Available online at www.sciencedirect.com

ScienceDirect

Computer-Aided Design 40 (2008) 76–93

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Detecting approximate symmetries of discrete point subsets[☆]

Ming Li[*], Frank C. Langbein, Ralph R. Martin

School of Computer Science, Cardiff University, Cardiff, UK

Received 2 October 2006; accepted 5 June 2007

Abstract

Detecting approximate symmetries of parts of a model is important when attempting to determine the geometrical design intent of approximate boundary-representation (B-rep) solid models produced e.g. by reverse engineering systems. For example, such detected symmetries may be enforced *exactly* on the model to improve its shape, to simplify its analysis, or to constrain it during editing. We give an algorithm to detect *local approximate symmetries* in a discrete point set derived from a B-rep model: the output comprises the model's potential local symmetries at various *automatically* detected tolerance levels. Non-trivial symmetries of subsets of the point set are found as *unambiguous permutation cycles*, i.e. vertices of an approximately regular polygon or an anti-prism, which are sufficiently separate from other points in the point set. The symmetries are detected using a rigorous, tolerance-controlled, incremental approach, which expands symmetry seed sets by one point at a time. Our symmetry cycle detection approach only depends on inter-point distances. The algorithm takes time $O(n^4)$ where n is the number of input points. Results produced by our algorithm are demonstrated using a variety of examples.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Local approximate symmetry; Design intent; Reverse engineering

1. Introduction

Many manufactured objects exhibit global and local symmetries as a feature of their design or function, or for ease of manufacturing or analysis [1]. Furthermore, symmetry is also common in natural shapes [2] and designers prefer symmetrical shapes for reasons of aesthetics and simplicity [3]. This is particularly true for engineering objects conventionally represented by boundary-representation (B-rep) models, such as the one shown in Fig. 1.

While such symmetries *may* be explicitly represented along with a B-rep model, often they are not explicitly given, for example, where a model has been created by reverse engineering, or where a model has been transferred from one CAD system into another. Furthermore, in cases like these, the symmetries are often not *exactly* present, but only approximately present, due to measurement errors in the scanning process, and approximation and numerical errors in model reconstruction during reverse engineering [4]. Different CAD systems often use different tolerances [5], and what is symmetric in one CAD system may not be symmetric in another.

Explicit detection of symmetries in such geometrical models has many potential uses: for example, to improve the shape of a model by enforcing intended symmetries, to enable faster analysis, to place constraints on editing operations, and so on. We are thus interested in detecting the symmetries intended by a designer in a B-rep model, but which are only approximately present.

Our previous methods for geometric design intent detection can detect *global* approximate symmetries [6], approximate *congruencies* between sub-parts [7], and other local *regularities*, e.g. parallel and orthogonal planes [8]. This paper considers a different issue not solved by such approaches: finding *local* approximate symmetries in a B-rep model. For example, the model in Fig. 1 has cylindrical holes arranged with an eight-fold rotational symmetry, and slots with a sixteen-fold rotational symmetry.

To detect local symmetries, we use similar ideas to those used for global symmetric detection in [6,7]. As in these papers, we extract *characteristic points* from a solid model, which when used with connectivity and face type information, are sufficient to determine its symmetry, should symmetry

[☆] Supported by EPSRC UK Grant GR/S69085/01.

[*] Corresponding author. Tel.: +44 2920876751.
E-mail addresses: M.Li@cs.cf.ac.uk (M. Li), F.C.Langbein@cs.cf.ac.uk (F.C. Langbein), Ralph.Martin@cs.cf.ac.uk (R.R. Martin).

0010-4485/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2007.06.007

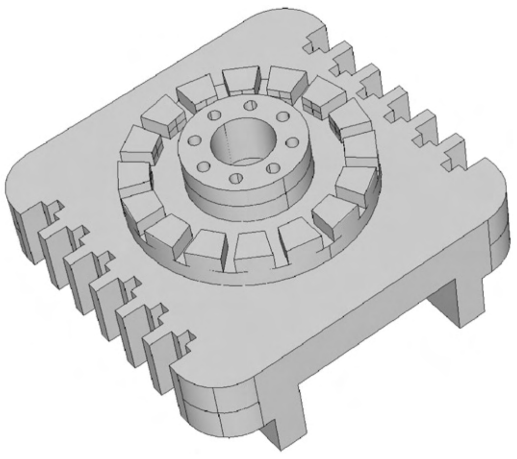
Fig. 1. A B-rep model with many local symmetries.

be present. Essentially, these points are the vertices of the B-rep model, together with other special points needed to characterize curved edges and faces. For example, a straight line is characterized by its two end points, whereas for an edge known to be a circular arc, using one other point taken to be the mid-point of the arc is sufficient to both fix its radius, and to determine which of the two arcs between those end points we want. A discussion of how to select these characteristic points is given in [7].

Note, then, that we start from a very different point of view to symmetry approaches used in image processing, e.g. [9–12], and mesh processing [13–15]. These are designed to work on dense point data, where the point distributions are far more important than locations of individual points. Generally, their aim is to detect one or a few *dominant* approximate symmetries by partial matching of images or meshes under user selected tolerances. In contrast, we wish to find symmetries in B-rep models that are intended to be exact, but are approximate due to their origins. We furthermore wish to generate *all* possible subset symmetries, where any one may belong to quite a small part of the model (such as a hole, or row of slots in a complex model). We use as a basis a carefully selected and generated point set from a B-rep model, not a dense point set covering the whole surface of the B-rep model. Our algorithm thus processes far fewer points than a mesh symmetry algorithm; in our algorithm, both the position, and existence, of every individual point is significant. However, we speculate that it might be possible to apply our method to detecting symmetries of meshes *if* a suitable means could be found for defining and extracting carefully chosen key feature points.

In summary, the main novel contribution of the paper is a rigorous definition of and an algorithm to detect *local approximate symmetries* possessed by subsets of a set of points in 2D or 3D. By letting this set of points be carefully chosen characteristic points extracted from a B-rep model as described above, these approximate point subset symmetries in turn directly correspond to approximate local symmetries of an approximate B-rep model. Finding approximately symmetrical subsets of a point set is an important topic not addressed by previous work. Here we are considering points which the symmetry maps in a one-to-one fashion onto each other. Mitra et al. [15] have considered the different problem of approximate maps of dense point clouds representing part of the surface of an object onto other dense point clouds from the same object—but these are not pointwise maps. Other related work is later discussed in Section 2.

The detected symmetries include rotational symmetries and rotation-reflection symmetries, i.e. vertices of an approximately regular polygon or an anti-prism. Each symmetry is represented as an *unambiguous (permutation) cycle* on that subset of points. Each *symmetry* corresponds to a transformation which maps a subset of the point set onto itself. As we assume that the input model is *approximate*, the subset may only map approximately onto itself under the symmetry. A *(permutation) cycle* is a subset of a permutation whose elements trade places with one another. It describes the orbit of a single point under consecutive application of a symmetry transformation (in the exact case), specifically under rotation and rotation-reflection. By *unambiguous* cycles we mean that the points involved in the subset are sufficiently far away from other points in the input point set, so that there is no chance of confusion as to which point maps to which under the symmetry transformation, given its approximate nature. These ideas are explained more carefully and rigorously in Section 3.

Note that in this paper we *only* concern ourselves with finding each individual cycle *separately*. Thus, given a regular prism, we will output one cycle corresponding to the vertices at one end, and a separate cycle for the vertices at the other end—even though (at least in the exact case) these two point subsets share the *same* symmetry transformation. Clearly, extracting higher level information is important: *merging* the cycles found by the method given in this paper will be addressed in future work. In the following, we shall always mean a cycle when we talk about a symmetry, unless we say otherwise.

Before we go further, we should just mention a special case. Clearly, every *pair* of points trivially defines an *exact* two-fold rotational symmetry, a reflection symmetry, and an inversion symmetry cycle. These cycles can be trivially 'found' by simply enumerating every pair of points, and so are not further discussed here. Thus, we consider how to find rotational symmetry cycles, i.e. vertices of an approximately regular polygon, and rotation-reflection symmetries, where the symmetry transformation comprises reflection in a plane followed by rotation about an axis perpendicular to that plane, i.e. vertices of an anti-prism.

There are seven elementary symmetry transformations: reflection, inversion, translation, rotation, glide reflection, rotation reflection and screw translation [16]. However, discounting inversion and reflection, only two other kinds – rotation and rotation-reflection – have finite cycles (i.e. if we apply the symmetry operation enough times, the points go back to their original permutation). Translational symmetries, and glide reflections and screw translations, which are combinations of translation respectively with reflection and rotation, must always be incomplete for finitely many points. Handling incomplete symmetries, both of this kind, and e.g. incomplete rotational symmetries, is not straightforward and will also be addressed in future work.

Although exact symmetry detection has been widely studied, e.g. [17,18], these methods cannot be directly extended to *approximate* symmetries by simply replacing tests for equality by tests for approximate equality. Algorithms for exact symmetry detection rely on making *local* decisions about which elements match under symmetry. For approximate symmetries, such decisions must be based on *global* properties. Point matching is no longer a Boolean property—points match to a certain degree, and in general, multiple potential matches have to be considered, increasing algorithmic complexity [6].

We build on the basic idea of Brass' work on detecting exact subset symmetries [18]. For exact n-fold rotational symmetries, given three ordered points forming an isosceles triangle, these uniquely determine a rotation mapping point one to point two, and point two to point three. Under recursive application of the map, we get point four and so on, expanding the initial seed set. During the expansion process, if a point is finally mapped back onto the first point, the expansion stops and the result is a symmetric regular polygon. While it is trivial to determine these point mappings in the exact case, for approximate symmetries the points do not map exactly onto each other, but are only matched within a certain tolerance. Thus it becomes nontrivial to determine them.

Detecting *local* approximate symmetries raises further issues—which point subsets should be examined for symmetry, and under what tolerance they display symmetry. These two issues depend on each other. Our approach allows us to *automatically determine* tolerances at which approximate symmetries of subsets are present, and does not require predefined tolerance bounds as input (also see [6]). Fixing some coarsely chosen upper limit to acceptable tolerances can help to reduce the number of unwanted approximate symmetries detected. However, it cannot directly help in finding local approximate symmetries, where appropriate tolerances must be derived from the point set itself. Choosing tolerances appropriate to the data is important, as typically for engineering components, quite different tolerances are used for different features of the same component.

Downstream processes may typically wish to merge the cycles found, or in other cases choose between them. We note that the tolerance information output by our method is important for any such processes which use the symmetries, e.g. to beautify reverse engineered models by solving geometrical constraint systems [8,19] or other applications for further selection between these detected symmetries [20]. After symmetries at certain tolerances have been detected it is simpler to select suitable symmetries at suitable tolerances.

Based on the definition of global approximate symmetries given in [6], we introduce in Section 3 a definition of approximate subset symmetries, or unambiguous cycles, leading to clear conditions under which approximate symmetries can be said to be present. Using this definition, we then give an algorithm for detecting approximate symmetries, by selecting symmetry seed sets and expanding them point by point. The initial seed sets are approximate isosceles triangles. Each detected cycle that is sufficiently separate from other points in the input point set represents an approximate subset symmetry.

We next discuss in more detail how our ideas are related to earlier work on exact and approximate symmetry detection. Our definition of approximate symmetry is given in Section 3. An overview of our algorithm for finding approximate symmetries in 2D is provided in Section 4, with further details in Section 5. Extension of the algorithm to 3D symmetry detection is given in Section 6. The time complexity of our symmetry detection algorithm is analysed in Section 7. Practical examples are discussed in Section 8, and conclusions are drawn in Section 9.

2. Previous work

We now discuss previous work on detection of exact and approximate symmetry. *Symmetry detection* is used to refer to two slightly different problems. Sometimes, it refers to finding the symmetry transformation (if any) under which a certain point set is mapped onto itself. Alternatively, it can mean finding a symmetrical set which is close (according to some similarity measure) to a set of points; in this case the transformation itself need not be explicitly found. Symmetry detection algorithms can be further classified as detecting either *global* or *subset* (local) symmetries, and as trying to find *exact* or *approximate* symmetries. Global symmetries involve mapping the whole set onto itself, whereas subset symmetries only map a subset onto itself. The latter are harder to compute as the subset in addition to the symmetry has to be identified. Exact symmetries preserve the point set exactly under transformation, whereas approximate symmetries map the point set onto itself within a tolerance. Different definitions for approximate symmetry exist depending on how matching under a tolerance is defined. This paper considers approximate subset symmetries, for sets of discrete points. As noted earlier, such sets suffice to identify local symmetries in B-rep models.

Algorithms for detecting *exact, global* symmetries of point sets and objects have been widely studied, e.g. [17,21–24]. Exact symmetry detection for planar collections of points and lines can be done in $O(n \log n)$ time [21]. Detecting symmetries of 3D point and line configurations, and polyhedra, has the same complexity [17,21]. The basic idea used is to sort the points according to distances from the centroid, and then to check how many there are at each distance, which essentially reduces the complexity of the problem to that of a sorting algorithm. Brass and Knauer [24] recently extended the idea to general 3D objects.

There are relatively few results on *exact* symmetries of *subsets*. Brass [18] detects rotational symmetries by finding rotational mappings based on isosceles triangles, and combines them into symmetrical subsets efficiently using a tree data structure. Mirror symmetries can easily be detected by combining mirror planes generated by point pairs. This work was improved by Aloupis [25] using a randomised approach. Another approach to finding subset symmetries in solid models was presented by Tate [26]. It is based on matching pairs of edge loops and finding the isometries that relate them. The isometries are then grouped according to similarity. Their implementation finds mirror planes of symmetry.

Most previous work on *approximate* symmetries has considered *global* approximate symmetries, using various definitions of approximate symmetry. Iwanowski [27] pointed out that testing approximate symmetry in the plane is NP-hard if approximate symmetry is defined in terms of the existence of an exactly symmetrical object near to the approximate object. Alternatively, approximate symmetry may be defined as existence of a transformation mapping the point set approximately onto itself within a certain tolerance; this yields high-order polynomial time algorithms [22] for symmetry detection. Mills et al. [6] give a method for approximate global symmetry detection that combines the combinatorial and geometrical nature of symmetries, resulting in a low-order polynomial time algorithm. A completely different approach to detecting global approximate symmetries is to define an *asymmetry* measure [28,29]. Zabrodski [28] defines this as the minimum, taken over all exactly symmetric shapes with the desired symmetry, of the mean squared distance between points of the original shape and the symmetrical shape.

Finally, we turn to the case of *approximate* symmetries of *subsets* of points. To our knowledge, no previous work addresses this topic. (We again emphasize that we seek one-to-one correspondences between points, so our work is quite different from previous work on detecting local symmetries in images e.g. [9,12] or meshes [14,15] which use dense sets of points). The difficulties of the problem lie in choosing point subsets for consideration, inferring the symmetry transformations, and automatically determining the tolerance for each symmetry; furthermore, all of these issues depend on each other. The requirement to find *all* potential approximate symmetries at appropriate tolerances increases the difficulty of the problem. We employ a similar definition of *approximate* symmetry to Mills et al. [6], but modify it so that it can find *subsets*, based on the ideas for finding exact symmetrical subsets given by Brass [18].

3. Approximate symmetries of point subsets

Exact symmetry is a well-defined concept and there are efficient algorithms to detect it. Approximate symmetry is harder to define—there is more than one way to do so, and to some extent, the most appropriate definition depends on the particular application. In this section, we give a rigorous formalization of our particular concept of approximate symmetry for use in design intent detection.

Throughout this paper we use the following notation:

$\mathbb{E}^d$	d-dimensional Euclidean space (here, $d = 2$ or 3).
$\|P - Q\|$	Euclidean distance between points $P, Q \in \mathbb{E}^d$.
$\mathcal{D}(\mathcal{P})$	The set of distances $\{\|P - Q\| : P, Q \in \mathcal{P}\}$ for a set $\mathcal{P}$ of points.
$\lvert\mathcal{P}\rvert$	The number of elements in a set $\mathcal{P}$.
$\lfloor r \rfloor$	The largest integer not greater than a real number r.
$a =_\epsilon b$	Equality of real numbers within tolerance ϵ: $\lvert a - b\rvert \leq \epsilon$.
a mod n	Remainder of division of integer a by integer n except when the remainder is 0 whereupon we set it to n as we use indices starting from 1.

Assume that a point set $\mathcal{P}$ has an *exact*, global symmetry. In $\mathbb{E}^d$ this symmetry T of $\mathcal{P}$ is fully determined by a mapping from $d + 1$ points $\mathcal{S}_0 \subset \mathcal{P}$ onto another $d + 1$ points $\mathcal{S}_1 \subset \mathcal{P}$. The images of the remaining points $\mathcal{P} \setminus \mathcal{S}_0$ are fully determined by mapping their distances from the points in $\mathcal{S}_0$ onto the corresponding distances from the points in $\mathcal{S}_1$.

This mapping induces a permutation on the points in $\mathcal{P}$. In the exact case these distances match exactly and uniquely, which allows for efficient symmetry detection algorithms that extend partial matches to complete ones. In the *approximate* case, however, T maps points and their distances only approximately onto each other, and care is needed to find a *globally consistent* matching between points. Just taking the best match locally is insufficient, and in general, an expensive backtracking approach is required. We have carefully chosen the definition below for approximate symmetry so that it allows an algorithm to be devised based on expanding local matches without backtracking, enabling us to keep the efficiency of the approach used in [6]. *Approximate symmetry* of a point set is defined in terms of a permutation of the points which maps distances between the points approximately onto each other:

Definition 1 (*Approximate Symmetry*). Let $\mathcal{P} \subset \mathbb{E}^d$ be a point set. We call a pair (ϵ, σ), for which $\epsilon \geq 0$ and σ is a permutation on $\mathcal{P}$, an *approximate symmetry* of $\mathcal{P}$ if $=_\epsilon$ is an equivalence relation on $\mathcal{D}(\mathcal{P})$, and $\|P - Q\| =_\epsilon \|\sigma(P) - \sigma(Q)\|$ for all $P, Q \in \mathcal{P}$.

Note that an approximate symmetry with $\epsilon = 0$ yields an exact symmetry.

We now explain the ideas behind this definition. The condition that $=_\epsilon$ forms an equivalence relation means that the set $\mathcal{D}(\mathcal{P})$ of all distances between the points $\mathcal{P}$ is grouped into pairwise distinct subsets of approximately equal distances (equivalence classes). This partly recovers the exact matching property of exact symmetries. In the exact case, given the images $\mathcal{P}'$ of $d + 1$ points $\mathcal{P}$, we can determine the other points and their images by the distances from $\mathcal{P}$ and $\mathcal{P}'$. In the approximate case with an arbitrary tolerance the distances only match approximately and hence do not uniquely determine the points. By requiring that $=_\epsilon$ forms an equivalence relation we avoid this situation: if two points have approximately (within ϵ) the same distances from a set of $d + 1$ points in $\mathcal{P}$ they

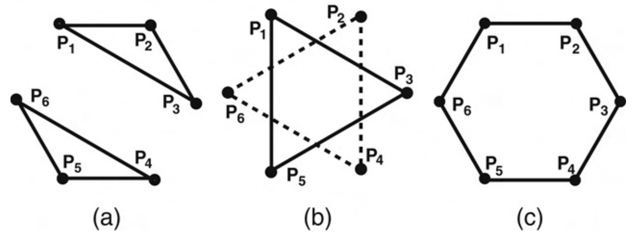

Fig. 2. Some symmetries of a hexagon.

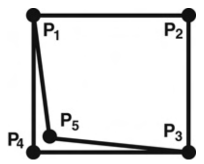

Fig. 3. Unambiguous symmetry.

are effectively identified with each other and treated as having the same approximate position. But note that this does not hold true for points not in $\mathcal{P}$, which means for subsets we have to introduce further conditions involving the complete point set.

For example, consider a 2D point set $\mathcal{P} = \{P_k : 1 \leq k \leq 6\}$ forming an approximate hexagon. Fig. 2 shows three of its symmetries described by permutations: the mirror symmetry $\sigma_1 : (P_1, P_6), (P_2, P_5), (P_3, P_4)$ in Fig. 2(a); the three-fold rotational symmetry $\sigma_2 : (P_1, P_3, P_5), (P_2, P_4, P_6)$ in Fig. 2(b); and the six-fold rotational symmetry $\sigma_3 : (P_1, P_2, P_3, P_4, P_5, P_6)$ in Fig. 2(c). If we were detecting subset symmetries in the case in which $\mathcal{P}$ were a subset of a larger point set, we would, however, only be interested in detecting the six-fold symmetry σ_3 for $\mathcal{P}$. This is the only symmetry whose permutation consists of a single cycle. The other permutations would be found at other times when considering different subsets of the complete, larger point set. Thus, our algorithm detects symmetries in terms of single cycles as these form the elementary symmetry structures of a point set. In 2D the cycles correspond to the ordered vertices of regular polygons, while in 3D they may in addition be the ordered alternate vertices taken from opposite ends of anti-prisms as further discussed in Section 6.2.

Following Definition 1, a tolerance interval $E_{\min}(\mathcal{C}) \leq \epsilon < E_{\max}(\mathcal{C})$ exists for ϵ at which a symmetry cycle $\mathcal{C}$ is present. These are called the *minimal* and *maximal tolerances* of $\mathcal{C}$. $E_{\min}(\mathcal{C})$ ensures equality of all the distances in the same distance class—if $E_{\min}(\mathcal{C})$ were too small, various distances would no longer be considered equal, and the approximate symmetry would no longer exist. $E_{\max}(\mathcal{C})$ separates different classes—if $E_{\max}(\mathcal{C})$ were too large, the approximate symmetry would map a given point onto more than one possibility, and the approximate symmetry would no longer be unambiguous. We require that $E_{\min}(\mathcal{C}) < E_{\max}(\mathcal{C})$ for $\mathcal{C}$ to be a proper cycle—see Theorem 1. Further discussion of this issue is given in Section 5.

For symmetries of point *subsets* in an input point set, we have to consider another issue, to specify the subsets we are interested in. Consider, for example in Fig. 3, P_1, P_2, P_3, P_4 construct an exact square and P_5 is a point close to P_4.

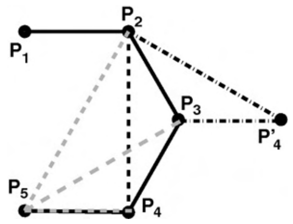

Fig. 4. Illustration of the expansion process.

Suppose $P_1, \ldots, P_5$ form an input point set $\mathcal{P}$. They yield two subset symmetries given by the permutation cycles $\mathcal{C}_1 : (P_1, P_2, P_3, P_4)$, $\mathcal{C}_2 : (P_1, P_2, P_3, P_5)$ at proper tolerances ϵ_1, ϵ_2 respectively, where $\epsilon_1 < \epsilon_2$. However, $\mathcal{C}_2$ is ambiguous as P_5 can be replaced by P_4 without changing the symmetry transformation at tolerance ϵ_2. We must avoid such cycles containing ambiguous points if the description of a symmetry by a cycle is to be unique. The following definition gives a condition for when a given cycle $\mathcal{C}$ belonging to a point set $\mathcal{P}$ is symmetric and sufficiently separate from the other points in $\mathcal{P}$ to avoid ambiguity: no point in $\mathcal{P} \setminus \mathcal{C}$ can replace a point in $\mathcal{C}$ such that $\mathcal{C}$ is still symmetric at the same tolerance. As an approximate symmetry can be present at a range of tolerance values, we must enforce this requirement at the minimal tolerance $E_{\min}(\mathcal{C})$.

Definition 2 (*Unambiguous Cycle*). Let $\mathcal{C}$ be a cycle formed by a subset of points from a point set $\mathcal{P}$ at the minimal tolerance $e^* = E_{\min}(\mathcal{C})$. We say that $\mathcal{C}$ is *unambiguous* with respect to $\mathcal{P}$ if $\mathcal{C}$ stops being a cycle at tolerance e^* if we replace any point in $\mathcal{C}$ by any point in $\mathcal{P} \setminus \mathcal{C}$.

Finally, we can now state what our algorithm computes: given a point set $\mathcal{P} \subset \mathbb{E}^d$, $d = 2$ or $d = 3$, our algorithm computes each of its unambiguous cycles $\mathcal{C}_k$ together with their minimal tolerance $\epsilon_k = E_{\min}(\mathcal{C}_k)$, i.e. $(\mathcal{C}_k, \epsilon_k)$, $1 \leq k \leq r$.

4. Algorithm overview

In this section we provide an overview of our algorithm for detecting approximate symmetries of a 2D point subset, each represented as an unambiguous cycle. Further 2D algorithm details are given in Section 5. The algorithm for 3D point sets is similar, and is further described in Section 6.

The basic idea behind our symmetry detection algorithm comes from the following ideas in the exact case. Let P_1, P_2, P_3 describe an isosceles triangle in which $\|P_1 - P_2\| = \|P_2 - P_3\|$ and $\|P_1 - P_3\| \geq \|P_1 - P_2\|$ (see Fig. 4). These three points define a rotation, under a (partial) permutation that P_1 moves to P_2, and P_2 moves to P_3; this rotation potentially represents the symmetry of a regular polygon. Suppose we now look for another point P_4 to extend this symmetry such that the rotation maps P_3 onto P_4. We can find P_4 by noting that P_4 must satisfy $\|P_4 - P_3\| = \|P_2 - P_1\|$ and $\|P_4 - P_2\| = \|P_3 - P_1\|$. These constraints correspond to two possible locations for P_4, and only the one lying on the same side of the line P_2P_3 as P_1 is valid and chosen. (In the approximate case we have to consider both as explained below.) To proceed further, we can find the next expansion point P_5 by replacing (P_1, P_2, P_3, P_4) by

```
Algorithm: Detect approximate symmetries (unambiguous cycles) present in a 2D point set
Input:     $\mathcal{P} \subset \mathbb{E}^2$, a set of pairwise distinct 2D points
Output:    $\mathcal{T} = \{(\mathcal{C}_k, \epsilon_k) : 1 \leq k \leq r\}$, a set of unambiguous cycles, each represented by
           an ordered list of points $\mathcal{C}_k$, and its corresponding minimal tolerance $\epsilon_k$
01 $\mathcal{T} \leftarrow$ empty
02 $\mathcal{D} \leftarrow$ array of distances between each pair of points in $\mathcal{P}$
03 for each point triple $\mathcal{S} = (P_1, P_2, P_3)$
             with $\|P_1 - P_3\| \geq \|P_2 - P_3\| \geq \|P_1 - P_2\|$ from $\mathcal{P}$ do
04   if $\mathcal{S}$ is a contiguous part of a cycle in $\mathcal{T}$ then continue // skip this triple
05   if IsUnambiguousCycle($\mathcal{S}$) then   // test for unambiguous triangle
06     $\mathcal{T} \leftarrow \mathcal{T} \cup \{(\mathcal{S}, E_{\min}(\mathcal{S}))\}$
07   end if
08   do
09     $M \leftarrow \{P : E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\}),\ P \in \mathcal{P} \setminus \mathcal{S}\}$
10     if $M =$ empty then break   // no cycle found
11     $P^* \leftarrow \arg\min\{E_{\min}(\mathcal{S} \cup \{P\}) : P \in M\}$
12     $\mathcal{S} \leftarrow \mathcal{S} \cup \{P^*\}$   // add $P^*$ to the end of the ordered list $\mathcal{S}$
13   while not IsCycle ($\mathcal{S}$)
14   if IsUnambiguousCycle($\mathcal{S}$) then   // test for unambiguous polygon
15     $\mathcal{T} \leftarrow \mathcal{T} \cup \{(\mathcal{S}, E_{\min}(\mathcal{S}))\}$
16   end if
17 end for
18 return $\mathcal{T}$
```

Fig. 5. 2D approximate symmetry (unambiguous cycle) detection algorithm.

(P_2, P_3, P_4, P_5) in the above process. Applying this expansion process iteratively will eventually lead to a cycle corresponding to the ordered vertices of a regular polygon if it is present.

However, in the approximate case, determining the points involved in a symmetry using an expansion process is more complicated, due to the difficulty in determining whether two distances are approximately the same, both because of the need to choose a tolerance, and due to the possible accumulation of errors. To avoid accumulating expansion errors, we add further constraints to determine the expansion point. For example, in order to determine the expansion point P_5 from the seed set (P_1, P_2, P_3, P_4) in Fig. 4, instead of simply requiring at some tolerance ϵ that $\|P_5 - P_4\| =_\epsilon \|P_4 - P_3\|$ and $\|P_5 - P_3\| =_\epsilon \|P_4 - P_2\|$, we require the equality of *all* the distances within the same distance group, i.e. $\|P_5 - P_4\| =_\epsilon \|P_{k+1} - P_k\|, k = 1, 2, 3$, $\|P_5 - P_3\| =_\epsilon \|P_{k+2} - P_k\|, k = 1, 2$ and $\|P_5 - P_2\| =_\epsilon \|P_4 - P_1\|$. Note here that P_5 is uniquely determined by these distance equalities without any further conditions, at least in the exact case.

Note that we do not base our approximate symmetry detection algorithm on a predetermined tolerance, but instead, we find suitable tolerances as the algorithm proceeds. The ideas are based on three theorems discussed later in the paper. We use Theorem 1 to detect potential expansion points: if a point P is to be a *valid* expansion point for seed set $\mathcal{S}$, the following relationship must be satisfied: $E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\})$; $E_{\min}(\mathcal{S} \cup \{P\})$ and $E_{\max}(\mathcal{S} \cup \{P\})$ are computed during the point expansion process from inter-point distances as explained in detail in Section 5.2.2. However, more than one potential expansion point may exist in the input point set. We choose the one which minimizes $E_{\min}(\mathcal{S} \cup \{P\})$. This strategy tries to avoid adding any points that violate the unambiguity condition while also ensuring that we always find any expansion point which exists. Taking this idea further, a condition under which a complete cycle can be found is provided in Theorem 2. While we always choose the expansion point which minimizes the tolerance, this tolerance may increase as further points are added during the expansion process. Hence, we still need to check the unambiguity condition for the final detected cycle before it is output; doing so is based on Theorem 3.

The algorithm itself, for detecting approximate symmetries expressed as unambiguous cycles, is listed in Fig. 5. It takes as input a set of distinct 2D points $\mathcal{P}$ and outputs all unambiguous cycles as pairs $(\mathcal{C}_k, \epsilon_k)$ where $\mathcal{C}_k$ lists the points in order forming a cycle, and ϵ_k is the minimal tolerance at which $\mathcal{C}_k$ forms an unambiguous cycle. We assume no two input points are at the same location.

In Line 1 the output list of cycles $\mathcal{T}$ is initialized as empty. In Line 2 the distances between all pairs of points are precomputed for efficiency. Lines 3–17 form an outer loop over all triples of points in $\mathcal{P}$. These triples are seeds for the point expansion process, some of which will lead to unambiguous cycles. Each initial seed set forms an approximate, rather than exact, isosceles triangle. The triple is put into an ordered list (P_1, P_2, P_3) such that P_1 maps to P_2, and P_2 to P_3, and $\|P_1 - P_3\| \geq \|P_2 - P_3\| \geq \|P_1 - P_2\|$. We now consider expanding a cycle starting from P_1, P_2, P_3 (Line 3). Note that the same symmetry can be found starting with several different triples. Hence, we further check in Line 4 whether the triple is already a contiguous part of a previously detected cycle and if so, ignore it.

A given point triple may form an approximately regular triangle, or may be expanded to a regular polygon with more than 3 sides, or even both. We consider these cases respectively in Lines 5–7 and Lines 8–16. (Note that deciding which of these different cases is actually present in the model is not considered in this paper, but left as a problem for a downstream process, guided by some higher level information: see [8]; here we only detect what is unambiguously present in the model data). We first check, as a special case, if the three points form an approximately regular triangle, and output them if so (Lines 5–7). How this is done is described in Section 5.3.1.

We next check whether an unambiguous cycle containing more than three points can be generated from the triple (Lines 8–16). To find the correct expansion point from a set $\mathcal{S}$, first a set M of potential expansion points is determined (Line 9). As explained above, each such point P must satisfy $E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\})$. If any such points exist, i.e. M is not empty, we choose the point which minimizes the tolerance required for the potential approximate symmetry (Line 11) and add it to $\mathcal{S}$ (Line 12). Expansion from the initial three points is handled slightly differently from further expansion points, as there are two possible locations for which the distances between the points match approximately, and they both have to be considered, as we explain in detail in Section 5.3.2; this is omitted from the algorithm listing for simplicity. Expansion stops when either no more expansion points can be found (Line 10) or a complete cycle has been detected (Line 13). If a cycle is detected, we must further verify that it satisfies the unambiguity condition, i.e. that there is no other point close to some point in $\mathcal{S}$ which can replace it and still give the same symmetry at the same tolerance (Line 14). If verified, we accept this detected cycle as an unambiguous cycle (Line 15).

5. Unambiguous cycles of 2D point sets

We now give further details of our 2D unambiguous cycle detection algorithm outlined in Section 4. We first explore the basic properties of the point expansion approach in Section 5.1. Using these ideas, we then show in Section 5.2 how to select suitable expansion points, compute the associated tolerance ranges, and verify whether an expanded cycle is unambiguous. Section 5.3 considers conditions on the valid initial seed sets for point expansion and how to expand them in the special case of the fourth point.

5.1. Tolerance conditions for symmetry seed set expansion

In this section we derive the conditions imposing limits on the tolerances under which a set $\mathcal{S}$ expanded by a point P may still lead to an approximate symmetry. We also give theorems stating when an expansion gives a cycle and when the cycle is unambiguous. We obtain these by analysing what conditions are to be fulfilled for a list of consecutive points if they form a cycle.

Let $\mathcal{C} \subset \mathbb{E}^2$ be a cycle of $c = |\mathcal{C}|$ points at tolerance ϵ. The cycle $\mathcal{C} = \{P_k : 1 \le k \le c\}$ can be seen as generated by a permutation that maps P_l to P_k, satisfying for each $1 \le r \le c-1$

$$\|P_l - P_{(l+r) \bmod c}\| =_\epsilon \|P_k - P_{(k+r) \bmod c}\|, \quad \text{for } 1 \le l, k \le c. \tag{1}$$

Thus, all the distances between point pairs (P_l, P_k) with $l - k = r$ or $l - k = c - r$ are approximately the same at tolerance ϵ. Moreover, as $=_\epsilon$ is an equivalence relation on the set $\mathcal{D}(\mathcal{C})$ of all distances between the points in $\mathcal{C}$, each group of distances between point pairs with index differences r or $c - r$ corresponds to one equivalence class. Therefore the number of distance equivalence classes is $\lfloor c/2 \rfloor$.

The equivalence classes play an essential role in the approximate symmetry definition, Definition 1, and enforce constraints on the tolerances allowed for a symmetrical set. In the following, we apply them to infer what condition a list of consecutive points should satisfy if it lies in a cycle.

Consider a subset $\mathcal{S}$ of $s = |\mathcal{S}|$ consecutive points $P_{k+1}, P_{k+2}, \ldots, P_{k+s}$ from $\mathcal{C}$, with indices taken modulo c. Clearly, Eq. (1) still holds for the distances between points in $\mathcal{S}$ and groups these in the same way as for $\mathcal{C}$. We call a set with such properties a *(symmetry) seed set* at tolerance ϵ. The fact that the number of different distance classes in $\mathcal{D}(\mathcal{C})$ is $\left\lfloor \frac{c}{2} \right\rfloor$ yields that the number of different distance classes in $\mathcal{D}(\mathcal{S})$ is

$$g(s, c) = \min\left(s - 1, \left\lfloor \frac{c}{2} \right\rfloor\right). \tag{2}$$

Then for each $1 \le r \le g(s, c)$ we get a distance class

$$\mathcal{G}^r(\mathcal{S}) = \{\|P_k - P_{k+r}\| : 1 \le k \le s - r\} \cup \{\|P_k - P_{k+r-c}\| : 1 \le k \le s + c - r\}. \tag{3}$$

Let

$$D^r_{\min}(\mathcal{S}) = \min(\mathcal{G}^r(\mathcal{S})), \qquad D^r_{\max}(\mathcal{S}) = \max(\mathcal{G}^r(\mathcal{S})), \tag{4}$$

respectively be the minimal and maximal distance in each class $\mathcal{G}^r(\mathcal{S})$. Clearly, $D^r_{\min}(\mathcal{S}) \le D^r_{\max}(\mathcal{S})$.

As $\mathcal{S}$ satisfies Eq. (1) and $=_\epsilon$ is an equivalence relation on $\mathcal{D}(\mathcal{S})$, $\mathcal{S}$ being a seed set is equivalent to

$$\begin{aligned} &D^r_{\max}(\mathcal{S}) - D^r_{\min}(\mathcal{S}) \le \epsilon, \quad 1 \le r \le g(s, c), \\ &D^{r+1}_{\min}(\mathcal{S}) - D^r_{\max}(\mathcal{S}) > \epsilon, \quad 1 \le r \le g(s, c) - 1. \end{aligned} \tag{5}$$

These inequalities give a range of tolerance values ϵ for $\mathcal{S}$ to be a seed set based on the actual distances between the points in $\mathcal{S}$: ϵ must be large enough for all distances in each distance class $\mathcal{G}^r(\mathcal{S})$ to be approximately the same and small enough not to confuse different $\mathcal{G}^r(\mathcal{S})$. Furthermore, Eq. (5) is a condition for $\mathcal{S}$ being a seed set. Thus any subset of points satisfying Eq. (5) is a seed set which may lead to an unambiguous cycle $\mathcal{C}$ on expansion.

For simplicity in the above tolerance condition derivation, we assume the potential cycle $\mathcal{C}$ that a seed set $\mathcal{S}$ may lie in (see Eq. (2)) is known. We discuss further in Section 5.2 how, during the point expansion process, the cycle and its size $c = |\mathcal{C}|$ are determined.

By defining the maximum size of all distance classes to be the minimum matching tolerance

$$E_{\min}(\mathcal{S}) = \max_{1 \le r \le g(s,c)} (D^r_{\max}(\mathcal{S}) - D^r_{\min}(\mathcal{S})), \tag{6}$$

and the minimum distance between two consecutive distance classes as the maximum separation tolerance

$$E_{\max}(\mathcal{S}) = \min_{1 \le r \le g(s,c)-1} (D^{r+1}_{\min}(\mathcal{S}) - D^{r}_{\max}(\mathcal{S})), \tag{7}$$

we can simplify Eq. (5) to

$$E_{\min}(\mathcal{S}) \le \epsilon < E_{\max}(\mathcal{S}). \tag{8}$$

Finally, then, to verify whether $\mathcal{S}$ is a suitable seed set, we only have to check whether *some* ϵ exists for Eq. (8) to be satisfied, i.e. that:

$$E_{\min}(\mathcal{S}) < E_{\max}(\mathcal{S}). \tag{9}$$

Thus Eq. (9) allows us to check the validity of a seed set without reference to any explicit tolerance ϵ, using only quantities derived from the set itself. We may now apply this result to give a condition to decide whether a seed set can be expanded:

Theorem 1. *For a point set $\mathcal{P} \subset \mathbb{E}^2$, let $\mathcal{S} \subset \mathcal{P}$ be a seed set. The set $\mathcal{S}_+ = \mathcal{S} \cup \{P\}$ with $P \in \mathcal{P} \setminus \mathcal{S}$ is also a seed set if and only if*

$$E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+). \tag{10}$$

We now consider how to find $E_{\min}(\mathcal{S}_+)$ and $E_{\max}(\mathcal{S}_+)$.

For each $1 \le r \le g(s+1,c)$, let $\mathcal{D}^r_+$ be the set of distances between P_{s+1} and the points in $\mathcal{S}$ that may have index gaps r or $c-r$ to P_{s+1} when P_{s+1} is seen as the expansion point of $\mathcal{S}$. We get

$$\mathcal{D}^r_+ = \begin{cases} \{\|P_{s+1} - P_{s+1-r}\|\}, & \text{if } s \le \left\lfloor \frac{c}{2} \right\rfloor; \\ \{\|P_{s+1} - P_{s+1-r}\|, \|P_{s+1} - P_{s+1-(c-r)}\|\}, & \\ \quad \text{if } s > \left\lfloor \frac{c}{2} \right\rfloor, r \ge c-s. & \end{cases}$$

Hence, we can write

$$D^r_{\min}(\mathcal{S}_+) = \min(\mathcal{D}^r_+ \cup \{D^r_{\min}(\mathcal{S})\}),$$
$$D^r_{\max}(\mathcal{S}_+) = \max(\mathcal{D}^r_+ \cup \{D^r_{\max}(\mathcal{S})\}). \tag{11}$$

In the special case that $r = g(s+1,c) = s+1$, $D^r_{\min}(\mathcal{S})$ and $D^r_{\max}(\mathcal{S})$ do not exist and we treat the sets $\{D^r_{\min}(\mathcal{S})\}$ and $\{D^r_{\max}(\mathcal{S})\}$ as empty.

Eq. (11) tells us that instead of computing $D^r_{\min}(\mathcal{S}_+)$, $D^r_{\max}(\mathcal{S}_+)$, $1 \le r \le g(s+1,c)$, based on Eq. (4), we can efficiently calculate them from the known values $D^r_{\min}(\mathcal{S})$, $D^r_{\max}(\mathcal{S})$, $1 \le r \le g(s,c)$, and the distance set $\mathcal{D}^r_+$. Similarly this also leads to an efficient way to compute $E_{\min}(\mathcal{S}_+)$, $E_{\max}(\mathcal{S}_+)$.

From Eq. (11) it follows that $D^r_{\max}(\mathcal{S}_+) \ge D^r_{\max}(\mathcal{S})$ and $D^r_{\min}(\mathcal{S}_+) \le D^r_{\min}(\mathcal{S})$, and hence

$$\begin{aligned} E_{\min}(\mathcal{S}_+) &= \max_{1 \le r \le g(s+1,c)} (D^r_{\max}(\mathcal{S}_+) - D^r_{\min}(\mathcal{S}_+)) \\ &\ge \max_{1 \le r \le g(s,c)} (D^r_{\max}(\mathcal{S}) - D^r_{\min}(\mathcal{S})) = E_{\min}(\mathcal{S}). \end{aligned}$$

Similarly $E_{\max}(\mathcal{S}_+) \le E_{\max}(\mathcal{S})$. Furthermore, Eq. (8) is satisfied for $\mathcal{S}_+$, so

$$E_{\min}(\mathcal{S}) \le E_{\min}(\mathcal{S}_+) \le \epsilon < E_{\max}(\mathcal{S}_+) \le E_{\max}(\mathcal{S}), \tag{12}$$

i.e. the additional point distances introduced by adding a new point to $\mathcal{S}$ such that the expanded set $\mathcal{S}_+$ remains a seed set generally reduces the range of valid tolerance values ϵ.

We now consider how expansion terminates. An initial seed set $\mathcal{S}$ is expanded consecutively by finding additional points satisfying Eq. (10) until either no further expansion points exist, or a cycle is found. In the former case there is no symmetry, as there is no cycle. In the latter case the fully expanded seed set $\mathcal{S}$ is a symmetrical set at tolerance $\epsilon = E_{\min}(\mathcal{S}) < E_{\max}(\mathcal{S})$ as it satisfies Eq. (1) and $=_\epsilon$ is an equivalence relation on $\mathcal{D}(\mathcal{S})$.

A seed set describes a complete cycle if $\|P_{s+1} - P_1\| \in \mathcal{G}^1(\mathcal{S})$, because in this case due to the definition of $\mathcal{G}^r(\mathcal{S})$, Eq. (3), we have $k = s+1, r = 1, k+r-c = 1$ and hence $s+1 = c$. Thus we get

Theorem 2. *Let $\mathcal{P} \subset \mathbb{E}^2$ be a point set and $\mathcal{S} \subset \mathcal{P}$ with s points $P_1, P_2, \ldots, P_s$ ordered in sequence by the expansion process according to Theorem 1. $\mathcal{S}$ describes a cycle if $\|P_s - P_1\| \in \mathcal{G}^1(\mathcal{S})$ and is approximately symmetric at tolerance $E_{\min}(\mathcal{S})$.*

Note how our algorithm avoids accumulating expansion errors by considering *all* possible distances to *all* other points in the seed set $\mathcal{S}$. Errors could rapidly accumulate if the process simply determines an expansion point merely by its distances to the three previous points in the seed set.

A cycle $\mathcal{S}$ for a point set $\mathcal{P}$ is only unambiguous with respect to $\mathcal{P}$ if there is no point in $\mathcal{P} \setminus \mathcal{S}$ close to the points in $\mathcal{S}$ with respect to the symmetry tolerance ϵ. The following theorem, which follows from the above considerations, gives the condition to verify this after a complete cycle has been found.

Theorem 3. *Let $\mathcal{C} = \{P_k, 1 \le k \le c\}$ be a cycle within $\mathcal{P} \subset \mathbb{E}^2$ at tolerance $\epsilon = E_{\min}(\mathcal{C})$. For each point P_k, let $\mathcal{C}^k_-(P)$ be the point set generated by replacing P_k by another point $P \in \mathcal{P} \setminus \mathcal{C}$. $\mathcal{C}$ is an unambiguous subset of $\mathcal{P}$ if and only if $E_{\min}(\mathcal{C}^k_-(P)) > \epsilon$ for all choices of $P \in \mathcal{P} \setminus \mathcal{C}$.*

How to efficiently apply the theorem to check unambiguity of a cycle is further explained in Section 5.2.1.

5.2. Symmetry seed set expansion

In this section we illustrate in detail the point expansion process of a seed set $\mathcal{S}$ with $s = |S| \ge 3$, such that the expansion leads to an unambiguous cycle. How to obtain seed sets consisting of three points from the input set $\mathcal{P}$ is described in Section 5.3. Repetitive application of this point expansion process is used in the overall algorithm to find unambiguous cycles.

5.2.1. Expansion point selection

This section describes how to expand a seed set by an additional point and verify that a resulting cycle is unambiguous. We use Theorems 1–3 for expansion point selection, the termination condition, and unambiguity verification respectively.

Given a seed set $\mathcal{S}$ within an input point set $\mathcal{P}$, selecting a unique expansion point (if one exists) requires the determination of a suitable tolerance ϵ: once this has been done, the unique expansion point $P \in \mathcal{P} \setminus \mathcal{S}$ fulfilling Eq. (8) can be determined. Eq. (12) tells us that adding an additional point may cause the range $E_{\min}(\mathcal{S})$ to $E_{\max}(\mathcal{S})$ of acceptable tolerance values to become narrower. Theorem 1 tells us that we can expand $\mathcal{S}$ by adding an additional point as long as some point exists for which this interval does not become empty. We choose amongst these points, if there are several, by taking the one which keeps the tolerances smallest, as we describe shortly.

Note that at no time *during* the expansion process are we able to *fix* a specific value for the tolerance ϵ. If we were to set ϵ to the minimal tolerance in the initial set $E_{\min}(\mathcal{S})$, then it might not be possible to expand the initial set to a cycle as this value could be too restrictive. Moreover, even if we were to try all possible initial seed sets, and set the tolerance from them, the tolerance might have to be increased when adding additional points as indicated by Eq. (12).

If we were to choose a tolerance ϵ larger than necessary, e.g. $E_{\max}(\mathcal{S}) - \delta$ for a very small δ, to include the additional point in the symmetry, it might be possible to replace certain points in $\mathcal{S}$ with other points from $\mathcal{P}$ at the tolerance, violating the unambiguity condition, when a smaller value of ϵ would not have this problem. In Fig. 3, for example, if we expand from $\mathcal{S} = \{P_1, P_2, P_3\}$ at tolerance $E_{\max}(\mathcal{S})$, Eq. (8) is satisfied by adding either P_4 or P_5. However, P_5 is an unnecessary point and can actually be ruled out using a smaller tolerance, say $\|P_4 - P_5\|/2$.

So instead of fixing the tolerance, we keep track of the tolerance interval and check whether there is any expansion point which can be added while keeping the tolerance within the interval given in Theorem 1.

To select a unique expansion point, we consider each potential expansion point $P \in \mathcal{P} \setminus \mathcal{S}$ satisfying $E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+)$ for $\mathcal{S}_+ = \mathcal{S} \cup \{P\}$. We compute $E_{\min}(\mathcal{S}_+)$ and choose as the expansion point the one for which this tolerance is minimal.

If there is no point in $\mathcal{P} \setminus \mathcal{S}$ satisfying $E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+)$, no valid expansion point can be found and the expansion process stops. If there is more than one point minimizing $E_{\min}(\mathcal{S}_+)$, we have multiple expansion points and expansion will also stop. However, using real arithmetic, such a situation is unlikely to arise in practice.

Only after a complete cycle has been found, by checking the condition $\|P_{s+1} - P_1\| \in \mathcal{G}^1$ (Theorem 2) for the expansion point P_{s+1}, can we compute the actual minimal tolerance for the approximate symmetry. However, the resulting cycle may not be unambiguous with respect to the input point set $\mathcal{P}$, and further checking is required based on Theorem 3 as we now explain. For example, in Fig. 6, P_1, P_2, P_3, P_4 are points of an exact six-fold rotation. P_5 does not exactly lie in the set, and P_4' is a point close to P_4. Starting from P_1, P_2, P_3, we find the next expansion point P_4 with a tolerance $\epsilon = 0$ rather than P_4', which gives a larger tolerance $\epsilon' > 0$. The expansion process continues with P_5 which increases the tolerance ϵ. At this new tolerance, it may be possible that P_4' can be used to replace P_4

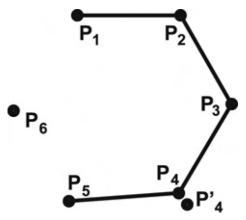

Fig. 6. Verification of unambiguity condition of a cycle.

while giving the same symmetry. We cannot simply conclude that P_4' can replace P_4 simply based on $\epsilon > \epsilon'$, however, as these tolerances are not based on the final cycle.

Unambiguity verification must be based on Theorem 3. Specifically, suppose $\mathcal{C} = \{P_l : 1 \le l \le c\}$ is a cycle of $\mathcal{P}$ at tolerance $\epsilon = E_{\min}(\mathcal{C})$ and let $\mathcal{C}_-^k(P)$ be the point set generated by replacing P_k with another point $P \in \mathcal{P} \setminus \mathcal{C}$. We need to check that for each $1 \le k \le c$

$$E_{\min}(\mathcal{C}_-^k(P)) > \epsilon \tag{13}$$

for each point $P \in \mathcal{P} \setminus \mathcal{C}$. However, instead of applying this directly, we note that, for $k > 3$, if the point set $\mathcal{C}_-^k \cup \{P\}$ is symmetrical, it must satisfy $E_{\min}(\mathcal{S}^* \cup \{P\}) < E_{\max}(\mathcal{S}^* \cup \{P\})$ for $\mathcal{S}^* = \{P_l, 1 \le l \le k-1\}$. Hence, for each seed set $\mathcal{S} = \{P_k, 1 \le k \le s\}$ generated during the point expansion process, if P_{s+1} is the unique point in $\mathcal{P} \setminus \mathcal{C}$ such that $E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\})$, we know that Eq. (13) will not be satisfied. If more than one point exists in $\mathcal{P} \setminus \mathcal{C}$ such that $E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\})$ is satisfied, we only need to verify Eq. (13) for these points to improve the algorithm's efficiency.

As can be seen from the above, the validity and efficiency of the point expansion approach mainly depends on the computation of $E_{\min}(\mathcal{S}_+)$ and $E_{\max}(\mathcal{S}_+)$ from given $E_{\min}(\mathcal{S})$ and $E_{\max}(\mathcal{S})$, which we discuss next.

5.2.2. *Minimal and maximal tolerance computation*

In this section, an efficient algorithm is given for the computation of $E_{\min}(\mathcal{S}_+)$ and $E_{\max}(\mathcal{S}_+)$ for $P \in \mathcal{P} \setminus \mathcal{S}$, where $\mathcal{S} = \{P_k : 1 \le k \le s\}$ is a given seed set and $\mathcal{S}_+ = \mathcal{S} \cup \{P\}$. Eq. (11) is used for the computation of $D^r_{\min}(\mathcal{S}_+)$, $D^r_{\max}(\mathcal{S}_+)$, $1 \le r \le g(s+1, c)$, from the known values $D^r_{\min}(\mathcal{S})$, $D^r_{\max}(\mathcal{S})$, $1 \le r \le g(s, c)$ where $s = |\mathcal{S}|$ and c is the number of points in the c-fold rotational symmetry that $\mathcal{S}$ may finally produce. While we do not know c in advance, it can be computed during point expansion as described below.

Let $L_k = \|P - P_{s+1-k}\|$, $1 \le k \le s$. From Eq. (2), it can be seen that $g(s+1, c) = g(s, c)$ or $g(s+1, c) = g(s, c)+1$. Using Eq. (11) for the computation of $D^r_{\min}(\mathcal{S}_+)$ and $D^r_{\max}(\mathcal{S}_+)$, we just need to decide which distance class $\mathcal{G}^r(\mathcal{S}_+)$, $1 \le r \le g(s+1, c)$, each L_k, $1 \le k \le s$ should lie in if $\mathcal{S}_+$ is also a seed set. This is based on the following observation depending on the slot P occupies in the symmetry:

1. If $s \le \lfloor c/2 \rfloor$, L_k increases with k (see Fig. 7(a)).
2. If $s > \lfloor c/2 \rfloor$ and c is an odd integer, L_k increases with k at first until it reaches a maximum at $k = \lfloor c/2 \rfloor$ and $k = \lfloor c/2 \rfloor + 1$ and then decreases (see Fig. 7(b)).

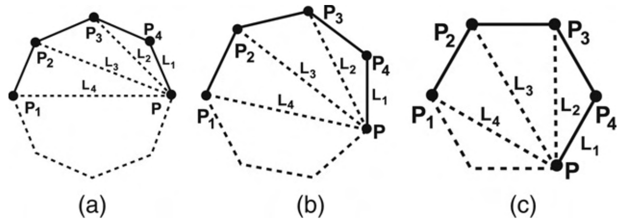

Fig. 7. Various cases of distance classes during the point expansion process.

3. If $s > \lfloor c/2 \rfloor$ and c is an even integer, L_k increases with k at first until it reaches a maximum at $k = \lfloor c/2 \rfloor$ and then decreases (see Fig. 7(c)).

Therefore, in each step of our consecutive point expansion, as s increases, we can determine whether L_k has reached a maximum. If not, we must have $L_k \in \mathcal{G}^k(\mathcal{S}_+)$; otherwise we can now determine c. Once we know c, the computation of $D^r_{\min}(\mathcal{S}_+)$ and $D^r_{\max}(\mathcal{S}_+)$ can be based on Eq. (11). Further details are now explained.

Suppose c is not yet known. In this case, we know that $s \le \lfloor c/2 \rfloor$. Therefore, $g(s, c) = s - 1$ and correspondingly, $D^{s-1}_{\min}(\mathcal{S}) = D^{s-1}_{\max}(\mathcal{S})$. Furthermore, we have $L_k \in \mathcal{G}^k(\mathcal{S}_+)$, $1 \le k \le s-1$. However, L_s may possibly belong to any of $\mathcal{G}^{s-2}$, $\mathcal{G}^{s-1}$ or $\mathcal{G}^s$ for $c = 2(s-1)$, $c = 2s-1$ and $c \ge 2s$ respectively, when $\mathcal{S}_+$ is a seed set. In Fig. 7, for example, given $\mathcal{S} = \{P_1, P_2, P_3, P_4\}$ for a valid expansion point P, cases (a), (b), (c) are those for which L_4 belongs to $\mathcal{G}^4$, $\mathcal{G}^3$ and $\mathcal{G}^2$ respectively. Therefore, in order to decide to which distance class L_s actually belongs, we compute $E_{\min}(\mathcal{S}_+)$ and $E_{\max}(\mathcal{S}_+)$ for the three possibilities of adding L_s to $\mathcal{G}^s$, $\mathcal{G}^{s-1}$ and $\mathcal{G}^{s-2}$. L_s is then added to the class such that $E_{\min}(\mathcal{S}_+)$ has the smallest value while $E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+)$. If L_s is added to $\mathcal{G}^s$, we leave c undetermined, and to get the next expansion point after this one, the above process is repeated. If L_s is added to $\mathcal{G}^{s-1}$, we set $c = 2s-1$. If L_s is added to $\mathcal{G}^{s-2}$, we set $c = 2(s-1)$. Once c has been determined, $D^r_{\min}(\mathcal{S}_+)$ and $D^r_{\max}(\mathcal{S}_+)$ can be computed according to Eq. (11).

Note that we do not estimate c from the angles set up by each consecutive triple of points in $\mathcal{S}$, since it is difficult to say in the approximate case which c-fold rotational symmetry an angle corresponds to without additional information (especially when c is large). The above approach is compatible with our point expansion principle, choosing the point P that makes $E_{\min}(\mathcal{S}_+)$ minimal among all points in $\mathcal{P} \setminus \mathcal{S}$.

The following observation further simplifies the computation required during the point expansion. For a valid expansion point P, $E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+)$ must be satisfied. From Eqs. (6) and (7), we have that

$$\max_{1\le k\le s} (|L_k - D^{k'}_{\min}(\mathcal{S})|, |L_k - D^{k'}_{\max}(\mathcal{S})|) \le E_{\min}(\mathcal{S}_+) < E_{\max}(\mathcal{S}_+) \le E_{\max}(\mathcal{S}),$$

where $k' = k$ when $k \le c$ or $k' = k - c$ when $k > c$. Thus, L_k must lie in the open interval $(D^{k'}_{\min}(\mathcal{S}) - E_{\max}(\mathcal{S}), D^{k'}_{\max}(\mathcal{S}) + E_{\max}(\mathcal{S}))$ for a valid expansion point, and we can ignore any point not meeting this condition. This simple observation greatly reduces the amount of computation required since most points in $\mathcal{P} \setminus \mathcal{S}$ typically fall into this category.

5.3. Initial symmetry seed sets

Finally, we describe in Section 5.3.1 conditions on a proper initial seed set consisting of three points for them to produce a potential cycle, and in Section 5.3.2 how to compute the first expansion point (which must be treated as a special case) from such an initial seed set.

5.3.1. Conditions on initial seed sets

Our point expansion process considers all unordered triples of points in the input point set as candidates for cycle generation. However, Eq. (9) adds further conditions on an initial seed for it to be capable of producing cycles, as we now explain. For this, let an initial seed set triple be the three points P_1, P_2, P_3. Without loss of generality we assume the points are ordered such that $\|P_1 - P_3\| \ge \|P_2 - P_3\| \ge \|P_1 - P_2\|$.

First we consider the possibility that $\mathcal{S}$ possesses a three-fold rotational symmetry. In this case the minimal tolerance for $\mathcal{S}$ is $E_{\min}(\mathcal{S}) = \|P_1 - P_3\| - \|P_1 - P_2\|$, the maximal difference between the edge lengths amongst these points. In order for these points to represent a three-fold symmetry, in which we cannot confuse two points, we require that $E_{\max}(\mathcal{S}) = \|P_1 - P_2\|$. Hence, from Eq. (9),

$$\|P_1 - P_3\| \le 2\|P_1 - P_2\|. \tag{14}$$

Now suppose that $\mathcal{S}$ may be expanded to a symmetric set possessing a c-fold symmetry with $c > 3$ and correspondingly $g(s, c) \ge 2$ (see Eq. (2)). From Eqs. (6) and (7) we obtain

$$E_{\min}(\mathcal{S}) = \|P_2 - P_3\| - \|P_1 - P_2\|,$$
$$E_{\max}(\mathcal{S}) = \|P_1 - P_3\| - \|P_2 - P_3\|.$$

For $\mathcal{S}$ to be a valid seed set, it must satisfy $E_{\min}(\mathcal{S}) < E_{\max}(\mathcal{S})$, i.e.

$$2\|P_2 - P_3\| < \|P_1 - P_3\| + \|P_1 - P_2\|. \tag{15}$$

If only one of Eqs. (14) and (15) is satisfied, we can immediately decide whether the points have three-fold symmetry, or c-fold symmetry for some $c > 3$. However, Eqs. (14) and (15) are frequently *both* satisfied, specifically, if $2\|P_2 - P_3\| < 3\|P_1 - P_2\|$. In this case, $\mathcal{S}$ can be plausibly either a regular triangle *or* a suitable seed set for point expansion; this happens, for example, for the seed set $\mathcal{S} = \{P_1, P_2, P_3\}$ in Fig. 6. In such situations, we output both as potential symmetries detected in the point set (the problem of deciding which one should be present is left for downstream processing, where higher level information may be used; also see [8]).

5.3.2. Initial symmetry seed set expansion

Expansion from the initial seed set $\mathcal{S}$ of three points is different from the general point expansion process for a seed set $\mathcal{S}$ of $|\mathcal{S}| > 3$ points, as we now discuss.

Before considering what may happen when we expand an initial seed set $\mathcal{S} = \{P_1, P_2, P_3\}$ in the approximate case, we first discuss the exact case (i.e. $\epsilon = 0$). Eq. (1) requires that the

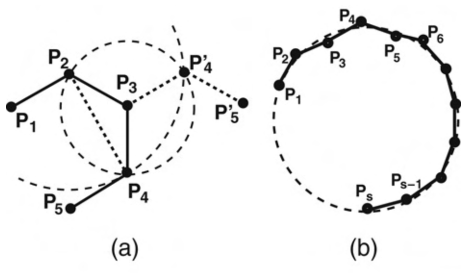

Fig. 8. Special cases for point expansion.

following conditions must be satisfied for an expansion point P_4 with $\epsilon = 0$:

$$\|P_4 - P_3\| = \|P_2 - P_3\| = \|P_1 - P_2\|,$$
$$\|P_4 - P_2\| = \|P_1 - P_3\|. \tag{16}$$

These conditions determine *two* possible locations for P_4, denoted P_4 and P_4': see Fig. 8(a). They are the intersection points of two circles with centre P_3 and radius $\|P_2 - P_3\|$, and centre P_2 with radius $\|P_1 - P_3\|$. However, only P_4 can lead to rotational symmetry and is the desired expansion point. The two points can be distinguished in the exact case by which side of the line P_2P_3 they lie on. Specifically, we choose the point P which lies on the same side of the line P_2P_3 as P_1 as the expansion point, i.e. the point which fulfils

$$((P - P_3) \times (P_2 - P_3)) \cdot ((P_3 - P_2) \times (P_1 - P_2)) > 0. \tag{17}$$

For $\epsilon > 0$ the case is less straightforward, especially if P_1, P_2, P_3 almost lie in a line, which corresponds to a c-fold rotational symmetry with large c. In this case determining the sign of the left hand side of Eq. (17) is ill posed due to numerical instabilities. Furthermore, note that a point which lies on the wrong side of the line may still belong to an approximate symmetry. For example, see Fig. 8(b). Starting from the initial seed set $\mathcal{S} = \{P_1, P_2, P_3\}$, the point P_4 should still be selected even though it lies on the opposite side of P_2P_3 from P_1: a symmetric set with small tolerance can still be generated using this point.

Our solution to this problem is to consider points on both sides of the line and check whether a complete cycle can be found. In the exact case, if we expand P_1, P_2, P_3, P_4 further, we get a potential cycle, while exact expansion from P_1, P_2, P_3, P_4' would produce a zig-zag line, not a cycle. Consider Fig. 8, for example. Applying a further expansion step from both $\mathcal{S} \cup \{P_4\}$ and $\mathcal{S} \cup \{P_4'\}$, we would obtain P_5 and P_5' respectively. $P_1, \ldots, P_5$ form a seed set that may finally lead to a cycle, while P_1, P_2, P_3, P_4', P_5' only yield a zig-zag line.

Now consider the approximate case ($\epsilon > 0$). Denote by $\mathcal{P}^*$ the set of all points fulfilling $E_{\min}(\mathcal{S} \cup \{P\}) < E_{\max}(\mathcal{S} \cup \{P\})$, $P \in \mathcal{P} \setminus \mathcal{S}$. This set is divided into two categories: points $\mathcal{P}_s^*$ lying on the same side of the line P_2P_3 as P_1, and points $\mathcal{P}_d^*$ lying on the opposite side. We choose whichever point minimizes $E_{\min}(\mathcal{S} \cup \{P\})$ in $\mathcal{P}_s^*$, and again in $\mathcal{P}_d^*$, and do further point expansion from both in order not to lose potential symmetries. No unnecessary symmetries will be introduced by this strategy, as further explained. Provided we have no symmetries like those in Fig. 8(b), expansion from the point chosen from $\mathcal{P}_s^*$ produces a cycle (if it exists), while the point from $\mathcal{P}_d^*$ can only give a zig-zag line. If, however, we have a case like Fig. 8(b), the selected points from $\mathcal{P}_s^*$ and $\mathcal{P}_d^*$ may produce two cycles which are only different in the fourth point, and are otherwise very close to each other. Selection between them will be decided by further unambiguity checking—either one of them will be chosen, or neither of them.

Note that this situation can only arise when expanding an initial seed set consisting of three points. If $s = |\mathcal{S}| \geq 4$, P_{s+1} is uniquely determined by the point P that minimizes $E_{\min}(\mathcal{S} \cup \{P\})$ amongst all valid expansion points in $\mathcal{P} \setminus \mathcal{S}$, as illustrated in Section 5.2.1.

6. Unambiguous cycles of 3D point sets

In this section we describe the modifications needed to our 2D unambiguous cycle detection algorithm for 3D point sets. It is again based on the idea of expanding an initial seed set consisting of three points. The main difference is that in the 3D case, we have to consider rotation-reflection cycles, i.e. vertices of an anti-prism, as well as rotation cycles.

6.1. Unambiguous cycle types in 3D

The unambiguous cycles that can arise in 3D are the ordered vertices of the following shapes: tetrahedrons, regular polygons and anti-prisms (alternate vertices taken from opposite ends). Here, again, we start by considering exact symmetries of such types.

The following theorem states the condition under which an initial seed set can *only* be expanded into a point sequence lying in a plane, leading to symmetry defined by a regular polygon: the fourth expansion point lies on the plane determined by the initial seed set. As in the 2D case, expansion points from an initial seed set are still determined by the distance constraints as prescribed by Eq. (1) for $\epsilon = 0$ or equivalently $E_{\min}(\mathcal{S}) = 0$.

Theorem 4. *Assume throughout the tolerance $\epsilon = 0$. Let $\mathcal{S} = \{P_k : 1 \leq k \leq 4\} \subset \mathbb{E}^3$ be a coplanar seed set. The next expansion point P_5 is uniquely determined by $\mathcal{S}$ such that $\mathcal{S} \cup \{P_5\}$ is a seed set, and any further expansion of $\mathcal{S}$ results in a coplanar set.*

Proof. A seed set is only prescribed by the distances between its points. From the analysis in 2D, P_4 may lie to the left or right of the line P_2P_3 in the plane P_1, P_2, P_3. We assume that P_4 lies on the same side of the line P_2P_3 as P_1. The proof of the case where P_4 lies on the other side proceeds in a similar manner. From Eq. (1), the fifth expansion point from $\mathcal{S}$ is determined by

$$\|P_5 - P_4\| = \|P_1 - P_2\|, \qquad \|P_5 - P_3\| = \|P_1 - P_3\|,$$
$$\|P_5 - P_2\| = \|P_1 - P_4\|. \tag{18}$$

This means that P_5 lies at the intersection of three spheres: S_1 with centre P_4 and radius $\|P_1 - P_2\|$, S_2 with centre P_3 and radius $\|P_1 - P_3\|$, S_3 with centre P_2 and radius $\|P_1 - P_4\|$. S_1 and S_2 always intersect in a circle C with centre O on the line P_3P_4 (see Fig. 9(a)). Let P_5 and P_5' be the intersection

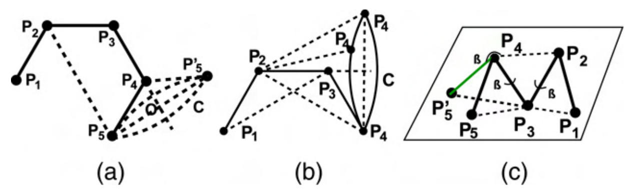

Fig. 9. Analysis of 3D point expansion.

points between circle C and plane P_1, P_2, P_3. P_5 lies on the same side of line P_3P_4 as P_2 and P'_5 lies on the other side. P_5 is the unique expansion point we are looking for. Firstly, $\|P_5 - P_2\| = \|P_1 - P_4\|$, as the polygons P_1, P_2, P_3, P_4 and P_2, P_3, P_4, P_5 are congruent. Next, as the line P_2P_5 is parallel to P_3P_4 and P_3P_4 is perpendicular to the plane the circle C lies in, line P_2P_5 is also perpendicular to the plane of C. Thus, the distance between P_2 and all other points on C than P_5 is greater than $\|P_2 - P_5\|$. Thus P_5 is the unique point satisfying Eq. (18). □

We now give a further analysis for other 3D symmetries.

Let $\{P_1, P_2, P_3\}$ be an initial seed set of 3D points (see Fig. 9(b)), and P_4 be the next expansion point determined by Eq. (1) at $\epsilon = 0$. Hence, P_4 must lie on the circle C formed by the intersection of two spheres, with centre P_3 and radius $\|P_2 - P_3\|$, and P_1 with radius $\|P_1 - P_3\|$, respectively. Depending on the different locations of P_4, different types of symmetry are possible.

Clearly, a regular tetrahedron is produced if all the distances between each pair of points are the same. As noted in Theorem 4, if P_4 lies on the plane $P_1P_2P_3$, the points can belong to a regular polygon. Finally, suppose P_4 does not lie in the plane $P_1P_2P_3$, and the dihedral angle (at edge P_2P_3) between the adjacent triangles $P_1P_2P_3$ and $P_2P_3P_4$ is β. Continuing from P_1, P_2, P_3, P_4 there are two possible expansion points, P_5 and P'_5, such that Eq. (18) is satisfied (see Fig. 9(c)). P_5 and P'_5 lie on different sides of plane $P_2P_3P_4$ and they determine two triangles $P_3P_4P_5$ and $P_3P_4P'_5$ respectively, each of which makes a dihedral angle β with triangle $P_2P_3P_4$ along edge P_3P_4. Suppose P_5 lies on the same side of the plane $P_2P_3P_4$ as P_1 while P'_5 on the other side. In a similar manner to the 2D case, further expanding $(P_1, P_2, P_3, P_4, P'_5)$ produces a zigzag line in 3D space, which is caused by evenly distributed points on a helix (leading to an infinite screw symmetry, which we ignore here). In the other case, $(P_1, P_2, P_3, P_4, P_5)$ leads to a uniquely determined rotation-reflection symmetry case.

In 3D, when finding unambiguous cycles involving the detection of the orbit of a single point, there are two cases to consider: isometries which preserve orientation and isometries which invert the orientation of space. The orientation preserving isometries are rotations giving regular polygons as unambiguous cycles. The orientation inverting isometries consist of combinations of rotations and reflections, which means we have to also consider elementary rotation-reflection symmetries as described above.

6.2. *Point expansion in 3D in the approximate case*

Section 6.1 analyses the cycles that may arise in 3D in the exact case. Based on these ideas, we now explain how similar cycles can be obtained in 3D in the approximate case using a point expansion approach starting from an initial three-point seed set. In particular, we illustrate how the point expansion process used in the 2D case can be applied in 3D to rotation-reflection symmetry detection.

The core algorithm used for 3D unambiguous cycle detection is similar to the one used in 2D. Given an initial seed set $\mathcal{S} = \{P_1, P_2, P_3\}$ taken from the input point set $\mathcal{P} \subset \mathbb{E}^3$, we choose all possible expansion points P_4 and then expand further from each of them if possible. As noted in Section 6.1, the fifth expansion point can be in two different positions of which only one can lead to a valid unambiguous cycle. Thus, following a similar approach to the 2D case, we divide all potential expansion points into two sets according to which side of the plane $P_2P_3P_4$ they lie on and use the same reasoning as in the 2D case to find in each set an optimal expansion point leading to an unambiguous set, if one exists. Finally, its unambiguity condition is checked as in the 2D case.

However, applying this expansion process directly prevents certain rotation-reflection symmetries from being detected. This is because for a regular polygon, the distance equivalence classes as defined by Eq. (3) increase (in the sense of their elements) as r (the index difference between points as used in Eq. (3)) increases, but this is not satisfied for an anti-prism. We handle this case by distinguishing distance classes that contain distances between points at opposite ends of an anti-prism from classes containing distances between points from the same end of the anti-prism, as we now explain.

We recall some ideas from 2D point expansion: we start with a 2D seed set $\mathcal{S}$, $s = |\mathcal{S}|$ with maximal and minimal distances $D^r_{\min}(\mathcal{S})$, $D^r_{\max}(\mathcal{S})$, $1 \le r \le g(s, c)$ for each distance class as defined by Eq. (4). For any point $P \in \mathcal{P} \setminus \mathcal{S}$ we let $L_k = \|P - P_{s+1-k}\|$, $1 \le k \le s$. The point expansion algorithm is based on Theorem 1, which states that the next expansion point P must satisfy $E_{\min}(\mathcal{S} + \{P\}) < E_{\max}(\mathcal{S} + \{P\})$. The computation of $E_{\min}(\mathcal{S} + \{P\})$, $E_{\max}(\mathcal{S} + \{P\})$ involves two important monotonicity conditions:

(i) $D^r_{\min}(\mathcal{S}) \le D^r_{\max}(\mathcal{S}) < D^{r+1}_{\min}(\mathcal{S}) \le D^{r+1}_{\max}(\mathcal{S})$ for $\mathcal{S}$ to be a valid seed set, which can be seen from Eq. (5);

(ii) L_k increases with k at first until a maximum is reached at $k = \lfloor \frac{c}{2} \rfloor$ and then decreases (see Section 5.2.2).

In 3D, however, these monotonicity properties are in general not satisfied by a point set with rotation-reflection symmetry. For example, consider the points forming a rotation-reflection symmetry in Fig. 10. We may have $\|P_5 - P_3\| < \|P_5 - P_4\|$ or even $\|P_5 - P_1\| < \|P_5 - P_4\|$ if we increase the distance between the points on the top plane and bottom plane along the normal direction. In such cases $E_{\min}(\mathcal{S} + \{P\})$, $E_{\max}(\mathcal{S} + \{P\})$ have to be handled differently so that Theorem 1 remains valid.

When rotation-reflection symmetry is present, if we split each set $D^r_{\min}(\mathcal{S})$, $D^r_{\max}(\mathcal{S})$ and L_k into subsets with even and odd indices, denoted as $D^{r,\mathrm{o}}_{\min}(\mathcal{S})$, $D^{r,\mathrm{e}}_{\min}(\mathcal{S})$, $D^{r,\mathrm{o}}_{\max}(\mathcal{S})$, $D^{r,\mathrm{e}}_{\max}(\mathcal{S})$ and L^{o}_k, L^{e}_k, we can see that a monotonicity condition similar to the 2D condition (conditions (i) and (ii) listed above) is separately satisfied by each subset. We can then consider the even and odd cases separately first, and then

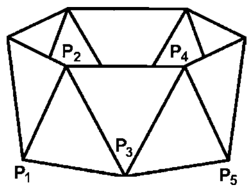

Fig. 10. Crown symmetry detection.

combine them together for rotation-reflection cycle generation. Specifically, for each point $P \in \mathcal{P} \setminus \mathcal{S}$, we first compute from $D^{r,\mathrm{o}}_{\min}(\mathcal{S})$, $D^{r,\mathrm{o}}_{\max}(\mathcal{S})$, L^{o}_k and $D^{r,\mathrm{e}}_{\min}(\mathcal{S})$, $D^{r,\mathrm{e}}_{\max}(\mathcal{S})$, L^{e}_k as described in Section 5.2.2 to give $E^{\mathrm{o}}_{\min}(\mathcal{S} \cup \{P\})$, $E^{\mathrm{o}}_{\max}(\mathcal{S} \cup \{P\})$ and $E^{\mathrm{e}}_{\min}(\mathcal{S} \cup \{P\})$, $E^{\mathrm{e}}_{\max}(\mathcal{S} \cup \{P\})$ respectively. If $E^{x}_{\min}(\mathcal{S} \cup \{P\}) \geq E^{x}_{\max}(\mathcal{S} \cup \{P\})$ for $x = \mathrm{o}$ or $x = \mathrm{e}$, the point P is not a valid expansion point. Otherwise, we next set $E_{\min}(\mathcal{S} \cup \{P\}) = \max(E^{\mathrm{o}}_{\min}(\mathcal{S} \cup \{P\}), E^{\mathrm{e}}_{\min}(\mathcal{S} \cup \{P\}))$ and $E_{\max}(\mathcal{S} \cup \{P\}) = \min(E^{\mathrm{o}}_{\max}(\mathcal{S} \cup \{P\}), E^{\mathrm{e}}_{\max}(\mathcal{S} \cup \{P\}))$ to maintain the interval of valid tolerances. If $E_{\min}(\mathcal{S} \cup \{P\}) \geq E_{\max}(\mathcal{S} \cup \{P\})$, the point is again not a valid expansion point. A similar idea is used to generalize the unambiguity verification from 2D to 3D using the same computation for $E_{\min}(\mathcal{S} \cup \{P\})$.

Finally, based on the above analysis, we note that for a rotation-reflection symmetry, it may be possible that $\|P_1 - P_3\| < \|P_1 - P_2\|$ for the initial seed set. So the condition on finding the initial three points for the seed set is relaxed to $\max(|\|P_1 - P_3\| - \|P_1 - P_2\||, |\|P_1 - P_3\| - \|P_2 - P_3\||) < \epsilon^*$ with $\epsilon^* = \min(\|P_1 - P_2\|, \|P_2 - P_3\|, \|P_1 - P_3\|)$.

7. Time complexity

The time complexity of our algorithm is $O(Cn^4)$, where n is the number of points in the input point set and C is the maximal symmetry order present (usually a small integer). This is easily seen as all triples of points are taken as seed points, and then each remaining point is considered as an expansion point at each step until we have a cycle (or we exit without finding one).

Brass' [18] presents a deterministic algorithm to find all exact local symmetries in $O(n^{2.136+\epsilon})$ time. Aloupis et al. [25] improve this using a randomised algorithm which can detect regular polygons with high probability in $O(n^{2.068+\epsilon})$ time. Although the time order of these algorithms is smaller, they only solve the exact problem and approximate symmetry detection is much more complex as global properties have to be considered. No previous work on approximate *local* symmetry (i.e. regular polygon) detection exists as far as we know. Finding *global* approximate symmetry is NP-hard under the definition of Iwanowski [27]. Under other different definitions provided to facilitate detection, approximate global symmetry detection requires high-order polynomial time: $O(n^8)$ using Alt et al.'s method [22] and $O(n^{3.5} \log^4 n)$ using Mills et al.'s method [6]. In this context, we consider time $O(n^4)$ to be acceptable.

For large point sets, in the approximate case, there is potentially a large number of unambiguous cycles. To increase the speed of the algorithm, as well as the usefulness of the answers, we suggest that feature detection algorithms, such as regularity feature trees [30], may help to reduce the complexity of the problem for practical use, by dividing the input into several smaller sub-problems.

8. Experiments

In this section we present various examples of using our approach to detect unambiguous cycles in 2D and 3D point sets. The algorithm was implemented on Linux in Matlab running on a Pentium 4 3.4 GHz with 1GB RAM. We would expect a C++ implementation to be much quicker than this Matlab prototype. Section 8.1 considers 2D cases, and Section 8.2 shows cases derived from 3D CAD models.

Note that if $\mathcal{C}$ has a c-fold rotational symmetry at tolerance ϵ, by definition it must have subsets displaying a c'-fold rotational symmetry with tolerance not greater than ϵ for any factor $c' \geq 1$ of c. For clarity of presentation, such symmetries are not shown in these examples although they are also output. Furthermore, we do not show the detected three-fold symmetries or 3D symmetries consisting of four points, of which there are usually many.

Most spurious symmetries are only present at large tolerance values and can easily be suppressed if a user-supplied maximal tolerance of interest is given. Often it is very simple to select a suitable cut-off tolerance after the symmetries at various tolerances have been detected. However, as we are detecting approximate symmetries, not all symmetries at small tolerance are automatically also symmetries which are intended. Our algorithm detects symmetries which are unambiguously present in the data at various tolerance levels. Downstream processes must decide which of these symmetries should be considered for further processing.

8.1. 2D examples

This section discusses the results of determining 2D unambiguous cycles in various examples.

First, we considered the algorithm's ability to detect intended symmetries in sets of points derived from regular shapes. We took 25 points from the vertices of a twelve-fold regular polygon of radius 1.0, one five-fold regular polygon of radius 0.5 and two squares of radii 0.5 and 0.2. The twelve-fold regular polygon was placed at the origin and the others centre at various locations. In the example shown, these centres were as $(0. -0.25)$, $(-0.2, 1.05)$, $(1.5, 0.2)$, chosen to carefully place points from different polygons close to each other so we could assess the quality of the results produced by the point expansion process. The coordinates of the points in each original exact regular polygon were disturbed by uniformly distributed random errors less than 0.01, 0.04, 0.033 and 0.01 respectively. Overall 30 randomly generated arrangements of this type were used for testing, one of which is shown in Fig. 11. The left image shows the symmetries used to generate the set, which were also detected successfully, and the right image shows additional unintended symmetries also detected. The tolerances at which these were present are also given in the top-right legends.

All four initial polygons were detected in each test, taking an average time of 2.5s. Furthermore, at most five additional

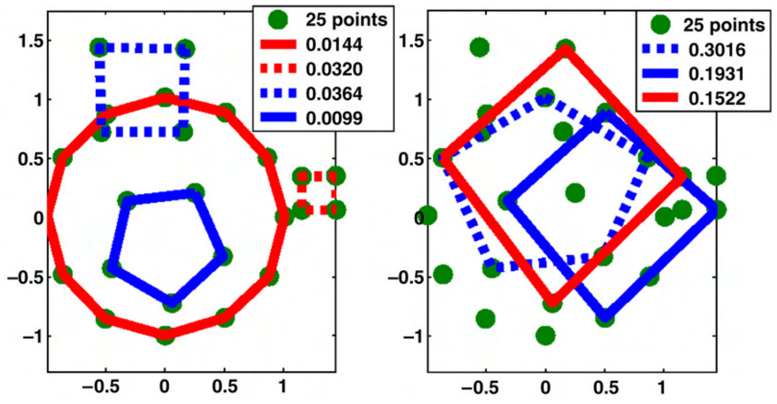

Fig. 11. 2D points with intended (left) and unintended (right) unambiguous cycles; running time for this example: 2.37 s.

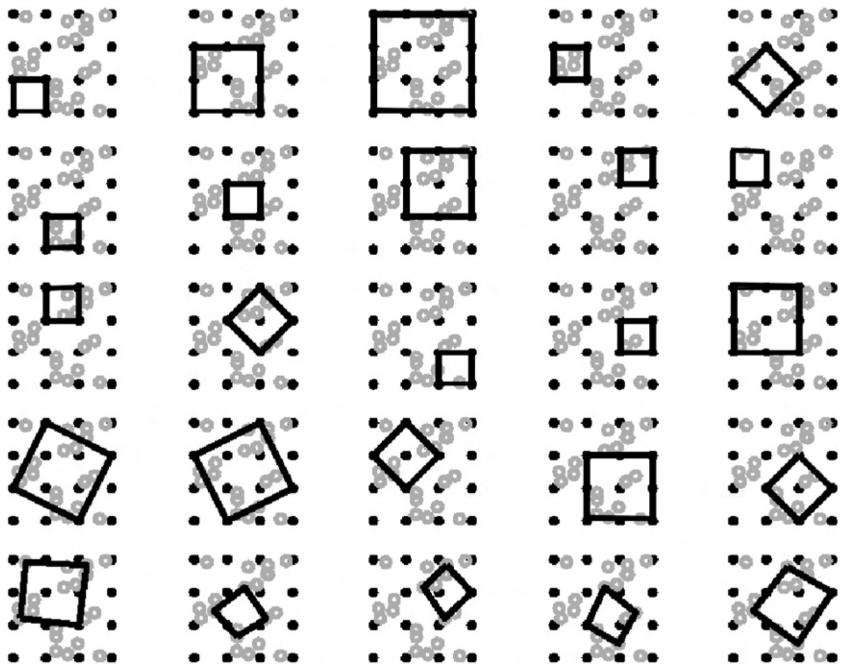

Fig. 12. 2D points on a 4 × 4 regular grid and 20 random points.

symmetries were detected. As we are looking for approximate symmetries, such additional sets cannot be avoided in general. As can be seen in Fig. 11, the unintended symmetries not only have much larger tolerances (at least 10 times larger) than the intended symmetries, but also are inconsistent with some of the intended symmetries—for example, three points are shared by the intended twelve-fold symmetry and the unintended five-fold symmetry. The tolerance information can be used in combination with a consistency check to separate the intended from the unintended symmetries (see [8]).

The example in Fig. 11 shows the algorithm's stability with respect to noisy point sets with intended symmetries. To test the algorithm's behaviour given random points, examples like Fig. 12 were also considered. 36 points were generated of which 16 were arranged on a 4 × 4 regular grid in [0, 1] × [0, 1] (dark points) and the other 20 were chosen randomly with a uniform probability distribution in the same area (light points). These light points disturb the symmetries generated by the dark points. Over 30 tests on such arrangements were executed. Within at most 10s, the 20 square symmetries induced by the points on the 4 × 4 grid were always detected at tolerance zero (see the first four rows of Fig. 12). The random points led to several other symmetries (at most five per test) also being detected; examples being given in the last row, with tolerances of 0.0459, 0.0471, 0.0570, 0.0801, 0.1063 respectively.

Tests on uncorrelated random points uniformly distributed in [0, 1] × [0, 1] were also performed to test the algorithm's performance in detecting unintended symmetries. The results of 30 tests using 30 random points are summarized in Fig. 13. The black line gives the running time in seconds, the red line shows the number of detected symmetries, and the blue line indicates approximate tolerances multiplied by 100 in each test. As seen from the figure, in five tests no symmetry was found, in 16 tests either one or two symmetries were found, in seven tests either three or four symmetries, and in two tests six symmetries were detected. The tolerances of the symmetries found range from 0.0159 to 0.0838. Such good performance strongly suggests that only few unintended symmetries, as detected by our

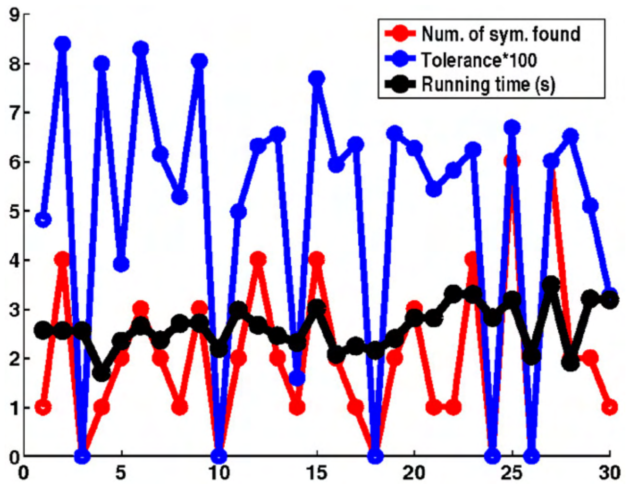

Fig. 13. Experimental results on 30 tests of 30 2D random points. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

algorithm, are introduced by randomly distributed points. All the tests took 2–4s to compute, which shows the algorithm's efficiency.

In summary, these and other 2D examples show that our algorithm can robustly detect all potential intended 2D symmetries within one minute for up to about 100 points. Also note that for many typical CAD models, planar symmetries dominate in engineering, e.g. see the complex model in Fig. 1. In such cases, the 2D symmetry detection algorithm can be used directly if these planes are detected explicitly beforehand.

8.2. 3D examples

In this section we show the performance of the 3D version of our algorithm on discrete point sets extracted from CAD models.

The MISUFA model in Fig. 15(a) came from Cadalog, Inc., http://www.ohyeahcad.com/, and the other two in Figs. 1 and 14(a) from the National Design Repository, http://www.designrepository.org/. For simplicity, the discrete points used in our tests here were the model vertices instead of

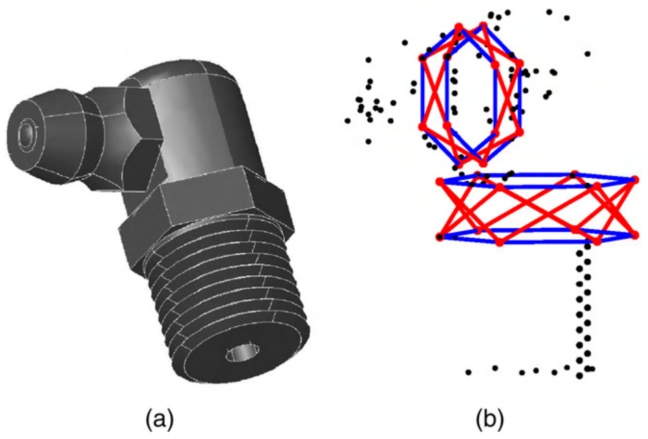

Fig. 15. The symmetries of extracted points from the MISUFA model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

being the complete set of characteristic points discussed earlier. These exact models were perturbed using uniformly distributed random errors dependent on the objects' sizes; details are given later. This allows us to compare the symmetries that our algorithm detects with the intended symmetries that really exist.

In the first test, 68 discrete points were extracted from the model in Fig. 14(a). The maximal and minimal distances between these distinct points are 2.3607 and 0.054 respectively. These extracted points were then disturbed by uniformly distributed random errors less than 0.01. It took 38.9s to detect the four hexagonal symmetries shown in Fig. 14(b). The same symmetries are obtained when running our algorithm on the exact point set yielding tolerances 0, 0, 0.0154, 0.0154 for the hexagonal symmetries in 36.2s. The two symmetries at tolerance zero correspond to the two in Fig. 14(b) drawn in dashed lines. The non-zero tolerances for the other two symmetries originate from the actual model. To further check the algorithm performance, the extracted points were disturbed by larger errors: uniformly distributed random errors of less than 0.05. Note that such errors are quite close to the minimal inter-point distance 0.054. Then, only two hexagonal symmetries were detected respectively at tolerance 0.114 and

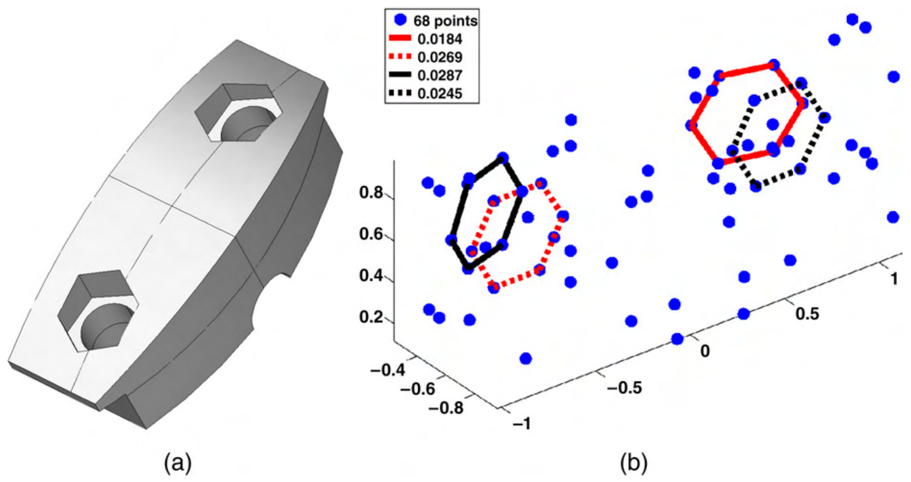

Fig. 14. The symmetries of extracted points from the Vise-guide model.

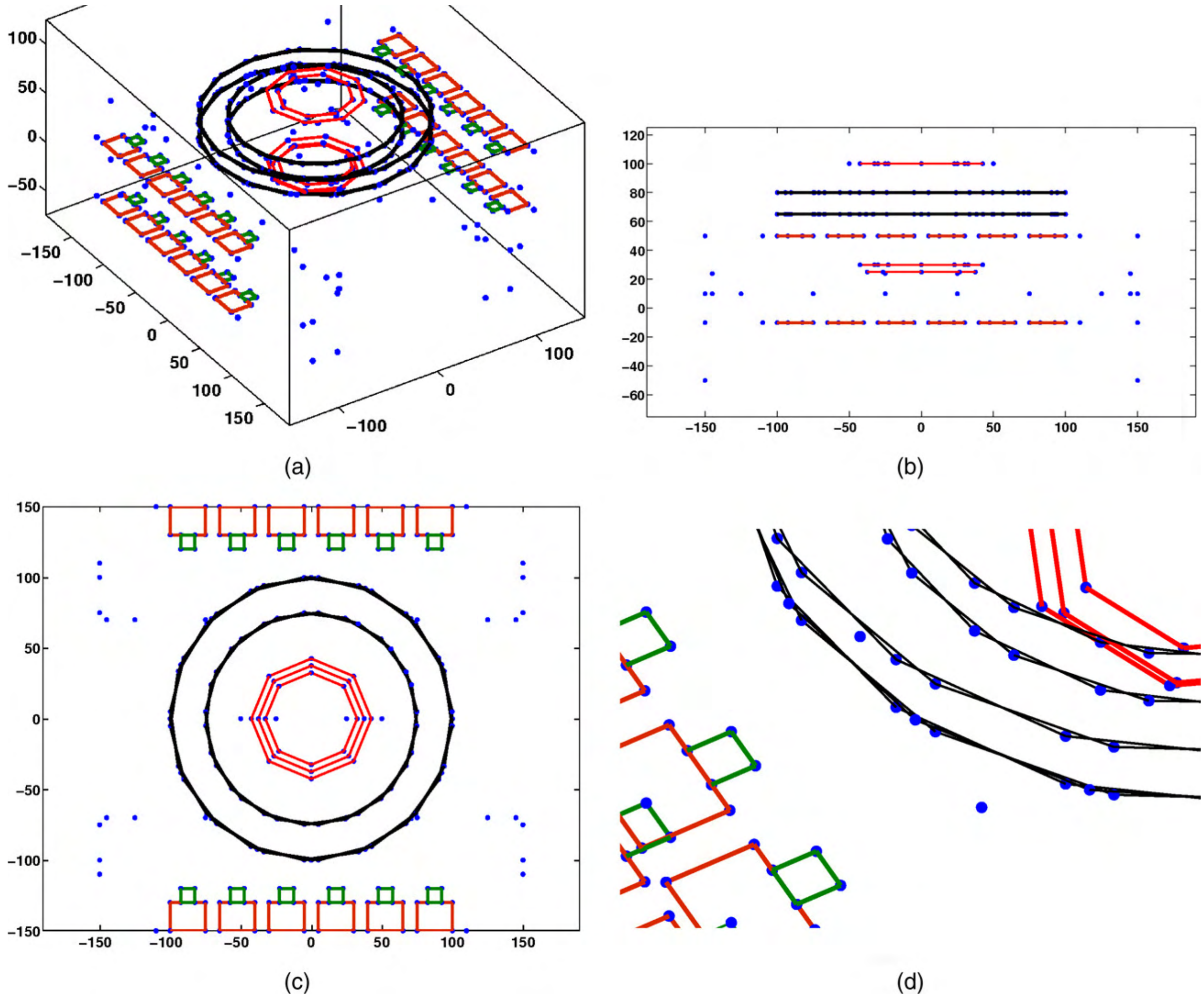

Fig. 16. The symmetries of extracted points from the Monster model (see Fig. 1). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

0.122, which corresponds to the two symmetries originally at tolerance zero (in dashed lines). The other two symmetries were no longer detected, as their regular distribution were destroyed by such large induced errors. This demonstrates that our algorithm is capable of detecting intended symmetries at large tolerances, but also only finds unambiguous symmetries and yields overall fewer unintended symmetries. No other unintended symmetries were detected during all the tests.

In the next example, the 123 vertices of the MISUFA model in Fig. 15(a) were disturbed by uniformly distributed random errors of less than 0.02. (Maximal and minimal inter-point distances in the original exact model are 2.3607 and 0.054 respectively). All the eight detected symmetries are shown in Fig. 15(b) at similar tolerances on average 0.0247. The model has no global symmetries. The cycles detected by our algorithm include four rotational cycles (blue lines) and four rotation-reflection cycles (red lines). The rotational symmetries are obvious hexagonal parts of the model; the rotation-reflection symmetries come from combination of rotational symmetries.

We also tested our algorithm with the Monster part in Fig. 1. A total of 438 points were extracted. As other examples, these points were disturbed by uniformly distributed random errors of less than 0.1. To improve the algorithm performance, we first divided the extracted points into 8 regions by planes orthogonal to the z coordinate axis at z heights -25, 0, 20, 40, 60, 70, 90 (see Fig. 16(c)); the original z coordinates were in $[-50, 100]$. (This planar division is straightforward for a user to do and is a reasonable approach, as planar symmetries dominate in engineering, and rotation-reflection symmetries also consist of two planar symmetries).

Using this approach, it took 345s to detect 58 unambiguous cycles. An overall view, a top view, a side view and a close-up view are shown in Fig. 16(a), (b), (c) and (d) respectively. The detected symmetries can be explained as (i) the five eight-fold rotational symmetries, drawn in red, with an average tolerance 0.094, which come from the eight cylindrical holes; (ii) the five sixteen-fold rotational symmetries, drawn in black, with average tolerance 0.1764, which come from the 16 slots around the centre; (iii) the 24 smaller square symmetries along the sides, drawn in green, with average tolerance 0.1364, from the square slots at the sides; (iv) the 24 large square symmetries along the sides, drawn in red, with average tolerance 5.0100, from the rectangular slots at the sides. As seen from the model, the first three types at smaller tolerance are all intended symmetries, while the fourth type were originally rectangles but detected as squares at a large tolerance.

In a manner similar to the above examples, we have tested our algorithm with various other approximate CAD models, most of which were obtained from the National Design Repository, http://www.designrepository.org/. They were perturbed by uniformly distributed random errors. These models include simple models such as cubes with some blends or complex models such as the Monster model (Fig. 1). The number of discrete points extracted from these models ranged from 36 to 438 and it took between 5s and 345s to detect symmetries. In each case, all intended symmetries that could be detected at zero tolerance in the original CAD model were detected as unambiguous cycles from the disturbed extracted points. In addition to these symmetries, other unintended symmetries were also detected; in all cases there was clear evidence in the model for their presence. Most unintended symmetries, however, were present at larger tolerances than the intended symmetries and, hence, could easily be identified.

Depending on the amount of noise added to the points compared to the distances between the points, some unintended symmetries were at the same tolerances as the intended symmetries. For these a downstream process would have to employ a more sophisticated selection process, e.g. as discussed in [8]. When we added a larger amount of noise (on the level of the smallest distance between points), some intended symmetries were not detectable anymore. We note that such cases cannot in general be avoided when detecting approximate symmetries and our algorithm only detects symmetries for which there is clear, unambiguous evidence present in the model.

Again, we note the advantage of our algorithm that instead of selecting an arbitrary tolerance in advance, the algorithm selects suitable tolerance levels as the symmetries are detected.

9. Conclusion

In this paper a novel definition of approximate subset symmetries of discrete point sets has been given, together with an algorithm to find them. It is suitable for finding approximate symmetries in B-rep models, by carefully choosing characteristic points from the model, including vertices and other special points. Our algorithm avoids generating large numbers of spurious approximate symmetries by automatically deducing tolerance levels at which subsets unambiguously exhibit symmetries. Our experiments demonstrate the ability of our algorithm to find all expected symmetries. Only few unintended symmetries are detected when tested in practice on CAD models. The latter are almost always found at larger tolerance values, giving a simple criterion to decide which symmetries are likely to be intended by the designer. Our algorithm takes time $O(Cn^4)$ for a point set with n points, and maximal symmetry order C. In practice, our prototype Matlab implementation detects symmetries in an acceptable amount of time, in the context of a complete reverse engineering process, for example; furthermore we would expect a C++ implementation to be considerably faster.

In this paper we detect unambiguous cycles which describe orbits of points only. In future work we intend to consider the problem of combining such orbits into sets which approximately share the same geometric symmetry—for example, the five eight-fold symmetries in Fig. 16 can actually be merged into a single eight-fold rotational symmetry. We also intend to investigate the use of this algorithm for design intent detection.

Acknowledgements

We thank the reviewers for their comments, which have helped to improve the paper.

References

[1] Mills B, Langbein F, Marshall A, Martin R. Estimate of frequencies of geometric regularities for use in reverse engineering of simple mechanical components. Tech. rep. GVG 2001–1, Geometry and Vision Group, Dept. Computer Science, Cardiff University, http://ralph.cs.cf.ac.uk/papers/Geometry/survey.pdf;2001.
[2] Thompson DW. On growth and form. Cambridge University Press; 1917. p. 1942.
[3] Barratt K. Logic and design in art, science and mathematics. London: Herbert Press; 1989.
[4] Varady T, Martin R, Cox J. Reverse engineering of geometric models — an introduction. Computer-Aided Design 1997;29(4):255–68.
[5] Pratt M, Anderson B, Ranger T. Towards the standardized exchange of parameterized feature-based CAD models. Computer-Aided Design 2005; 37(12):1251–65.
[6] Mills B, Langbein F, Marshall A, Martin R. Approximate symmetry detection for reverse engineering. In: Proc. 6th ACM symp. solid and physical modeling. 2001. p. 241–8.
[7] Gao C, Langbein F, Marshall A, Martin R. Approximate congruence detection of model features for reverse engineering. In: Proc. int. conf. shape modelling and applications. 2003. p. 69–77.
[8] Langbein F, Marshall D, Martin R. Choosing consistent constraints for beautification of reverse engineered geometric models. Computer-Aided Design 2004;36(3):261–78.
[9] Reisfeld D, Wolfson H, Yeshurun Y. Context-free attentional operators: The generalized symmetry transform. International Journal of Computer Vision 1995;14(2):119–30.
[10] Sun C, Sherrah J. 3D symmetry detection using the extended Gaussian image. IEEE Transactions on Pattern Analysis and Machine Intelligence 1997;19(2):164–8.
[11] Shen D, Ip H, Cheung K, Teoh E. Symmetry detection by generalized complex (GC) moments: A close-form solution. IEEE Transactions on Pattern Analysis and Machine Intelligence 1999;21(5):466–76.
[12] Loy G, Eklundh J. Detecting symmetry and symmetric constellations of features. In: ECCV 2006, Part II. LNCS, vol. 3952. Springer-Verlag; 2006. p. 508–21.
[13] Martinet A, Soler C, Holzschuch N, Sillion FX. Accurate detection of symmetries in 3D shapes. ACM Transactions on Graphics 2006;25(2): 439–64.
[14] Podolak J, Shilane P, Golovinskiy A, Rusinkiewicz S, Funkhouser T. A planar-reflective symmetry transform for 3D shapes. ACM Transactions on Graphics 2006;25(3):549–59.
[15] Mitra N, Guibas L, Pauly M. Partial and approximate symmetry detection for 3D geometry. ACM Transactions on Graphics 2006;25(3):560–8.
[16] Lockwood E, Macmillan R. Geometric symmetry. Cambridge University Press; 1978.
[17] Sugihara K. An $n \log n$ algorithm for determining the congruity of polyhedra. Journal of Computer and System Sciences 1984;29(11):36–47.
[18] Brass P. On finding maximum-cardinality symmetric subsets. Computational Geometry 2003;24:19–25.

[19] Gao X, Lin Q, Zhang G. A C-tree decomposition algorithm for 2D and 3D geometric constraint solving. Computer-Aided Design 2006;38(1):1–13.

[20] Kanatani K. Comments on 'symmetry as a continuous feature'. IEEE Transactions on Pattern Analysis and Machine Intelligence 1997;19(2): 246–7.

[21] Wolter J, Woo T, Volz R. Optimal algorithm for symmetry detection in two and three dimensions. The Visual Computer 1985;1:37–48.

[22] Alt H, Mehlhorn K, Wagener H, Welzl E. Congruence, similarity and symmetries of geometric objects. Discrete Computational Geometry 1988;3:237–56.

[23] Jiang X, Yu K, Bunke H. Detection of rotational and involutional symmetries and congruity of polyhedra. The Visual Computer 1996;12(4): 193–201.

[24] Brass P, Knauer C. Testing congruence and symmetry for general 3-dimensional objects. Computational Geometry 2004;27:3–11.

[25] Aloupis G, Cardinal J, Collette S, Iacono J, Langerman S. Where to build a temple, and where to find one. In: 22nd European workshop on computational geometry. 2006. p. 1–4.

[26] Tate S, Jared G. Recognising symmetry in solid models. Computer-Aided Design 2003;35(7):673–92.

[27] Iwanowski S. Testing approximate symmetry in the plane is NP-hard. Theoretical Computer Science 1991;80:227–62.

[28] Zabrodsky H, Pelog S, Avnir D. Symmetry as a continuous feature. IEEE Transactions on Pattern Analysis and Machine Intelligence 1995;17(12): 1154–66.

[29] Martin R, Dutta D. Tools for asymmetry rectification in shape design. Journal of Systems Engineering 1996;6:98–112.

[30] Li M, Langbein FC, Martin RR. Constructing regularity feature trees for solid models. In: Proc. geometric modeling and processing. Springer; 2006. p. 267–86.