to a nonempty set $F' \subset \mathbb{S}^2$ of at most $\min\{12, |F|\}$ representative points, in $O(|F| \log |F|)$ time.

We start by computing the convex hull of $F$. If it does not contain the origin, we output the vector pointing to the centroid of the points as a representative. If the hull is one- or two-dimensional, we get two antipodal points as canonical representatives. We can thus assume that the hull is a three-dimensional polytope. Now we prune by vertex degree, prune and condense by face degree (replacing the point set by the centroids of some faces), and prune and condense by edge length (replacing the point set by the midpoints of some edges). This is repeated until the vertices form a regular tetrahedron, octahedron, or icosahedron. The details are given in [11, Section 20 ]. The most expensive part is the computation of the convex hull, and each condensing reduces the size by a constant factor. Thus, the overall running time is $O(|F| \log |F|)$.

## 7 2+2 Dimension Reduction: Algorithm T

▶ **Theorem 9.** *Given two sets $A, B$ of $n$ points, and two planes $P$ and $Q$, it can be checked in $O(n \log n)$ time if there exists a rotation that maps $A$ to $B$, and $P$ to $Q$.*

**Proof.** We begin with choosing a coordinate system $(x_1, y_1, x_2, y_2)$ for $A$ so that $P$ becomes the $x_1y_1$-plane, and similarly for $B$ and $Q$. We then look for rotations $R$ that leave the $x_1y_1$-plane invariant. Such rotations have the form $R = \left(\begin{smallmatrix} R_1 & 0 \\ 0 & R_2 \end{smallmatrix}\right)$ with two orthogonal $2 \times 2$ matrices $R_1$ and $R_2$. Since $\det R = \det R_1 \cdot \det R_2 = 1$, we try two cases: (a) $R_1$ and $R_2$ are planar rotations; (b) $R_1$ and $R_2$ are planar reflections. We can reduce (b) to (a) by applying the rotation $\left(\begin{smallmatrix} 0&1&0&0 \\ 1&0&0&0 \\ 0&0&0&1 \\ 0&0&1&0 \end{smallmatrix}\right)$ to $A$. Thus, it suffices to describe (a), where $R$ has the form $R_{\varphi\psi}$ in (2), i. e., a combination of a rotation by $\varphi$ in the $x_1y_1$-plane and an independent rotation by $\psi$ in the $x_2y_2$-plane. We use polar coordinates in these two planes by setting

$$\begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} r_1 \cos\alpha_1 \\ r_1 \sin\alpha_1 \\ r_2 \cos\alpha_2 \\ r_2 \sin\alpha_2 \end{pmatrix} \qquad \text{with } r_1 = \sqrt{x_1^2 + y_1^2} \text{ and } r_2 = \sqrt{x_2^2 + y_2^2}.$$

The rotation $R_{\varphi\psi}$ changes $(\alpha_1, \alpha_2)$ by adding $(\varphi, \psi)$ modulo $2\pi$, and leaves $r_1$ and $r_2$ unchanged. In other words, $R_{\varphi\psi}$ acts as a translation on the torus $\mathbb{T}^2 = [0, 2\pi) \times [0, 2\pi)$. We attach the distances $(r_1, r_2)$ to each point $(\alpha_1, \alpha_2) \in \mathbb{T}^2$ as a *label*. The problem becomes therefore a *two-dimensional problem* of testing *congruence under translation* for *labeled* points on a torus. We denote the two labeled point sets as $\tilde{A}$ and $\tilde{B}$, and a point of $\tilde{A}$ can only be mapped to a point of $\tilde{B}$ with the same label.

Points in the $x_1y_1$-plane should be considered separately, because $\alpha_2$ is not unique when $r_2 = 0$, and the same problem occurs for the points in the $x_2y_2$-plane. We will defer the treatment of these points to the end of this section, and start with other points first.

We now give an algorithm for the following problem: given two labeled point sets $\tilde{A}$ and $\tilde{B}$ on the torus $\mathbb{T}^2$, test if $\tilde{A}$ and $\tilde{B}$ are the same up to translations. We will find a canonical set of $\tilde{A}$ (and $\tilde{B}$) which is similar to a condensed set. In contrast to a condensed set, we add no new symmetries to a canonical set, by updating labels to preserve complete information. Let $\mathrm{Sym}(A)$ for a set $A \subset \mathbb{T}^2$ denote translational symmetry group of $A$, i.e., the set of translations that map $A$ to itself and preserve labels, if $A$ is a labeled set.

**Algorithm T (Reduce a labeled point set on the torus to a canonical set).** The input is a labeled point set $\tilde{A}$ on the torus $\mathbb{T}^2$. We assume that two labels can be compared in constant time. The output is an unlabeled *canonical set* $\hat{A}$ with the following two properties:

**(a)** $\mathrm{Sym}\,\hat{A} = \mathrm{Sym}\,\tilde{A}$.

**(b)** The group $\mathrm{Sym}\,\hat{A}$ acts transitively on $\hat{A}$: For any two points $x, y \in \hat{A}$, the translation from $x$ to $y$ leaves $A$ invariant.

In addition, $\hat{A}$ should be obtained from $\tilde{A}$ without making any arbitrary decisions.

**T1.** [Prune by labels.] Choose the label that occurs least frequently in $\tilde{A}$, and let $A'$ be the set of points with this label. (For a while we will now do ordinary pruning, using only the geometry of the point set $A'$.)

**T2.** [Compute the Voronoi diagram.] Compute the Voronoi diagram of $A'$ on the torus $\mathbb{T}^2$. This can be reduced to a Voronoi diagram in the plane by replicating the square region representing the torus together with the set $A'$ 9 times in a $3 \times 3$ tiling, see for

example [9]. Clipping the result to the central tile yields the Voronoi diagram on the torus in $O(|A'| \log |A'|)$ total time.

**T3.** [Prune by shape.] Translate each point $a \in A'$ to the origin together with its Voronoi cell. If the translated cells are not all equal, replace $A'$ by the subset of points whose cell shape occurs least frequently, and return to T2.

**T4.** [Restore information from the original set $\tilde{A}$ by labeling the points in $A'$.] (Now all Voronoi cells are translated copies of the same hexagon or rectangle.) Assign each point of $\tilde{A}$ to its Voronoi cell. A point on the boundary is assigned to all incident cells. Now for each Voronoi site $x \in A'$, collect the points in its cell and translate them so that $x$ lies at the origin. Represent each point as a triple ($\varphi$-coordinate, $\psi$-coordinate, label). Concatenate these triples in lexicographic order into a string of numbers that represents the cell contents, and attach the string of each cell as a label to the point $x$. (This string representation is obtained equivariantly, since two points $x, y \in A'$ get the same string if and only if the two Voronoi cells are exact translated copies of each other, including all points in the cells with their original labels. We have thus preserved complete information about the symmetries of $\tilde{A}$.)

**T5.** [Compress labels.] Sort the label strings and replace each label with its rank in sorted order.

**T6.** [Finalize.] If there are at least two labels, return to T1. Otherwise, return the set of points $A'$, without labels, as the canonical set $\hat{A}$.

▶ **Lemma 10.** *Algorithm T computes a canonical set $\hat{A}$ of a labeled set $\tilde{A}$ on the torus in time $O(|\tilde{A}| \log |\tilde{A}|)$.*

**Proof.** We first check that $\hat{A}$ has the claimed properties. Property (a) consists of two inclusions: We have $\mathrm{Sym}(\tilde{A}) \subseteq \mathrm{Sym}(\hat{A})$, because $\hat{A}$ is obtained in a equivariant way from $\tilde{A}$. None of the operations which are applied to $\tilde{A}$ to obtain $\hat{A}$ destroys any translational symmetries. The other inclusion $\mathrm{Sym}(\hat{A}) \subseteq \mathrm{Sym}(\tilde{A})$ is ensured by Step T4. (This would work for any set $A'$.) To see property (b), note that the Voronoi cell of a point fixes the relative positions of its neighbors. Starting from any point $a \in \hat{A}$ we can reconstruct the whole set $\hat{A}$ if we know the shape of each Voronoi cell. Step T3 ensures that all Voronoi cells are equal, and therefore the reconstructed set $\hat{A} - a$ is the same, no matter from which point $a$ we start.

Let us analyze the running time. Each iteration of the loop T2–T3 takes $O(|A'| \log |A'|)$ time and reduces the size of $A'$ to half or less. Thus, the total running time of this loop is $O(|\tilde{A}| \log |\tilde{A}|)$, since the initial size of $A'$ is bounded by the size of $\tilde{A}$.

Steps T4 and T5 involve point location in Voronoi diagrams and sorting of strings of numbers of total length $O(|\tilde{A}|)$. These operations can be carried out in $O(|\tilde{A}| \log |\tilde{A}|)$ time. Thus, each iteration of the whole loop T1–T6 takes $O(|\tilde{A}| \log |\tilde{A}|)$ time. Step T1 reduces the size of $\tilde{A}$ to a half or less after each iteration. Thus, the total running time is $O(|\tilde{A}| \log |\tilde{A}|)$. ◀

We use this procedure for comparing two labeled sets $\tilde{A}$ and $\tilde{B}$ on the torus as follows. We run Algorithm T in parallel for $\tilde{A}$ and $\tilde{B}$. We must make sure that all steps run identically. In particular, the sorted strings in Step T5 must be the same for $\tilde{A}$ and $\tilde{B}$. If the algorithm runs to the end without finding a difference between $\tilde{A}$ and $\tilde{B}$, and if the Voronoi cells of the canonical sets $\hat{A}$ and $\hat{B}$ are equal, we know that $\tilde{A}$ and $\tilde{B}$ are congruent by translation.

We still have to deal with points on coordinate planes. Let $A_1 \subseteq A$ be the points in the $x_1y_1$-plane, and let $A_2 \subseteq A$ be the points in the $x_2y_2$-plane. If $A_1 \neq \emptyset$, we compute the canonical axes of $A_1$, as described for Step C11 in Section 3. They form $k \geq 1$ equally spaced

angular directions $\bar{\alpha} + j\frac{2\pi}{k}$ modulo $2\pi$, $j \in \mathbb{Z}$. We know that $R$ must map the canonical axes of $A_1$ to the canonical axes for the corresponding set $B_1$. We incorporate this restriction into an additional label for each point $u \in A \setminus A_1 \setminus A_2$. If $u$ has a polar angle $\alpha_1$ in the $x_1y_1$-plane, we attach to it the difference to the nearest smaller canonical angle:

$$\min\{\, \alpha_1 - (\bar{\alpha} + j\tfrac{2\pi}{k}) \mid j \in \mathbb{Z},\ \alpha_1 - (\bar{\alpha} + j\tfrac{2\pi}{k}) \geq 0 \,\}$$

We also have to test if $A_1$ and $B_1$ are congruent. If $A_2 \neq \emptyset$, we treat this in the same way and attach another additional label to the points in $A \setminus A_1 \setminus A_2$. (In fact, Algorithm T is an extension of the canonical axes construction from the one-dimensional torus $\mathbb{S}^1$ to the two-dimensional torus $\mathbb{T}^2$.) ◀

## 8 Concluding Remarks

The running times of the algorithms for general dimensions are improved by incorporating our algorithm when the dimension is reduced to 4. The deterministic algorithm of Brass and Knauer runs now in $O(n^{\lceil (d-1)/3 \rceil} \log n)$ time, and the randomized algorithm of Akutsu takes time $O(n^{\lfloor (d-1)/2 \rfloor/2} \log n)$ for $d \geq 9$ and $O(n^{3/2} \log n)$ for $d \leq 8$. It is likely that our algorithm can be simplified, and the constants in the bounds are certainly too pessimistic.

**Approximate Congruence Testing.** It makes sense to test for congruence with an error tolerance $\varepsilon$, but this problem is known to be NP-hard even in the plane [8, 10]. However, the problem becomes polynomial if the input points are sufficiently separated compared to $\varepsilon$. We are confident that our algorithm, when implemented with standard floating-point arithmetic and with appropriate error margins to shield equality tests from round-off errors, would decide approximate congruence in the following weak sense. Given a tolerance $\varepsilon$ that is large compared to the machine precision, and two sets $A$ and $B$ whose points are separated by more than, say $10\varepsilon$, the algorithm would correctly distinguish the instances that are congruent with a tolerance of $\varepsilon$ from the instances that are not even congruent with a tolerance of, say $100\varepsilon$. Between $\varepsilon$ and $100\varepsilon$, the algorithm is allowed to make errors. Such a result will require a thorough analysis of the numerical stability of the operations in our algorithm.

**Regularity.** The general theme in our algorithm is whether *local* regularity implies *global* regularity; in other words, when sufficiently large neighborhoods of all points look the same, does a symmetry group have to act transitively on the point set? Our techniques might also shed light on symmetries on the 3-sphere, for example the classification of finite subgroups of the orthogonal group $O(4)$ of $4 \times 4$ orthogonal matrices, or on regular and semiregular tilings of $\mathbb{S}^3$.

## References


1. Tatsuya Akutsu. On determining the congruence of point sets in $d$ dimensions. *Computational Geometry: Theory and Applications*, 4(9):247–256, 1998.
2. Helmut Alt, Kurt Mehlhorn, Hubert Wagener, and Emo Welzl. Congruence, similarity, and symmetries of geometric objects. *Discrete & Computational Geometry*, 3(1):237–256, 1988.
3. M. J. Atallah. On symmetry detection. *IEEE Trans. Computers*, 100(7):663–666, 1985.
4. M. D. Atkinson. An optimal algorithm for geometrical congruence. *Journal of Algorithms*, 8(2):159–172, 1987.

5 Jon Louis Bentley and Michael Ian Shamos. Divide-and-conquer in multidimensional space. In *Proc. 8th Ann. ACM Symp. Theory of Computing (STOC)*, pages 220–230. ACM, 1976.

6 Peter Brass and Christian Knauer. Testing the congruence of $d$-dimensional point sets. *International Journal of Computational Geometry and Applications*, 12(1–2):115–124, 2002.

7 Harold S. M. Coxeter. *Regular Polytopes*. Dover Publications, 3rd edition, 1973.

8 C. Dieckmann. *Approximate Symmetries of Point Patterns*. PhD thesis, FU Berlin, 2012.

9 N. P. Dolbilin and D. H. Huson. Periodic Delone tilings. *Per. Math. Hung.*, 34:57–64, 1997.

10 Sebastian Iwanowski. Testing approximate symmetry in the plane is NP-hard. *Theoretical Computer Science*, 80(2):227–262, 1991.

11 H. Kim and G. Rote. Congruence testing of point sets in 4 dimensions. arXiv:1603.07269.

12 Glenn Manacher. An application of pattern matching to a problem in geometrical complexity. *Information Processing Letters*, 5(1):6–7, 1976.

13 Henry P. Manning. *Geometry of Four Dimensions*. Macmillan, 1914.

14 Kōkichi Sugihara. An $n \log n$ algorithm for determining the congruity of polyhedra. *Journal of Computer and System Sciences*, 29(1):36–47, 1984.