Computer Aided Geometric Design 98 (2022) 102138

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Rotational symmetry detection in 3D using reflectional symmetry candidates and quaternion-based rotation parameterization ☆

Lukáš Hruda [a,*], Ivana Kolingerová [a], Miroslav Lávička [b,c], Martin Maňák [c,a]

[a] Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 8, 301 00 Plzeň, Czech Republic
[b] Department of Mathematics, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 8, 301 00 Plzeň, Czech Republic
[c] New Technologies for the Information Society (NTIS), University of West Bohemia, Univerzitní 8, 301 00 Plzeň, Czech Republic

ARTICLE INFO

Article history:
Received 17 March 2022
Accepted 1 August 2022
Available online 10 August 2022

Keywords:
Symmetry detection
Rotation
Rotational symmetry
Rotation parameterization
Quaternion

ABSTRACT

The property of symmetry in 3D objects is helpful in various applications such as object alignment, compression, symmetrical editing or reconstruction of incomplete objects. However, its robust and efficient detection is a challenging task. The two most commonly occurring types of symmetry are probably reflectional and rotational symmetry. While reflectional symmetry detection methods are quite plentiful, this does not seem to be the case with rotational symmetry detection. In this paper a use of approximate reflectional symmetries to derive plausible approximate rotational symmetries is proposed that can be integrated with multiple different approaches for reflectional symmetry detection. One such specific approach, based on maximizing a given symmetry measure, is chosen and combined with this idea. A modification of the maximization step for rotations is further proposed using a simple, yet efficient, quaternion-based parameterization of the rotation transformation which seems novel in the field of symmetry detection. The results confirm that this combination provides a robust and efficient solution for finding rotational symmetry in a 3D point set and can handle approximate symmetry, noisy input or even partial data.

1. Introduction and related work

Both real-world and artificial 3D objects often possess some form of symmetry. By perfect symmetry of a given 3D object X, we understand a geometric transformation $\mathbf{T}$ (excluding identity) such that $\mathbf{T}(X) = X$, i.e., the object is invariant under the given transform. In practice, however, symmetries are never perfect but only approximate and never precisely satisfy the above definition. By approximate or weak symmetry of an object X we understand a transform $\mathbf{T}$ such that $\mathbf{T}(X)$ approximately matches with X. The knowledge of symmetry can be useful in variety of applications, e.g., object alignment Podolak et al. (2006); Li et al. (2016), compression Simari et al. (2006); Mitra et al. (2006), symmetrical editing Martinet et al. (2006), finding correspondences between related shapes of varying geometry Tevs et al. (2014) or reconstruction of

☆ Editor: Thomas Takacs.

* Corresponding author.
E-mail addresses: hrudalu@kiv.zcu.cz (L. Hruda), kolinger@kiv.zcu.cz (I. Kolingerová), lavicka@kma.zcu.cz (M. Lávička), manak@ntis.zcu.cz (M. Maňák).

https://doi.org/10.1016/j.cagd.2022.102138
0167-8396/

partial objects Sipiran et al. (2014); Mavridis et al. (2015); Schiebener et al. (2016); Sipiran (2017, 2018). Symmetry has also been of interest in the field of geometric morphometrics in biology Savriama and Klingenberg (2011); Klingenberg (2015) where it is used, e.g., for partitioning biological variation into symmetric and asymmetric components Savriama and Gerber (2018). To use symmetry in any application, it first needs to be found and, since in most cases we cannot count on perfect symmetries, robust symmetry detection algorithms are required. Such algorithms should handle detecting weak symmetries, be robust to noise and reasonably fast, and they should not put too many constraints on the input data. The object reconstruction application also requires symmetry detection to work on objects where some parts are missing. Designing algorithms for symmetry detection that satisfy all these requirements to a reasonable extent is challenging.

There are various types of symmetries but the two most common types in 3D are reflectional symmetry, where $\mathbf{T}$ is a reflection over a given plane, and rotational symmetry, where $\mathbf{T}$ is a rotation around a given axis by a given angle. We further use the term circular symmetry for the case where the symmetry holds for any rotation angle around a given axis - such symmetry only occurs in rotational shapes.

While there are quite many existing methods for finding reflectional symmetries in 3D shapes, many of which are relatively robust in different ways, among others see, e.g., Podolak et al. (2006); Simari et al. (2006); Combès et al. (2008); Sipiran et al. (2014); Li et al. (2016); Schiebener et al. (2016); Speciale et al. (2016); Ecins et al. (2017); Nagar and Raman (2020); Hruda et al. (2021), the options for finding rotational symmetries appear to be significantly sparser and most of them have some major limitations.

Some robust methods for finding circular symmetries exist but they work only with rotational shapes and find only the axis, not the rotation angle. Ecins et al. (2018b) find the axis of symmetry in a partial 3D rotational shape by creating a plane defined by the axis and a given point of the object, computing the angle between this plane and the normal in the given point and minimizing the sum of these angles over all the points. Sipiran (2017) and Sipiran (2018) pursuit the same goal by finding circular structures in the input.

The other existing methods can also find the rotation angle and do not require a rotational shape on the input. However, they all seem to constrain the detected symmetries by having the axis pass through some reference point, such as the object's centroid. Such methods cannot find weaker symmetries or symmetries in partial objects because in such cases the axis does not generally pass through such a point. Sun and Sherrah (1997) use a discrete version of the Extended Gaussian Image Horn (1984) called orientation histogram and the principal axes of the input object. Martinet et al. (2006) use generalized moment functions and the observation that these moments have at least the same symmetries as the input shape. Korman et al. (2015) employ a distortion measure representing the amount of mismatched volume between the original shape and the transformed shape together with a clever sampling of the transformation group. Gothandaraman et al. (2020) create images of the input shape from different views defined by rotating the input object by specified angles around the x, y, z axis and they match these images using the SIFT feature Lowe (1999).

There are, of course, also methods that are capable of detecting more general symmetries, such as Lipman et al. (2010); Mavridis et al. (2015) or the well-known method of Mitra et al. (2006) and its newer modification Shi et al. (2016). However, modifying such methods to find symmetries defined by a rotation around an axis would probably be difficult or even impossible because such a transformation is rather specific and different from, e.g., a general rigid transform with which such methods usually can work. On top of this, these more general methods are usually also quite computationally expensive.

To our knowledge, no existing method for detecting rotational symmetries in 3D objects can find both the axis and angle of rotation and be robust, at least to some extent, to weak symmetries and missing parts at the same time. However, as mentioned above, there are quite a few approaches for finding reflectional symmetries that are robust in several ways. This implies that finding a way of using potential reflectional symmetries to find plausible rotational symmetries would be highly beneficial because many of the existing approaches for reflectional symmetries in 3D data could be extended to find the rotational ones. In this paper we propose a new method for finding rotational symmetry in 3D objects by using this very idea to extend an existing method for reflectional symmetry detection proposed recently by Hruda et al. (2021) which is based on maximizing a specific symmetry measure. The main contributions of our paper are following:

1. We propose a simple way of creating plausible candidates for rotational symmetries by combining suitable pairs of approximate reflectional symmetries.
2. We generalize the symmetry measure and the step of its maximization from Hruda et al. (2021) to work for rotational symmetries using a very simple, yet efficient, quaternion-based parameterization of the rotation transformation that seems novel in this area and stems from algebraically converting the quaternion rotation formula to a vector expression.
3. We combine these two contributions to extend the overall approach of Hruda et al. (2021) into a robust and efficient method for detecting rotational symmetry in a 3D point set.

Note that contributions 1 and 2 are both very well usable on their own and can be implemented in a different context as well. However, we believe that their combination provides an excellent basis for an overall very good solution to rotational symmetry detection.

The rest of our paper is organized as follows. Section 2 describes the core idea and the overall proposed rotational symmetry detection method in which this idea is integrated, including the modification of the symmetry measure using quaternion-based parameterization. Section 3 shows the results of the method and demonstrates its robustness and computational efficiency. Section 4 describes the limitations of the method and Section 5 concludes the paper.

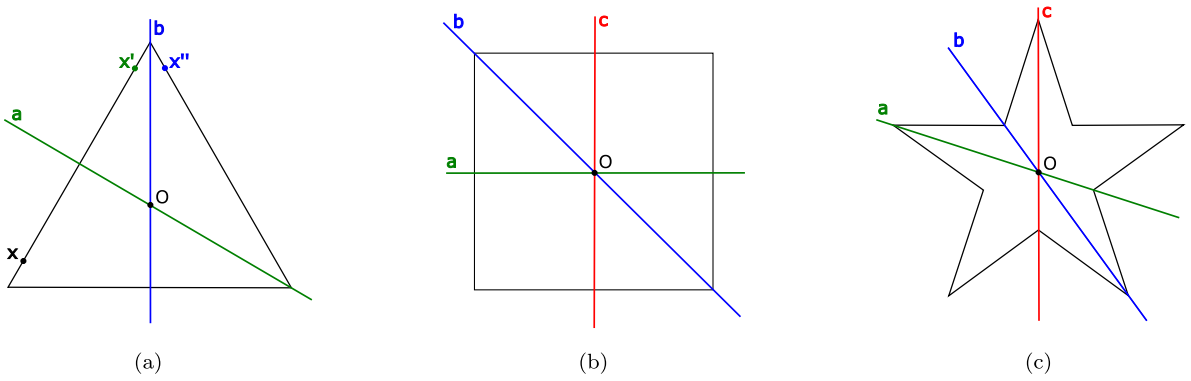


Fig. 1. 2D examples of reflectional symmetries where composing reflections over the lines results in rotational symmetries.

2. Rotational symmetry detection method

Let $\mathbf{r}(\mathbf{p}, \mathbf{x})$ be a transformation that reflects an arbitrary point $\mathbf{x}$ over a given plane $\mathbf{p}$. If we have two reflections $\mathbf{r}_1 = \mathbf{r}(\mathbf{p}_1, \mathbf{x})$, $\mathbf{r}_2 = \mathbf{r}(\mathbf{p}_2, \mathbf{x})$ with non-parallel planes $\mathbf{p}_1$, $\mathbf{p}_2$, composing $\mathbf{r}_1$ and $\mathbf{r}_2$ results in rotation around the axis, which is an intersection of the planes $\mathbf{p}_1$, $\mathbf{p}_2$, by the angle which is a double of the oriented angle between the planes Gallier (2011). In many cases a rotational symmetry in a 3D object follows this rule and is actually a composition of two reflectional symmetries. If an object has perfect reflectional symmetry w.r.t. both planes $\mathbf{p}_1$ and $\mathbf{p}_2$, the rotations $\mathbf{r}(\mathbf{p}_1, \mathbf{r}(\mathbf{p}_2, \mathbf{x}))$ and $\mathbf{r}(\mathbf{p}_2, \mathbf{r}(\mathbf{p}_1, \mathbf{x}))$ must also be its perfect symmetries - the object reflected over both $\mathbf{p}_1$ or $\mathbf{p}_2$ remains unchanged, i.e. it also remains unchanged when reflected over both these planes in succession (note, however, that the converse implication does not hold, i.e. not all rotationally symmetric objects have necessarily reflectional symmetries).

An object with a rotational symmetry often has multiple reflectional symmetries where the reflection planes intersect in the axis of the rotational symmetry and the rotation angle of the symmetry can be derived from the planes using the rule above. Let us demonstrate the idea on simple 2D cases (we replace reflection planes with lines and an axis of rotation with a center of rotation). Fig. 1 shows three objects with several marked lines of reflectional symmetries. In Fig. 1a, when the triangle is reflected over the symmetry line $\mathbf{a}$ (turning the point $\mathbf{x}$ into $\mathbf{x}'$) and then over $\mathbf{b}$ (turning $\mathbf{x}'$ into $\mathbf{x}''$) the resulting transformation is a rotation by 120° around the center O which lies at the intersection of the two lines that make an angle of 60°. Such a rotation is a symmetry of this object. Similarly, in Fig. 1b we get symmetry with 90° rotation around the center O for lines $\mathbf{a}$ and $\mathbf{b}$ or $\mathbf{b}$ and $\mathbf{c}$ and 180° for $\mathbf{a}$ and $\mathbf{c}$. In Fig. 1c we get 72° rotational symmetry around O with $\mathbf{a}$ and $\mathbf{b}$ or $\mathbf{b}$ and $\mathbf{c}$ and 144° with $\mathbf{a}$ and $\mathbf{c}$.

This idea is applicable in 3D as well. Therefore, if we have a reliable algorithm for finding planes of reflectional symmetries or plausible symmetry plane candidates, we can use them to generate meaningful candidates for rotational symmetries. It also holds when the symmetries are only approximate, i.e., no perfect reflectional symmetry exists. We use the method by Hruda et al. (2021) as the underlying symmetry plane detection algorithm and we propose its extension for finding rotational symmetries in 3D objects using the idea described above. We generate potential rotational symmetries from the reflectional ones and evaluate their fitness to select the best ones (see Section 2.3), in the end we perform refinement in the rotation space (see Section 2.4) using a proposed quaternion-based parameterization (see Section 2.2). Note, however, that this core idea could possibly be implemented with many other different symmetry plane detection approaches, as well as the proposed rotation refinement. Here is a brief overview of Hruda et al. (2021) which we chose because it is quite recent and seems to be fast and robust.

2.1. Underlying symmetry plane detection method

The method takes a set of points in E^3 as input and tries to find its best plane of reflectional symmetry. This is done by maximizing an objective function called symmetry measure. The symmetry measure $s_X(\mathbf{T})$ is for a point set $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}, \mathbf{x}_i \in E^3, i = 1, ..., n$ defined as

$$s_X(\mathbf{T}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \varphi(||\mathbf{T}(\mathbf{x}_i) - \mathbf{x}_j||) \tag{1}$$

where $\varphi(l) = \hat{\varphi}(\alpha l)$ is so called similarity function which transforms distance into similarity (the larger the distance l the lower the value of $\varphi(l)$) with α being its spread parameter and $\mathbf{T}$ is a geometric transformation which in Hruda et al. (2021) is $\mathbf{T}(\mathbf{x}) = \mathbf{r}(\mathbf{p}, \mathbf{x})$. The goal is to maximize s_X w.r.t. the parameters of the plane $\mathbf{p}$. The computation of s_X basically comes down to reflecting each point in X over $\mathbf{p}$, computing its similarity to all points of X and summing all these similarities. The more similar the reflected point set X is to the original X the larger the value of s_X should be. As $\hat{\varphi}(t)$ a specific Wendland's

function Wendland (1995) is used such that it closely resembles the Gaussian for $t \geq 0$ but is locally supported. It is a piecewise polynomial differentiable function and it is that $\hat{\varphi}(0) = 1$, $\frac{d}{dt}\hat{\varphi}(0) = 0$ and $\hat{\varphi}(t) = 0$ for $t > 2.6$. The final function φ is then just $\hat{\varphi}$ with its argument scaled by the spread parameter α which in Hruda et al. (2021) is set as $\alpha = \frac{15}{l_{avrg}}$ where l_{avrg} is the average distance of the points in the input point set from their centroid. See Hruda et al. (2021) for details and the exact definition of φ. With such a choice of φ the measure s_X is differentiable w.r.t. the parameters of the transformation **T** if **T** is parameterized differentiably. It can also be made higher-order differentiable if needed by choosing a higher-order differentiable Wendland's function.

The overall process of finding the best symmetry plane has several steps:

1. Two different point sets are created by simplifying the input set of points X using a fast grid-based simplification algorithm. One denoted X_{simp} with target point count 1000, and one denoted X_{cand} with target point count 100. The actual point counts are usually slightly larger than these values.
2. The point set X_{cand} is used to create candidates for symmetry planes. This is done by taking every pair of points in X_{cand} and creating the symmetry plane of that pair. If a new candidate is similar (distance smaller than δ) to some already created candidate it is not added into the candidate set but merged with the similar one. The number of resulting candidates is usually in the order of hundreds.
3. Each candidate is evaluated using the symmetry measure but instead of computing it for the original point set X it is computed for the simplified X_{simp} which has approximately 1000 points. Such point count is still sufficient to well represent the symmetry of the input object and the computation of the symmetry measure is usually significantly faster than for the original point set X. Since the similarity function φ is locally supported the points outside its support region can be excluded to further decrease the computational cost. This is done using a 3D grid as an auxiliary data structure. Now, S candidates with the largest symmetry measure $s_{X_{simp}}$ are selected for further processing. By default $S = 5$.
4. Local numerical optimization using the gradient-based L-BFGS method Liu and Nocedal (1989) is started from all the S candidates from the previous step to maximize their symmetry measure $s_{X_{simp}}$. This provides S local maxima of the measure among which the one with the largest measure value is selected as the resulting symmetry plane of the input point set X. The other local maxima can potentially be used as secondary symmetries if the input object has more than one significant reflectional symmetry.

After the optimization step, we could theoretically use the two best symmetries and if they were perfect, their combination would provide a perfect rotational symmetry. However, we aim for a robust method that also works for objects with weak symmetries, significant noise, or even for incomplete objects. In such cases testing only one rotation created by two reflectional symmetries does not have to be reliable enough. Furthermore, better results can be expected when optimizing the final rotation instead of optimizing the reflectional symmetries used to create it. This is why we omit step 4 completely.

Instead, we propose a method for detecting rotational symmetry of a set of points in E^3 built upon the first 3 steps of Hruda et al. (2021). We use the best reflectional symmetry candidates outputted by step 3 and pair them to create candidates for rotational symmetries. Then we use the symmetry measure from Eq. (1), modified to work with rotations, to evaluate the candidates and select a few best ones. As the final step, we perform the optimization directly in the space of rotations which is much more natural and, as will be shown in Section 2.4, can well compensate for the imprecision of the non-optimized reflectional symmetry candidates. Intuitively, this approach seems not to work for the detection of rotational symmetries in objects that are not reflectionally symmetric. However, we will show in Section 3 that it mostly works even in such cases. The details of the overall process of finding rotational symmetries will be provided in the rest of this section, as well as the description of any changes we made in the first three steps of Hruda et al. (2021).

2.2. Symmetry measure

In the following text we describe our modification of the symmetry measure to take rotation as the input transformation instead of reflection. We discovered that for finding rotational symmetries the symmetry measure needs to represent the symmetry of the input point set slightly more precisely than in the case of reflectional ones. This is probably because the rotation transformation has more degrees of freedom than a reflection. Therefore, we set the target point count of X_{simp} to 3000 instead of 1000 for the simplification algorithm and we use φ with slightly smaller radius by setting $\alpha = \frac{20}{l_{avrg}}$ instead of $\alpha = \frac{15}{l_{avrg}}$. Both these parameter values were set based on vast experiments to maximize the balance between the quality of the results and computation cost.

The symmetry measure allows finding its significant maxima by applying a rather sparse sampling of the candidate space and being able to converge to them from quite large distances using numerical optimization (see Hruda et al. (2021)). This saves a lot of time in the candidate evaluation step because the candidate count does not need to be very large. Since the symmetry measure is differentiable, and mainly because its gradient can be computed analytically, any gradient-based optimization technique can be employed to locate the maxima quickly (see, e.g., Wright et al. (1999) for various options or possibly the more recent Sterck (2013)). Such techniques are usually faster than those that do not require the gradient because they often need a much smaller number of iterations. Furthermore, the differentiability makes the measure smooth,

free of discontinuities and sudden changes in any direction which makes the optimization easier and faster in general, even with non-gradient methods. Therefore, when modifying the symmetry measure to work with rotations instead of reflections, it is useful to keep it differentiable with the possibility to compute its gradient analytically, retaining the option of using gradient-based methods and generally keeping the optimization time-efficient.

2.2.1. Rotation parameterization

As mentioned, the symmetry measure is differentiable if the transformation $\mathbf{T}$ is parameterized differentiably. In our case, $\mathbf{T}$ is a rotation around an axis. Rotations are usually represented by matrices but in our case this would be cumbersome due to the need to keep the rotation matrix orthogonal throughout the parameterization. Angle-based parameterizations, on the other hand, commonly suffer from singularities, such as poles, which might negatively influence the stability of the optimization. We would also like to keep the parameterization simple, easy to implement and possible to differentiate analytically. To fulfill these conditions, we propose using the following and rather simple quaternion-based representation instead.

Let $Q = a + bi + cj + dk$ be a quaternion where i, j, k are elements of the quaternion basis and $i^2 = j^2 = k^2 = -1, ij = k = -ji$, $jk = i = -kj$, $ki = j = -ik$ (see, e.g., Goldman (2010) for more details about quaternions). A quaternion conjugate of Q is defined as $Q^* = a - bi - cj - dk$ and the length of a quaternion is given as $|Q| = |Q^*| = \sqrt{Q\,Q^*} = \sqrt{a^2 + b^2 + c^2 + d^2}$. We denote $v(\mathbf{x}) = xi + yj + zk$ a quaternion that represents a vector or point $\mathbf{x} = [x, y, z]^T \in E^3$. Inversely, we denote $\mathbf{q}(u) = [x, y, z]^T \in E^3$ a vector representation of a quaternion $u = xi + yj + zk$.

Any quaternion can be expressed as $Q = a + v([b, c, d]^T)$. Subsequently, any unit quaternion can be expressed as $Q = \cos(\beta) + \sin(\beta)v(\mathbf{u})$ where $\mathbf{u}$ is a unit vector. Then the expression $Q\,v(\mathbf{x})\,Q^*$ represents the point $\mathbf{x}$ rotated by the angle 2β around the axis that passes through the origin in the direction of the vector $\mathbf{u}$. By denoting $a = \cos(\beta)$, $\mathbf{v} = [b, c, d]^T = \sin(\beta)\mathbf{u}$, $Q = a + v(\mathbf{v})$ we can expand this as

$$Q\,v(\mathbf{x})\,Q^* = (a + v(\mathbf{v}))v(\mathbf{x})(a - v(\mathbf{v})) = a^2 v(\mathbf{x}) - a v(\mathbf{x})v(\mathbf{v}) + a v(\mathbf{v})v(\mathbf{x}) - v(\mathbf{v})v(\mathbf{x})v(\mathbf{v}).$$

Now using the rule that the product of two vectors $\mathbf{a}$, $\mathbf{b}$ in quaternion algebra is $v(\mathbf{a})v(\mathbf{b}) = -\mathbf{a}\cdot\mathbf{b} + v(\mathbf{a}\times\mathbf{b})$ (see Goldman (2010)) we can further expand the expression as

$$a^2 v(\mathbf{x}) - a(-\mathbf{x}\cdot\mathbf{v} + v(\mathbf{x}\times\mathbf{v})) + a(-\mathbf{v}\cdot\mathbf{x} + v(\mathbf{v}\times\mathbf{x})) - (-\mathbf{v}\cdot\mathbf{x} + v(\mathbf{v}\times\mathbf{x}))v(\mathbf{v}).$$

Because the cross product is anticommutative we can replace $v(\mathbf{v}\times\mathbf{x})$ with $-v(\mathbf{x}\times\mathbf{v})$ and since the terms $a(\mathbf{x}\cdot\mathbf{v})$ and $-a(\mathbf{v}\cdot\mathbf{x})$ cancel out we can continue with the expansion as follows

$$a^2 v(\mathbf{x}) - 2a v(\mathbf{x}\times\mathbf{v}) + (\mathbf{v}\cdot\mathbf{x})v(\mathbf{v}) + v(\mathbf{x}\times\mathbf{v})v(\mathbf{v}).$$

The last term can be further expanded into

$$v(\mathbf{x}\times\mathbf{v})v(\mathbf{v}) = -(\mathbf{x}\times\mathbf{v})\cdot\mathbf{v} + v((\mathbf{x}\times\mathbf{v})\times\mathbf{v})$$

and because the vectors $\mathbf{x}\times\mathbf{v}$ and $\mathbf{v}$ are orthogonal the first term $-(\mathbf{x}\times\mathbf{v})\cdot\mathbf{v}$ is 0. Using the well known vector triple product rule, i.e. having three vectors $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ it is $(\mathbf{a}\times\mathbf{b})\times\mathbf{c} = (\mathbf{a}\cdot\mathbf{c})\mathbf{b} - (\mathbf{b}\cdot\mathbf{c})\mathbf{a}$, we get

$$v(\mathbf{x}\times\mathbf{v})v(\mathbf{v}) = v((\mathbf{x}\cdot\mathbf{v})\mathbf{v} - (\mathbf{v}\cdot\mathbf{v})\mathbf{x}) = (\mathbf{x}\cdot\mathbf{v})v(\mathbf{v}) - (\mathbf{v}\cdot\mathbf{v})v(\mathbf{x}).$$

By plugging this back into the rotation formula we get

$$Q\,v(\mathbf{x})\,Q^* = a^2 v(\mathbf{x}) - 2a v(\mathbf{x}\times\mathbf{v}) + 2(\mathbf{x}\cdot\mathbf{v})v(\mathbf{v}) - (\mathbf{v}\cdot\mathbf{v})v(\mathbf{x})$$

and to use the transformation in the symmetry measure it needs to be expressed in the vector space which leads to

$$\mathbf{q}(Q\,v(\mathbf{x})\,Q^*) = a^2\mathbf{x} - 2a(\mathbf{x}\times\mathbf{v}) + 2(\mathbf{x}\cdot\mathbf{v})\mathbf{v} - (\mathbf{v}\cdot\mathbf{v})\mathbf{x}. \tag{2}$$

The vector expression can be easily differentiated and is useful also for implementation purposes because vector operations are easier to implement then operations in quaternion algebra.

Since in general the rotation axis does not pass through the origin we can select an arbitrary point $\mathbf{s} = [s_x, s_y, s_z]^T$ that lies on the axis and express the rotation as $\mathbf{q}(Q\,v(\mathbf{x} - \mathbf{s})\,Q^*) + \mathbf{s}$. The rotation transformation can be parameterized using the coefficients of the quaternion Q, i.e. a, b, c, d, and the coordinates of the point on the axis $\mathbf{s}$, i.e. s_x, s_y, s_z. A general quaternion $Q = a + bi + cj + dk$ is not unit and since only unit quaternions represent rotations we need to incorporate normalization into the parameterization. We therefore define the rotation transformation as

$$\begin{aligned}\mathbf{T}(\mathbf{x}) = \mathbf{rot}(Q, \mathbf{s}, \mathbf{x}) &= \mathbf{q}\left(\frac{1}{|Q|}Q\,v(\mathbf{x} - \mathbf{s})\frac{1}{|Q|}Q^*\right) + \mathbf{s} \\ &= \mathbf{q}\left(\frac{1}{|Q|^2}Q\,v(\mathbf{x} - \mathbf{s})\,Q^*\right) + \mathbf{s} = \frac{1}{a^2 + b^2 + c^2 + d^2}\mathbf{q}\left(Q\,v(\mathbf{x} - \mathbf{s})\,Q^*\right) + \mathbf{s}.\end{aligned}$$

By again denoting $\mathbf{v} = [b, c, d]^T$, $Q = a + v(\mathbf{v})$ and using the vector expression from Eq. (2) we can finally rewrite the transformation as

$$\begin{aligned}\mathbf{rot}(Q, \mathbf{s}, \mathbf{x}) &= \frac{1}{a^2 + \mathbf{v}\cdot\mathbf{v}}\mathbf{q}\left(Q\, v(\mathbf{x}-\mathbf{s})\, Q^*\right) + \mathbf{s} \\ &= \frac{1}{a^2 + \mathbf{v}\cdot\mathbf{v}}\left(a^2(\mathbf{x}-\mathbf{s}) - 2a((\mathbf{x}-\mathbf{s})\times\mathbf{v}) + 2((\mathbf{x}-\mathbf{s})\cdot\mathbf{v})\mathbf{v} - (\mathbf{v}\cdot\mathbf{v})(\mathbf{x}-\mathbf{s})\right) + \mathbf{s}.\end{aligned}$$

This easy-to-implement formula is obviously differentiable w.r.t. all seven parameters a, b, c, d, s_x, s_y, s_z (except for $a = b = c = d = 0$ which is not a rotation) and it only uses basic algebraic operations.

As a result, by substituting $\mathbf{T}(\mathbf{x}) = \mathbf{rot}(Q, \mathbf{s}, \mathbf{x})$ we get a symmetry measure in the form

$$s_X(\mathbf{rot}) = s_X(Q, \mathbf{s}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\varphi(||\mathbf{rot}(Q, \mathbf{s}, \mathbf{x}_i) - \mathbf{x}_j||) \tag{3}$$

with the following useful properties all of which are only possible due to the simple, yet efficient, quaternion-based parameterization of the rotation transformation:

1. It is still differentiable w.r.t. all parameters of the transformation and its gradient can be analytically computed.
2. Apart from the square root function needed to compute Euclidean distances it only uses basic algebraic operations (φ is piece-wise polynomial) - no need for trigonometric functions and conversions between angles and their sines/cosines.
3. It lacks major singularities, such as poles that appear in the case of angle-based representations of the rotation, allowing for more stable computation and optimization.

2.2.2. Small angle penalization

The use of the symmetry measure from Eq. (3) often tends to rotations with a random axis and the rotation angle as small as possible, often near 0. The method obviously finds the rotation as close as possible to the identity which is not a symmetry but would provide the best possible match. We discovered that this phenomenon is wholly suppressed if rotations with angles with an absolute value below 30° are forbidden.

While this is easy to ensure when creating the candidates (see later in Section 2.3) modifying the symmetry measure to enforce this in the optimization step (see later in Section 2.4) is slightly more difficult since we still want to keep it differentiable. Therefore, we use the symmetry measure in the following form:

$$s_X(Q, \mathbf{s}) = \left(\sum_{i=1}^{n}\sum_{j=1}^{n}\varphi(||\mathbf{rot}(Q, \mathbf{s}, \mathbf{x}_i) - \mathbf{x}_j||)\right) Pen(Q) \tag{4}$$

where $Pen(Q)$ is a function that penalizes angles with small absolute values. Let γ be the rotation angle, we need $Pen(Q) = 0$ for $|\gamma| < 30°$ but we need it to start decreasing at some larger angle, we choose 43° for this threshold, i.e. $Pen(Q) = 1$ for $|\gamma| > 43°$ and for $|\gamma| \in \langle 30°, 43°\rangle$ it decreases with $|\gamma|$. The 43° value was chosen so that the method is still capable of detecting symmetries with rotation angle around 45° and the 2° difference is to provide some safe margin. We could define Pen for the angle γ but to avoid using trigonometric functions we instead define Pen around the cosine of $\beta = \frac{1}{2}\gamma$ which can be easily obtained from the quaternion Q as $\cos(\beta) = \frac{a}{\sqrt{a^2+\mathbf{v}\cdot\mathbf{v}}}$. We set $t_1 = \cos(21.5°)$, $t_2 = \cos(15°)$. The rotation is by 2β for $\cos(\beta)$ so for $-\cos(\beta)$ it is by $360° - 2\beta$, i.e. by 2β in the opposite direction. Therefore, Pen needs to satisfy $Pen(Q) = 1$ for $|\cos(\beta)| < t_1$, $Pen(Q) = 0$ for $|\cos(\beta)| > t_2$ and be decreasing with increasing $|\cos(\beta)|$ for $|\cos(\beta)| \in \langle t_1, t_2\rangle$. Our similarity function $\hat{\varphi}(t)$ is differentiable and behaves in the needed way for $t \in \langle 0, 2.6\rangle$ so we just need to transform $\langle t_1, t_2\rangle$ to this interval. So we define

$$\hat{Pen}(t) = \begin{cases} 1 & t \in \langle -t_1, t_1\rangle \\ \hat{\varphi}\left((|t| - t_1)\frac{2.6}{t_2 - t_1}\right) & otherwise \end{cases}$$

which is differentiable for any t because $\hat{\varphi}(t)$ is differentiable, $\frac{d}{dt}\hat{\varphi}(0) = 0$ so the transition through t_1 and $-t_1$ to the constant 1 is differentiable as well and the constant between $-t_1$ and t_1 is differentiable trivially (see Fig. 2). Now, we simply define $Pen(Q) = \hat{Pen}(\frac{a}{\sqrt{a^2+\mathbf{v}\cdot\mathbf{v}}})$ which is differentiable w.r.t. all four parameters of Q (again except for $a = b = c = d = 0$) and therefore the symmetry measure in Eq. (4) is differentiable as well. Fig. 3 shows how the Pen function changes with the rotation angle γ.

Of course, this basically prevents the method from detecting symmetries with rotation angles below 43°. However, for any rotational symmetry with the rotation angle $|\gamma| < 180°$ there should be also significant symmetries with the same rotation axis and rotation angles with different k-multiples of γ, some larger than 43°. Therefore, in the case of symmetry with $|\gamma| < 43°$ the method should find a corresponding and similarly good symmetry with some rotation angle $k\gamma$, $k \in \mathbb{Z}$,

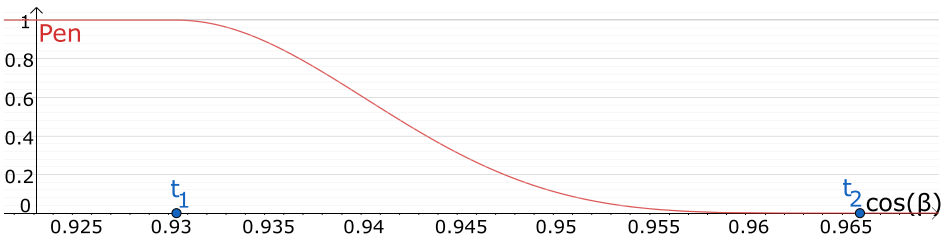


Fig. 2. The penalization function for $\cos(\beta)$ transiting through t_1 and t_2, the function is even and continues to be 1 to the left (up until $\cos(\beta) = -t_1$) and 0 to the right.

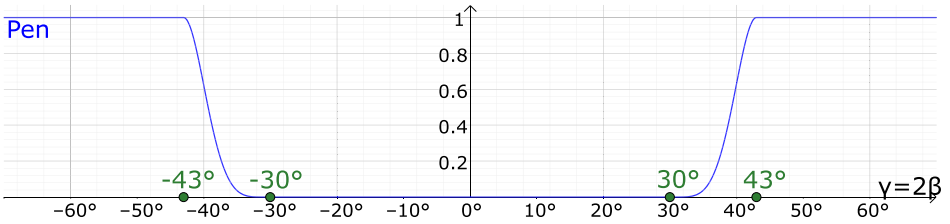


Fig. 3. The penalization function w.r.t. the rotation angle $\gamma = 2\beta$, the function continues to be 1 to both sides up until $\gamma = 180° / -180°$.

$|k\gamma| \geq 43°$, so we do not actually lose the symmetry. The corresponding lower angle symmetry could possibly be detected additionally by dividing the symmetry angle by different integers k and testing the resulting rotations but we leave this for future work.

2.3. Candidate creation and pruning

The first stage of our method consists of creating candidates for rotational symmetries. At the beginning we simply run steps 1, 2 and 3 of Hruda et al. (2021) (see Section 2.1) only with X_{simp} and α changed as described at the beginning of Section 2.2. Since we will use pairs of planes outputted by step 3 to create the rotation candidates and we need a decent amount of them, the default value of $S = 5$ is too small, so we increase it to $S = 30$ which proved to be more than enough in most cases. We therefore get a set of candidate planes $\{\mathbf{p}_i\}, i = 1, ..., 30$. For each pair of planes $\mathbf{p}_i$, $\mathbf{p}_j$, $i \neq j$ we compute the line of their intersection and an angle $\gamma = 2\arccos(\mathbf{n}_i \cdot \mathbf{n}_j)$ where $\mathbf{n}_i$, $\mathbf{n}_j$ are unit normal vectors of planes $\mathbf{p}_i$, $\mathbf{p}_j$ respectively. Now if $|\gamma| > 30°$ we create two candidate rotations around the intersection line by angles γ and $-\gamma$. The point $\mathbf{s}$ on the rotation axis is chosen so that it is the closest point on the axis to the centroid of the input object. From now on, we consider the rotation angle to be always positive where the rotations in the opposite direction are defined using the opposite axis direction.

Similar to Hruda et al. (2021), we incorporate pruning into the candidate creation process to reduce the final candidate count, i.e., to save time in the candidate evaluation step, and to ensure that there are no too similar candidates which would result in the optimization (see the next section) being started multiple times from almost the same place. Checking the similarity of two rotation transformations could be done using the theoretically appropriate distance function based on vertex sum of squares Hruda et al. (2019); Pottmann et al. (2006). However, it proved to be unnecessarily complex and computationally expensive mainly for the need to transform the candidates from quaternion-based to matrix-based representation. We discovered that the following much simpler approach provides almost identical results. When a new rotation candidate $\mathbf{c} = (Q, \mathbf{s})$ is created we set its weight $w(\mathbf{c}) = 1$ and find whether any previously created candidate $\mathbf{c}_i = (Q_i, \mathbf{s}_i)$ exists such that

$$|Q - Q_i \operatorname{sgn}(Q \cdot Q_i)| < \epsilon_Q,$$

$$||\mathbf{s} - \mathbf{s}_i|| < \epsilon_{\mathbf{s}}$$

where $Q \cdot Q_i$ represents a quaternion dot product which corresponds to dotting the four-dimensional vectors with the same coordinates as the quaternion coefficients. If so, we merge the candidates into a new one $\mathbf{c}_{new} = (Q_{new}, \mathbf{s}_{new})$, $w(\mathbf{c}_{new}) = w(\mathbf{c}) + w(\mathbf{c}_i)$ where $\mathbf{s}_{new}$ is a weighted average of $\mathbf{s}$ and $\mathbf{s}_i$ with $\frac{w(\mathbf{c})}{w(\mathbf{c})+w(\mathbf{c}_i)}$ and $\frac{w(\mathbf{c}_i)}{w(\mathbf{c})+w(\mathbf{c}_i)}$ as the corresponding weights. Similarly, Q_{new} is computed using the standard SLERP quaternion interpolation Goldman (2010) between Q and Q_i with the same weights. We set the thresholds experimentally as $\epsilon_Q = 0.05$, $\epsilon_{\mathbf{s}} = 0.05 l_{avrg}$, but we note that the method is not very sensitive to the exact value of either of them. We recenter $\mathbf{s}_{new}$ such that it is again the closest point on the axis of $\mathbf{c}_{new}$ to the centroid and the same process is then performed for $\mathbf{c}_{new}$ to see if it should be merged with some other candidate.

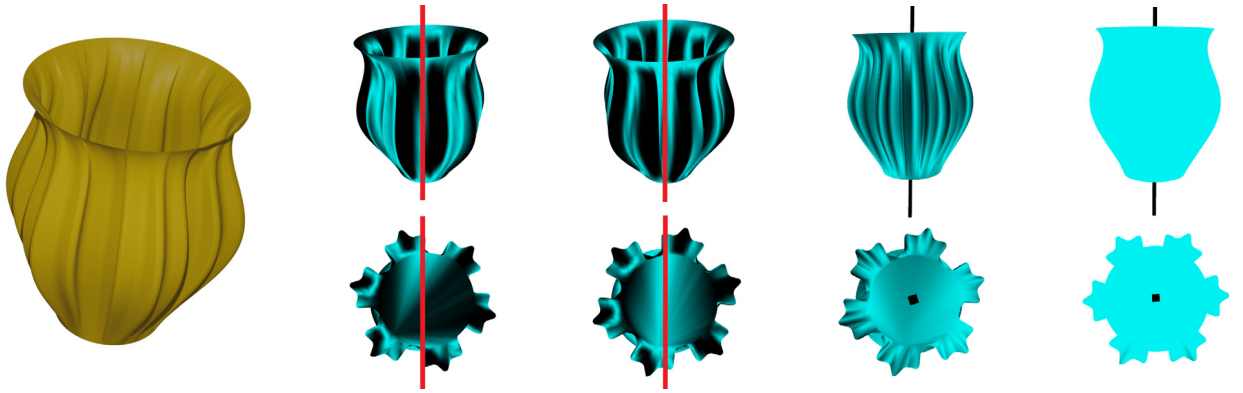


Fig. 4. The process of obtaining the rotational symmetry from the reflectional symmetry candidates, (a) shows the original object, (b) and (c) depict the candidate symmetry planes with the object colored according to the strength of the symmetry (the lighter the color, the stronger the symmetry in the given point), (d) shows the initial rotational symmetry candidate created by the two planes, the object is now colored according to the rotational symmetry, (e) shows the final rotational symmetry acquired by the optimization started from the initial one, the rotation angles for (d) and (e) are in the captions. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

The number of resulting candidates is usually in the lower or mid-hundreds. We compute the symmetry measure from Eq. (4) with X_{simp} for each one of the candidates. The calculation of $s_{X_{simp}}$ is accelerated in the same way as in Hruda et al. (2021) by exploiting the locality of φ and using a 3D grid to find the points that could be in its support region. Finally, we select M candidates with the largest measure where $M = 5$ by default.

2.4. Optimization

As the final step, local optimization using the L-BFGS Liu and Nocedal (1989) method is started from the M candidates outputted by the previous step to find M local maxima of the symmetry measure $s_{X_{simp}}$ from Eq. (4). Computation of the gradient of $s_{X_{simp}}$ is accelerated using the 3D grid in the same way as the computation of $s_{X_{simp}}$ itself. The one local maximum with the largest measure is then selected and can be declared the resulting rotational symmetry of the input point set X and considered the output of our method. The other local maxima can potentially be used as secondary symmetries.

Fig. 4 demonstrates the overall process of acquiring the final rotational symmetry from the initial symmetry plane candidates. Fig. 4a shows the original object; note that it is strongly (near perfectly) but not perfectly symmetrical. For visualization simplicity, the object is represented by a triangle mesh but we used only its vertices as the input point set. Figs. 4b and 4c show the two initial symmetry plane candidates respectively - the object is shown from two different viewpoints (side view, bottom-top view) in such a way that the candidate plane (marked by the red line) is perpendicular to the plane of the figure. The object is colored so that the points that are symmetric w.r.t. the plane are light (the lighter the color, the stronger the symmetry) while the others are dark. Notice that although there is non-negligible reflectional symmetry w.r.t. both these planes, neither of them is very precise. Fig. 4d shows the initial candidate rotation created by the two planes. The coloring now shows symmetry w.r.t. the rotation transformation, the rotation axis is shown in the figure and the angle is in the caption. Obviously, the symmetry is not very good - the axis is tilted and the angle is more than 3° off (60° is expected for this object). Finally, Fig. 4e depicts the resulting rotational symmetry outputted by the numerical optimization started from the initial one. We note that the candidate symmetry planes are often more precise than those in Figs. 4b, 4c and that this scenario is not entirely common. However, it demonstrates how robustly the symmetry measure, combined with the optimization approach, extends the core idea of combining candidates for reflectional symmetries into a solution for detecting rotational symmetries. From two rather weak reflectional symmetries we acquired an imprecise rotational symmetry candidate, which was then turned into a very strong symmetry by optimizing (maximizing) the symmetry measure in the space of rotations, which can be done quite easily using the proposed quaternion-based rotation parameterization. This robustness is very useful if the input object has rotational but not reflectional symmetries, as detailed later in Section 3.

The method attempts to find the best possible rotational symmetry. Suppose the input object has multiple similarly significant symmetries with the same axis and different k-multiples of (almost) the same angle. In that case, the method chooses mostly randomly which one is the best. For example, if there is a strong symmetry with rotation angle 60°, the resulting symmetry would have the corresponding axis and the rotation angle will basically be a random choice between 60°, 120° and 180°. The method simply chooses the k-multiple for which there is the best match according to the symmetry measure, even if the difference is marginal compared to the other k-multiples. However, from the user perspective, the smallest possible k-multiple might be preferred in these scenarios. We observe that such rotation is very often present among the other four detected local maxima of the symmetry measure. Therefore, our method has one more optional step which we execute by default. We simply check all the four other detected secondary rotations and if any of them has almost

the same symmetry measure as the best detected rotation (we allow at most 1% difference) but has a smaller rotation angle, we chose it as the output instead. If multiple rotations satisfy this condition, we choose the one with the smallest rotation angle. Even with this step the method cannot find rotations with angles below 43° (see Section 2.2.2). So if the smallest k-multiple of the angle of the best symmetry is, e.g., 30° the method will most likely output its 60° equivalent.

In the case of perfect symmetry, the rotation angle is always $\frac{360°}{k}$, $k \in \mathbb{Z}$ and, therefore, the resulting angle of the detected rotation could always be rounded to the nearest such value. However, this is not the case with approximate/weak symmetries. Since we do not seek the rotation axis and angle separately but rather search for a compound rotation, changing the angle alone would also require finding a new axis that provides optimal symmetry with the new angle. Even then, the new rotation with the rounded angle generally does not have to provide a better symmetric match than the original rotation for non-perfectly symmetrical objects. This is why we leave the resulting angle as outputted by the optimization as part of the resulting rotation but the rounding step can be very easily implemented if preferred by the user.

2.5. Identifying circular symmetry

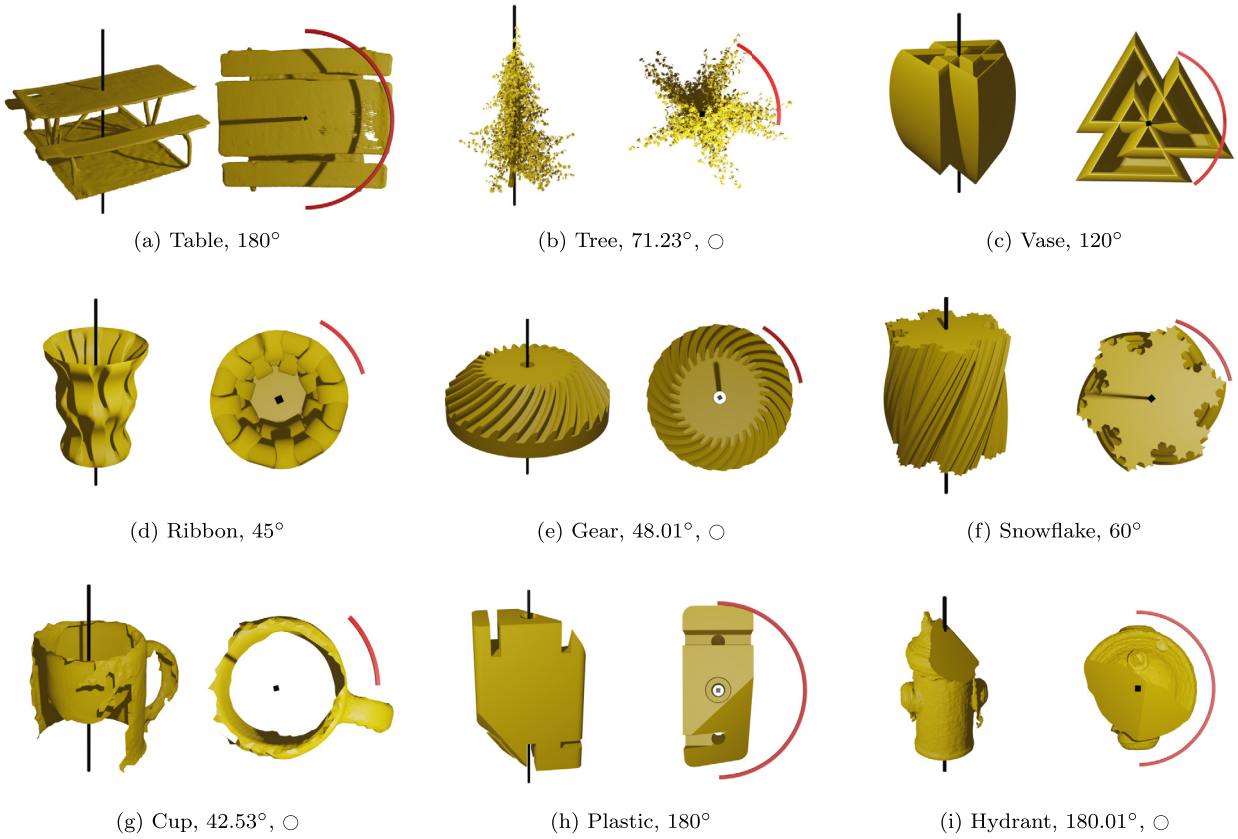
Our method cannot straightforwardly detect circular symmetries because they cannot be represented by specific geometric transformations and the symmetry measure cannot be applied for them. Therefore, if the input object possesses circular symmetry, our method will identify its axis and the rotation angle will be mostly arbitrary. However, we perform a simple check to identify whether the outputted symmetry could be contained in a potential circular symmetry. We take the axis of the outputted rotation and generate rotations with different angles γ around this very axis. We use all prime numbers between 43° and 180° as the values of γ so that there are no reoccurring multiples of some smaller angles. We compute the symmetry measure for all these rotations, and if their average is at least $\frac{2}{3}$ of the symmetry measure of the outputted rotational symmetry, we report potentially significant circular symmetry around the axis of the outputted rotation. The $\frac{2}{3}$ threshold might seem very low but note that we aim our method also for objects with missing parts and in such cases, the average measure would be noticeably lowered because large areas can be misaligned for some angles.

3. Results

The proposed method was implemented in C#, all the experiments were performed on a computer with CPU Intel® Core i7-10700F (frequency: 2.9 GHz, turbo boost: 4.8 GHz, 8 cores, L1 cache: 512 kB, L2 cache: 2 MB, L3 cache: 16 MB) and 32 GB memory (frequency: 3.2 GHz) running a Windows 10 operating system. Fig. 5 shows the results of our method on several objects from various different datasets Zhou and Jacobson (2016); Funk et al. (2017); Ecins et al. (2018a); Kohek et al. (2019). More results can be found in the supplementary material SupplementDownload (2022). For each object, there is a side view with the detected axis and a top-down view with the axis in the center and the red arc depicting the rotation angle of the detected symmetry. The angle is also in the caption, sometimes followed by a circle which means that the method indicated potential circular symmetry for the given object. All the objects are represented by triangle meshes because this is the most common object representation in computer graphics and it eases visualization. However, we only take the vertices as the input point set for our method. Furthermore, some of these objects are 3D scans of real objects and were probably created by triangulating raw point clouds (for the Cup we are certain of this) so using their vertices is the same as running the method on the original point cloud. Since the proposed method is not entirely rotationally invariant (mainly due to the grid-based simplification) all the objects were randomly rotated before starting the symmetry detection.

There are several objects with strong symmetries, some with weaker symmetries and even some incomplete objects (the bottom row). The Cup was incomplete originally, for Plastic and Hydrant the missing parts were removed artificially by clipping. For the Cup and Gear the detected angle is rather arbitrary because there is a quite strong circular symmetry in both cases so other angles would be plausible as long as the axis is the same. The similar applies for the Hydrant where the approximate 180° angle corresponds to the strongest symmetry but different angles with the given axis would still provide pleasing approximate symmetries. In fact, for different random rotations of the Hydrant our method sometimes detects the symmetry with a different arbitrary angle, still providing a plausible symmetry just not the best one possible. This can, however, be easily mitigated by increasing the M parameter, determining the number of best candidates selected for the optimization, from the default 5 to approximately 10 or more. For the Tree object the angle of the detected symmetry is 71.23°, which seems correct since an angle around 72° can be expected, and circular symmetry is indicated. This is due to its weak symmetry and overall conic shape which might not be obvious from the top-down view but is quite clear from the side view.

Notice that some of the objects do not have reflectional symmetries (Vase, Snowflake and somewhat even Gear) but the proposed method still finds a very good rotational symmetry for all of them. This is because the symmetry plane candidates that approximately intersect in the axis of rotational symmetry still have a noticeably larger symmetry measure than other arbitrary planes and they are more than likely to be among the best planes passed to the rotation creation step. Let us demonstrate this on a very simple shape depicted in the top row of Fig. 6. Fig. 6a shows the detected rotational symmetry axis for this object, the detected angle is 180°. Figs. 6b and 6c show the two planes that were used to create the initial candidate whose optimization lead to the final symmetry, whereas Fig. 6d shows a random plane passing through the object, the symmetry coloring is the same as in Fig. 4. Unlike the random plane, the first two planes capture symmetry in a rather large portion of the object's points, although its whole shape actually does not have a reflectional symmetry, which explains



(a) Table, 180° (b) Tree, 71.23°, ○ (c) Vase, 120°

(d) Ribbon, 45° (e) Gear, 48.01°, ○ (f) Snowflake, 60°

(g) Cup, 42.53°, ○ (h) Plastic, 180° (i) Hydrant, 180.01°, ○

Fig. 5. Results of the proposed method on several objects with both strong and weak symmetries and with missing parts, each figure shows a side view with the detected axis, a top-down view with the axis in the center and a red arc depicting the detected rotation angle which is also in the caption, the circles indicate detected possible circular symmetry.

Table 1
Computation times of the proposed method, first row shows the point counts of the objects, second row shows the computation times of the single-thread and the last row of the multi-thread implementation, times include simplification.

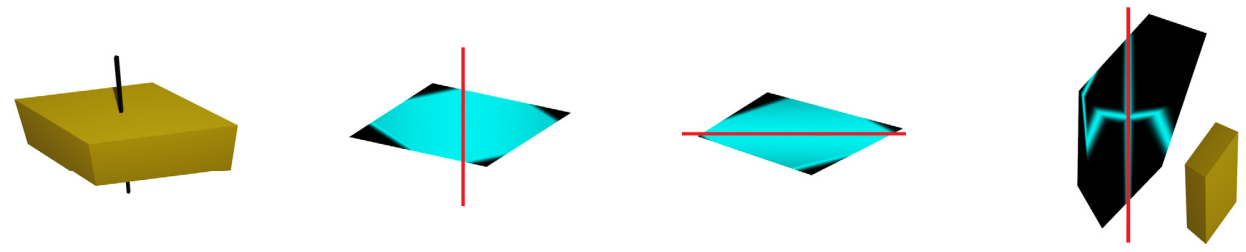
Object	Plastic	Vase	Snowflake	Cup	Gear	Ribbon	Hydrant	Tree	Table
Point count	617	3115	3264	4104	14566	16160	72012	134194	151026
Time s.t. [ms]	492	2459	2109	2074	3075	1814	2467	1380	2227
Time m.t. [ms]	209	1117	490	542	858	494	644	467	683

why they end up among the best symmetry plane candidates. Of course, the rotational symmetry created by such planes in similar cases cannot be expected to be very precise but this is solved by the robustness of the optimization step (see Section 2.4).

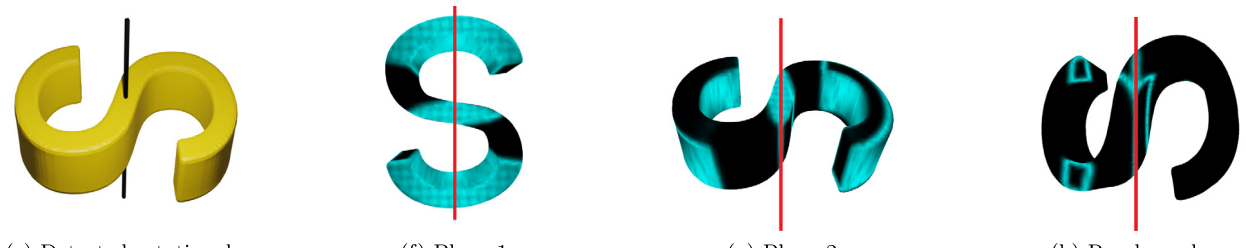
The bottom row of Fig. 6 provides the same demonstration for an S-shaped object with even less reflectional symmetry showing that the proposed method can truly find even rotational symmetries which are not created by significant reflectional symmetries. However, it should be noted that the probability of failure is naturally larger in such cases because the chance of *hitting* the right rotation candidate is more driven by random influence than by the reflectional symmetry of the object. Increasing the values of some parameters such as S (number of selected reflections) and M (rotations) can generally improve the chance of success but for the cost of longer computation time.

Table 1 demonstrates the computational efficiency of the proposed method. As some parts of the method can be easily parallelized, we show the computation time of both single-thread (s.t.) and multi-thread (m.t.) implementation. The running time obviously does not depend much on the object's point count, which is ensured by the fast simplification that is also included in the measurements. The single-thread implementation runs mostly under 3 seconds but since nowadays multi-core CPUs are standard, we are confident that our method can mostly run under or around 1 second, as suggested by the last row.

For some objects, other significant symmetries can be found by looking at the remaining local maxima of the symmetry measure outputted by the optimization step. Fig. 7 shows a dodecahedron-like object with the first three detected rotational symmetries. The fourth symmetry has a similar angle to the second but different axis (however, the images would be almost

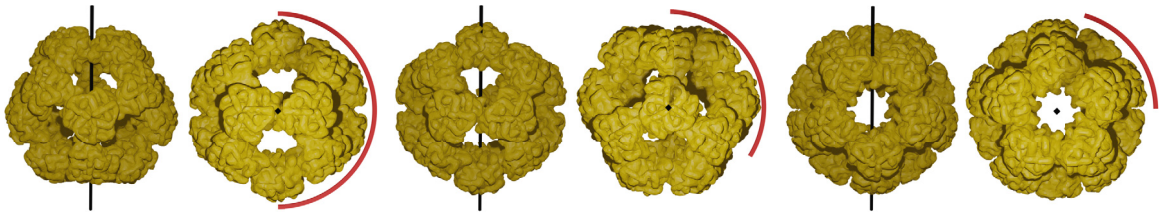


(a) Detected rotational symmetry, 180° (b) Plane 1 (c) Plane 2 (d) Random plane



(e) Detected rotational symmetry, 180° (f) Plane 1 (g) Plane 2 (h) Random plane

Fig. 6. Two different shapes without reflectional symmetries, the first figure in each row shows the detected rotational symmetry, the second and third show the two symmetry plane candidates, which created the initial rotation, with symmetry coloring (the objects are oriented so that the planes are orthogonal to the figure plane) and the last figure depicts a random plane with symmetry coloring (the small yellow image in the top row provides perspective).



(a) First symmetry, 180° (b) Second symmetry, 120.05° (c) Third symmetry, 72°

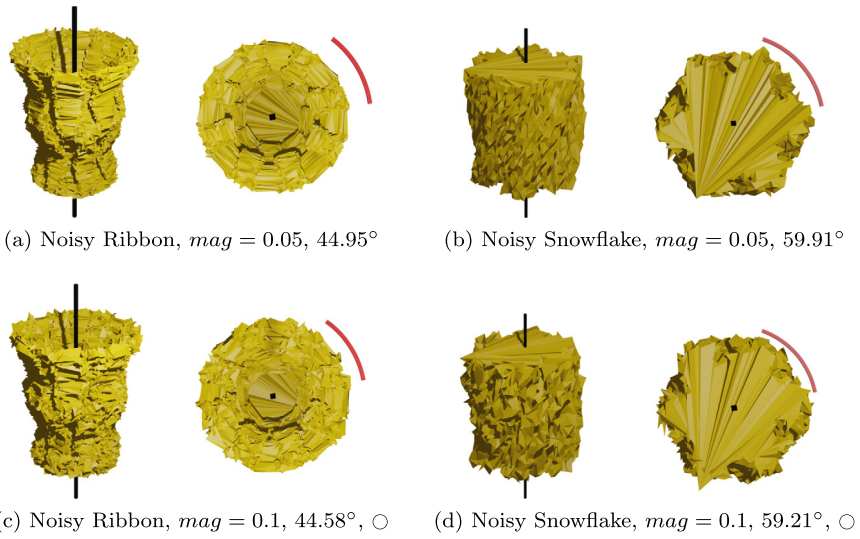
Fig. 7. The best three detected rotational symmetries of a dodecahedron-like objects.

identical) and the fifth is similar to the third. By increasing M and optimizing more candidates we could possibly obtain even more different symmetries.

To demonstrate the robustness of our method to noise we added the vector $[rand_x, rand_y, rand_z]^T \cdot l_{avrg} \cdot mag$ to every point of the input object where $rand_x, rand_y, rand_z$ are uniform random values from $\langle -1, 1 \rangle$, l_{avrg} is the average distance of points of the object from their centroid and mag determines the noise magnitude. Fig. 8 depicts the detected symmetries for the Ribbon and Snowflake objects with $mag = 0.05$ (top) and $mag = 0.1$ (bottom). For $mag = 0.1$ the method indicates circular symmetries because the noise noticeably decreases the precision of the present symmetry so the differences between symmetries with different angles are much smaller, presuming the axis is correct. Still, the detected symmetries are very good for all these noisy objects. The chart in Fig. 9 shows how the noise magnitude influences the precision of the symmetry detection. We used four different objects (Snowfake, Ribbon, Vase and Gear) and we normalized them in size such that $l_{avrg} = 1$. For each object we added noise of varying magnitude and applied our method to find rotational symmetry. Then we used the resulting rotation to transform the original noise-free object and measured its distance from its non-transformed version using the Metro mean error Cignoni et al. (1998) similarly as done, e.g., in Li et al. (2016). Due to the normalization, the units of both the error and noise magnitude are percents of l_{avrg}. The error starts growing noticeably only after the noise magnitude reaches roughly 5%, which corresponds to the top of Fig. 8, and a significant increase in error starts only after roughly 10%, which corresponds to the bottom of Fig. 8. Overall, the error seems to grow rather slowly with the noise magnitude, confirming the robustness of the proposed method to noise.

3.1. Comparisons

As already mentioned, not many methods for rotational symmetry detection exist. Moreover, their implementation is usually not publically available, at least not to our knowledge, and no benchmark datasets with ground truth rotational



(a) Noisy Ribbon, $mag = 0.05$, 44.95° (b) Noisy Snowflake, $mag = 0.05$, 59.91°

(c) Noisy Ribbon, $mag = 0.1$, 44.58°, ○ (d) Noisy Snowflake, $mag = 0.1$, 59.21°, ○

Fig. 8. Ribbon and Snowflake with added uniform noise, in (a,b) with $mag = 0.05$ and in (c,d) with $mag = 0.1$.

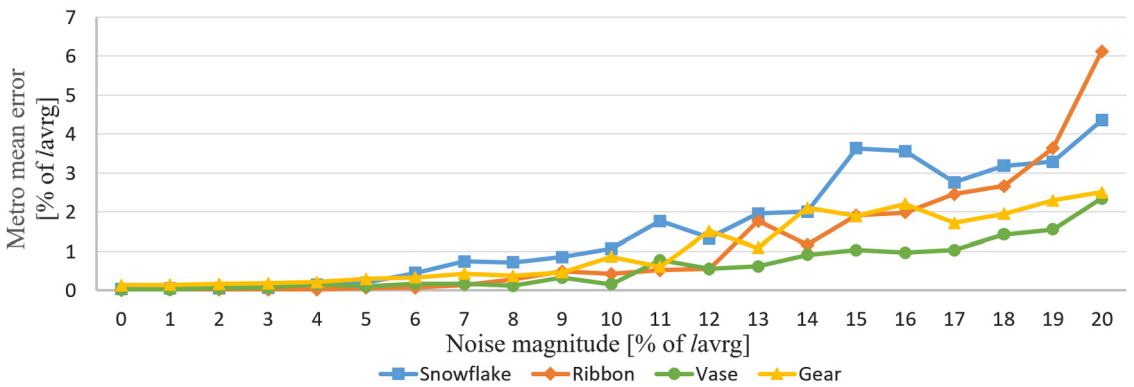
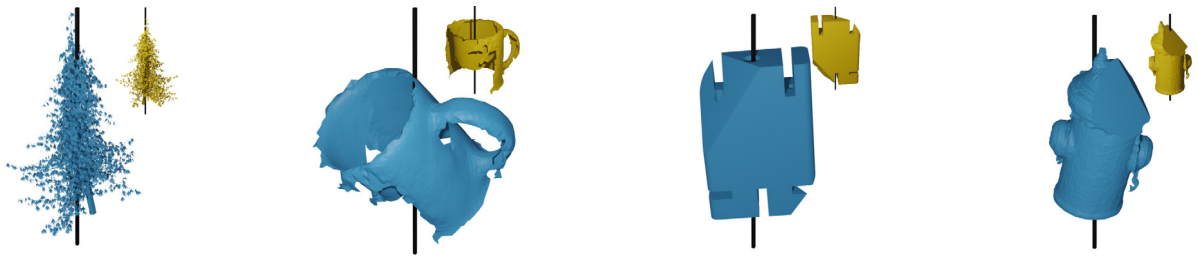


Fig. 9. The influence of noise magnitude on the precision of our method measured by Metro mean error Cignoni et al. (1998).

symmetries seem to exist. More importantly, most of the methods aim at some specific type of data either in terms of required shape, object representation or some other properties. Therefore, providing any qualitative or quantitative comparison to the existing methods would be very difficult on its own, let alone providing an objective one. Nevertheless, we at least show an experimental comparison to a simple yet surprisingly well-performing PCA method and we provide a side-by-side comparison to the existing methods based on information available in the corresponding papers.

The PCA algorithm outputs three principal directions. Provided the input point set has strong rotational symmetry, one of the three directions should be the direction of its symmetry axis that passes through its centroid. We manually selected the one of the three possible axes that seemed to be the closest to a plausible symmetry axis. For the strongly symmetrical shapes (Ribbon, Gear, Snowflake, Vase or even Table) this approach provides a surprisingly accurate axis of rotational symmetry. However, when used on the weakly symmetrical objects, this method failed to detect a plausible axis or it was at least noticeably inaccurate, as can be seen in Fig. 10. This renders any PCA-based method practically unusable for detecting weaker symmetries, at least without aggressive refinement. Also, PCA obviously cannot find the rotation angle, and in practice, some algorithm would be needed to select the best of the three possible axes automatically.

We selected 100 random objects from the Thingi10k dataset Zhou and Jacobson (2016) (10000 objects represented by triangle meshes of varying shapes and properties), we removed all objects that obviously did not exhibit any rotational symmetry and replaced them with new random objects and we kept repeating this until we acquired 100 random objects with at least some form of rotational symmetry. Then we used both our method and PCA to find rotational symmetry for each of these 100 objects. As mentioned in Hruda et al. (2021) the symmetry measure is not very robust to strongly non-uniform point distribution (e.g., typical CAD models) and PCA has the same problem. However, since the objects are meshes, this can be easily solved by some surface sampling technique. As the Thingi10k dataset contains quite a few such problematic objects we first used MeshLab's Cignoni et al. (2008) stratified triangle sampling to resample the objects and we applied both symmetry detection methods on the resampled point sets. We manually observed the detected symmetries and determined their correctness as objectively as possible, the results are reported in Table 2. If the resulting symmetry



(a) Tree (b) Cup (c) Plastic (d) Hydrant

Fig. 10. Results of the PCA method, the best of the three possible axes was chosen manually, the small yellow images show the corresponding results of our method from Fig. 5 for easier comparison.

Table 2
Comparison of proposed method to PCA on 100 randomly selected resampled objects with some rotational symmetry.

	Axis correct	Axis imprecise	Axis incorrect	Angle correct	Angle imprecise	Angle incorrect
Proposed	92	5	3	93	2	5
PCA	78	15	7	-	-	-

Table 3
Side-by-side comparison of existing methods for rotational symmetry detection in 3D including the proposed method.

Method	General axis	Can detect angle	Can handle weak symmetry or missing parts	Input object representation	Supported shape
Proposed	Yes	Yes	Yes	Point set	Any
Sun and Sherrah (1997)	No	Yes	No	Any with normals	Any
Martinet et al. (2006)	No	Yes	No	Surface	Any
Korman et al. (2015)	No	Yes	No	Binary volume	Any
Sipiran (2017)	Yes	No	Yes	Triangle mesh	Rotational
Sipiran (2018)	Yes	No	Yes	Triangle mesh	Rotational
Ecins et al. (2018b)	Yes	No	Yes	Point set with normals	Rotational
Gothandaraman et al. (2020)	No	Yes	No	Point set	Any
PCA	No	No	No	Point set	Any

axis was plausible (one with a significant symmetry around it, not necessarily the best possible one for the object), it was marked as correct. If it was close to a plausible axis but with noticeable imprecision, it was marked as imprecise and in other cases it was marked as incorrect. The angle was marked as correct if the axis was correct or imprecise and it was within 1° of a correct angle w.r.t. the given axis or in case of circular symmetry. If the deviation was greater than 1° but up to 5°, it was marked as imprecise, otherwise incorrect (in case of incorrect axis the angle was marked as incorrect automatically). Note that, since PCA always finds three axes and we selected the best one of them manually for each object, the chance of at least one of them being imprecise at worst is quite large, which explains its rather low number of incorrect axis detections (still larger than our method nevertheless). However, the number of correct ones is noticeably larger for our method, which further has the advantage of also finding the angle of the rotational symmetry. The results of our method for all objects in the Thingi10k dataset can be found in the supplementary material SupplementDownload (2022).

Table 3 shows side-by-side theoretical comparison of the methods for detecting rotational symmetries in 3D mentioned in Section 1 including ours and PCA. The first column indicates whether the given method can detect symmetry with a general axis or only with an axis passing through some reference points which severely limits applicability for detecting weaker symmetries. The second column indicates whether the method also finds the rotation angle of the symmetry while the third column states if the method can handle weakly symmetrical objects or missing parts. The fourth column says what object representation is required on the input where point set is the most general one while a triangle mesh or other surface representations are the least general ones and binary volume stands for any representation where it can be determined whether a point lies inside or outside the input shape. At last, the fifth column states what type of shape the method requires to work properly. Note, however, that we did not experimentally verify any of this information (except for PCA and our method) and we state what we derived from the corresponding publications. The proposed method appears to be the most general one of all, detecting symmetry with general axis plus the rotation angle, handles well objects with weak symmetries and missing parts, only requires a set of points on the input and does not put any constraints on the shape of the input object.

4. Limitations

The proposed method also has some limitations. First of all, it obviously depends on the quality and properties of the selected underlying symmetry plane detection approach, in our case Hruda et al. (2021). As already mentioned, the symmetry measure is not suitable for objects with very non-uniformly distributed points and the method can be expected to fail on such data. However, in the case of triangle mesh or other surface representation, it can easily be resolved using some surface sampling technique. Also, our method might be rather fast but still significantly slower than the original symmetry plane detection algorithm of Hruda et al. (2021). This is primarily due to the increased target point count of X_{simp} from 1000 to 3000. Therefore, in the future it would be more than appropriate to find a way of keeping the method sufficiently robust without this change. At last, it can be expected that the method will fail with non-negligible probability in cases when the rotational symmetry is not composed of reflectional ones. But, while this might be true theoretically, it turns out that in practice the method works even in these cases as we showed and explained throughout the previous sections.

5. Conclusion

We proposed a new method for detecting rotational symmetries in 3D objects using a simple idea of combining potential reflectional symmetries and modifying the optimization step from Hruda et al. (2021) for rotations with a quaternion-based parameterization which seems to be novel in the symmetry detection field. We showed that the method is well capable of detecting both strong and weak symmetries including partial symmetries for objects with missing parts and is robust to noise as well as computationally efficient. The robustness and precision of the proposed method could potentially be improved even more by adjusting some of its parameters, such as the target point count of the simplified point set or the number of best-selected rotation candidates. This would, however, come with the cost of larger computation time and as the results suggest, it is unnecessary in most cases. These adjustments can be used in the opposite way as well, i.e. if precision and robustness are not a priority, the method can be made faster. Furthermore, to our knowledge, the proposed method is the most general one of the existing methods for rotational symmetry detection. Also, the main theoretical contributions of our paper could be used in other context as well and possibly combined with different approaches for symmetry plane detection. The implementation of our method is in the supplementary material, which can be downloaded from SupplementDownload (2022).

The parameters of the proposed method are overall the same as those of the underlying symmetry plane detection method and their default values have been introduced throughout the paper. The meaning and effect of each parameter have already been described rather thoroughly in the supplementary material for Hruda et al. (2021). The only additional parameters in the proposed method are ϵ_Q, ϵ_s (see Section 2.3), which have the same effect on the candidate rotations as the δ parameter (see Section 2.1) has on candidate planes in the original method, and M (see Sections 2.3 and 2.4) which has the same effect as S (see Section 2.1) in the original method.

As described in Hruda et al. (2021), the symmetry measure allows incorporating weights that can be used to adjust importance of given point pairs based on some additional information, such as similarity of some local descriptors. In the future, we plan to examine this possibility also in the context of our rotational symmetry detection method. Also, it might be possible to extend our core idea to find symmetries w.r.t. isometries which can be composed of three reflections. We leave this for future work as well.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by the Czech Science Foundation, project GACR 21-08009K Generalized Symmetries and Equivalences of Geometric Data. Lukáš Hruda was also funded by Ministry of Education, Youth and Sports of the Czech Republic – the student research project SGS-2022-015 New Methods for Medical, Spatial and Communication Data.

We thank to all referees for their valuable comments which helped us improve the paper.

References

Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G., 2008. Meshlab: an open-source mesh processing tool. In: Eurographics Italian Chapter Conference, pp. 129–136.

Cignoni, P., Rocchini, C., Scopigno, R., 1998. Metro: measuring error on simplified surfaces. In: Computer Graphics Forum. Wiley Online Library, pp. 167–174.

Combès, B., Hennessy, R., Waddington, J., Roberts, N., Prima, S., 2008. Automatic symmetry plane estimation of bilateral objects in point clouds. In: Computer Vision and Pattern Recognition, IEEE Conference on. 2008. CVPR 2008. IEEE, pp. 1–8.

Ecins, A., Fermuller, C., Aloimonos, Y., 2017. Detecting reflectional symmetries in 3D data through symmetrical fitting. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1779–1783.

Ecins, A., Fermüller, C., Aloimonos, Y., 2018a. Cluttered tabletop dataset. http://www.aecins.umiacs.io/projects/symseg/. (Accessed 2 December 2021).
Ecins, A., Fermüller, C., Aloimonos, Y., 2018b. Seeing behind the scene: using symmetry to reason about objects in cluttered environments. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 7193–7200.
Funk, C., Lee, S., Oswald, M.R., Tsogkas, S., Shen, W., Cohen, A., Dickinson, S., Liu, Y., 2017. 2017 iccv challenge: detecting symmetry in the wild. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1692–1701.
Gallier, J., 2011. The Cartan–Dieudonné theorem. In: Geometric Methods and Applications. Springer, pp. 231–280.
Goldman, R., 2010. Rethinking Quaternions: Theory and Computation. Morgan and Claypool Publishers.
Gothandaraman, R., Jha, R., Muthuswamy, S., 2020. Reflectional and rotational symmetry detection of CAD models based on point cloud processing. In: 2020 IEEE 4th Conference on Information & Communication Technology (CICT). IEEE, pp. 1–5.
Horn, B.K.P., 1984. Extended Gaussian images. Proc. IEEE 72, 1671–1686.
Hruda, L., Dvořák, J., Váša, L., 2019. On evaluating consensus in RANSAC surface registration. In: Computer Graphics Forum. Wiley Online Library, pp. 175–186.
Hruda, L., Kolingerová, I., Váša, L., 2021. Robust, fast and flexible symmetry plane detection based on differentiable symmetry measure. The Visual Computer URL: https://doi.org/10.1007/s00371-020-02034-w.
Klingenberg, C.P., 2015. Analyzing fluctuating asymmetry with geometric morphometrics: concepts, methods, and applications. Symmetry 7, 843–934.
Kohek, Š., Strnad, D., Žalik, B., Kolmanič, S., 2019. Interactive synthesis and visualization of self-organizing trees for large-scale forest succession simulation. Multimed. Syst. 25, 213–227.
Korman, S., Litman, R., Avidan, S., Bronstein, A., 2015. Probably approximately symmetric: fast rigid symmetry detection with global guarantees. In: Computer Graphics Forum. Wiley Online Library, pp. 2–13.
Li, B., Johan, H., Ye, Y., Lu, Y., 2016. Efficient 3D reflection symmetry detection: a view-based approach. Graph. Models 83, 2–14.
Lipman, Y., Chen, X., Daubechies, I., Funkhouser, T., 2010. Symmetry Factored Embedding and Distance. ACM Transactions on Graphics (TOG). ACM, p. 103.
Liu, D.C., Nocedal, J., 1989. On the limited memory BFGS method for large scale optimization. Math. Program. 45, 503–528.
Lowe, D.G., 1999. Object recognition from local scale-invariant features. In: Proceedings of the Seventh IEEE International Conference on Computer Vision. IEEE, pp. 1150–1157.
Martinet, A., Soler, C., Holzschuch, N., Sillion, F.X., 2006. Accurate detection of symmetries in 3D shapes. ACM Trans. Graph. (TOG) 25, 439–464.
Mavridis, P., Sipiran, I., Andreadis, A., Papaioannou, G., 2015. Object completion using k-sparse optimization. In: Computer Graphics Forum. Wiley Online Library, pp. 13–21.
Mitra, N.J., Guibas, L.J., Pauly, M., 2006. Partial and Approximate Symmetry Detection for 3D Geometry. ACM Transactions on Graphics (TOG). ACM, pp. 560–568.
Nagar, R., Raman, S., 2020. 3DSymm: robust and accurate 3D reflection symmetry detection. Pattern Recognit., 107483.
Podolak, J., Shilane, P., Golovinskiy, A., Rusinkiewicz, S., Funkhouser, T., 2006. A planar-reflective symmetry transform for 3D shapes. ACM Trans. Graph. (TOG) 25, 549–559.
Pottmann, H., Huang, Q.X., Yang, Y.L., Hu, S.M., 2006. Geometry and convergence analysis of algorithms for registration of 3D shapes. Int. J. Comput. Vis. 67, 277–296.
Savriama, Y., Gerber, S., 2018. Geometric morphometrics of nested symmetries unravels hierarchical inter- and intra-individual variation in biological shapes. Sci. Rep. 8, 1–12.
Savriama, Y., Klingenberg, C.P., 2011. Beyond bilateral symmetry: geometric morphometric methods for any type of symmetry. BMC Evol. Biol. 11, 1–24.
Schiebener, D., Schmidt, A., Vahrenkamp, N., Asfour, T., 2016. Heuristic 3D object shape completion based on symmetry and scene context. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, pp. 74–81.
Shi, Z., Alliez, P., Desbrun, M., Bao, H., Huang, J., 2016. Symmetry and orbit detection via lie-algebra voting. In: Computer Graphics Forum. Wiley Online Library, pp. 217–227.
Simari, P., Kalogerakis, E., Singh, K., 2006. Folding meshes: hierarchical mesh segmentation based on planar symmetry. In: Symposium on Geometry Processing, pp. 111–119.
Sipiran, I., 2017. Analysis of partial axial symmetry on 3D surfaces and its application in the restoration of cultural heritage objects. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 2925–2933.
Sipiran, I., 2018. Completion of cultural heritage objects with rotational symmetry. In: Proceedings of the 11th Eurographics Workshop on 3D Object Retrieval, pp. 87–93.
Sipiran, I., Gregor, R., Schreck, T., 2014. Approximate symmetry detection in partial 3D meshes. In: Computer Graphics Forum. Wiley Online Library, pp. 131–140.
Speciale, P., Oswald, M.R., Cohen, A., Pollefeys, M., 2016. A symmetry prior for convex variational 3D reconstruction. In: European Conference on Computer Vision. Springer, pp. 313–328.
Sterck, H.D., 2013. Steepest descent preconditioning for nonlinear GMRES optimization. Numer. Linear Algebra Appl. 20, 453–471.
Sun, C., Sherrah, J., 1997. 3D symmetry detection using the extended Gaussian image. IEEE Trans. Pattern Anal. Mach. Intell. 19, 164–168.
SupplementDownload, 2022. Rotational symmetry detection in 3D using reflectional symmetry candidates and quaternion-based rotation parameterization: supplementary material. https://geosym.zcu.cz/cagd2022. (Accessed 3 August 2022).
Tevs, A., Huang, Q., Wand, M., Seidel, H.P., Guibas, L., 2014. Relating shapes via geometric symmetries and regularities. ACM Trans. Graph. (TOG) 33, 1–12.
Wendland, H., 1995. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. Adv. Comput. Math. 4, 389–396.
Wright, S., Nocedal, J., et al., 1999. Numerical Optimization. Springer Science, vol. 35, p. 7.
Zhou, Q., Jacobson, A., 2016. Thingi10k: a dataset of 10,000 3D-printing models. ArXiv preprint. arXiv:1605.04797.