

# Relating Shapes via Geometric Symmetries and Regularities

Art Tevs  
MPI Informatics  
Stanford University

Qixing Huang  
Stanford University

Michael Wand  
Utrecht University

Hans-Peter Seidel  
MPI Informatics

Leonidas Guibas  
Stanford University



Figure 1: Chandeliers data set with matched lamps over several geometrically varying configurations.

## Abstract

In this paper we address the problem of finding correspondences between related shapes of widely varying geometry. We propose a new method based on the observation that symmetry and regularity in shapes is often associated with their function. Hence, they provide cues for matching related geometry even under strong shape variations. Correspondingly, we decompose shapes into overlapping regions determined by their regularity properties. Afterwards, we form a graph that connects these pieces via pairwise relations that capture geometric relations between rotation axes and reflection planes as well as topological or proximity relations. Finally, we perform graph matching to establish correspondences. The method yields certain more abstract but semantically meaningful correspondences between man-made shapes that are too difficult to recognize by traditional geometric methods.

**CR Categories:** I.3.5 [Computing Methodologies]: Computer Graphics—Computational Geometry and Object Modeling;

**Keywords:** shape correspondences, symmetry, structural regularity, shape understanding

**Links:** [DL](#) [PDF](#)

## 1 Introduction

One of the big challenges in modern computer graphics is to organize, structure, and to some extent “understand” visual data and 3D geometry. We now have large data bases of shape collections, public ones such as Trimble 3D Warehouse<sup>TM</sup>, as well as other academic, commercial, and possibly in-house collections. These provide numerous 3D assets in varying composition, style, and level-of-detail.

Being able to explore, structure, and utilize the content in such data remains one of the key challenges in computer graphics research.

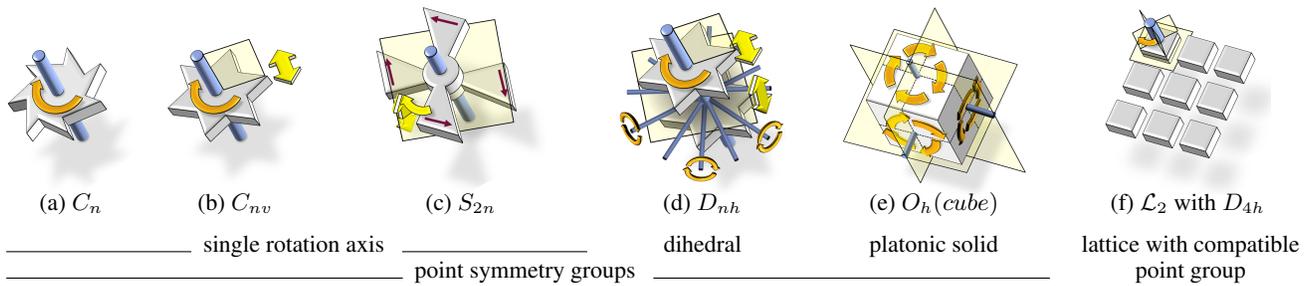
A core building block for this task is the ability to establish *correspondences* [van Kaick et al. 2011]: We need to be able to match objects, and parts of them according to multiple notions of similarity. In other words, correspondence computation means the recovery of a latent equivalence relation that identifies similar pieces of shapes. The equivalence classes unveil redundancy in a model collection, thereby providing a basic structuring tool.

As detailed in Section 2, a large body of techniques is available for estimating correspondences that involve simple transformations with few degrees of freedom, such as rigid motions or affine maps. More recently, this also comprises intrinsic isometries, which permit matching of objects in varying poses as long as there is no substantial intrinsic stretch.

However, matching semantically related shapes with considerable variation in geometry remains a very difficult problem. Objects such as humans in different poses can be matched by smooth deformation functions using numerical optimization [Allen et al. 2003], typically requiring manual initialization. Relating collections of shapes without manual intervention has become feasible if the collection forms a dense enough sampling of a smooth shape space [Ovsjanikov et al. 2011; Huang et al. 2012; Kim et al. 2012]. However, these assumptions are not always met. In particular, man-made artifacts of related functionality can have very different shape with non-continuous variability.

We propose a new technique for relating shapes of similar functionality but potentially very different geometry based on shared symmetry properties, complementary to extant approaches. We build upon the observation that symmetry and regularity (we will use these terms interchangeably) are often related to functionality: For example, a windmill is rotationally symmetric but not reflectively because it must be driven by wind. The same holds for screws and propellers. However, a windmill is different from a plane or a submarine in that the propeller is attached to a typically rotationally symmetric building, joining the rotation axes in a roughly perpendicular angle, unlike the other two. The idea of our paper is to systematically extract such regularity properties and pairwise relations between their invariant sets (rotation axes, reflection planes) in order to characterize shape families at a high level of abstraction.

We detect regularity in the form of Euclidean symmetry groups acting on the shape geometry. We describe the symmetry structure of shapes by a pairwise structure graphs on pieces of geometry. The nodes are symmetries and edges are pairwise relations between symmetries, such as angles or topological relations, such as set inclusion and adjacency. Afterwards, we employ graph matching to find correspondences between shapes in this more abstract and in-



**Figure 2:** Examples of Euclidean symmetry groups. Point-symmetries combine rotations and reflections and differ in the number of rotation axes and reflection planes, and their mutual relation. The remaining cases combine infinite translational lattices with optional point groups.

variant representation. Our method is complementary to traditional geometric methods. It does not require training data or collections to co-analyze but can directly match pairs of shapes.

Our approach is essentially a second-order shape comparator, comparing shapes not by traditional geometry-matching between the shapes, but by comparing the geometry matches (symmetries) within each shape to each other. Such symmetries and self-relations capture aspects of the shape that are important both to its function and to our perceptual understanding of it. While earlier work has addressed the estimation of such shape self-structure, this paper provides a canonical way of encoding these relations in a graph and for comparing such symmetry graphs even when the symmetries present are not exactly the same, effectively exploiting a novel metric structure in the space of symmetries.

## 2 Related Work

Correspondence in 3D geometry has received a lot of attention in recent years; van Kaik et al. [2011] provide a good survey. There are mature solutions for matching geometry under the action of groups of transformations with few degrees of freedom such as rigid matching. Matching objects that are semantically related but have widely varying geometry is very much an open problem; only a few specialized solutions exist. One possibility is to use machine learning to associate semantic categories with geometry [Kalogerakis et al. 2010]. This approach is versatile and powerful, but requires supervised training. Unsupervised methods rely on clustering or manifold reconstruction in descriptor space [Ovsjanikov et al. 2011; Sidi et al. 2011], which again requires a certain amount of geometric similarity. The same holds for recent attempts to extract correspondences in shape collections by combining a larger number of similar, pairwise maps between shapes [Nguyen et al. 2011; Huang et al. 2012; Kim et al. 2012] by enforcing transitive closure of correspondences. This assumes a densely sampled, continuous shape spaces, where geometrically similar neighbor models exists for each example.

Our approach is based on the observation that symmetry is often related to object function. The idea of using symmetry to characterize shapes is well established in the field: Thrun et al. [2005] utilize symmetry for shape completion. Mitra et al. [2010] use symmetry as a cue to understand the semantics of mechanical assemblies. Gal et al. [2009], Bokeloh et al. [2012], and others, have used symmetry properties of shapes as invariants for content-aware shape editing.

Kazhdan et al. [2004] use global rotational and reflective symmetry as shape descriptors. Our approach is more general as it examines constellations of multiple symmetric parts and their global relations. Hierarchies of symmetry relations have been used by Martinet et al. [2007] for compression. Wang et al. [2011; 2012] propose symmetry hierarchies for shape analysis with applications to segmentation and editing. Hierarchical decompositions have also

been employed by Cullen et al. [2011] for relating shapes and creating shape variations, as well as by Simary et al. [2009] for shape segmentation; the method incorporates relations such as perpendicularity of symmetry planes between adjacent segments.

Our method is different in two important aspects: First, we aim at correspondence estimation and develop the necessary tools (structure model, graph decomposition, comparison functions) for this task. Second, our technical approach is different: Hierarchies are usually not unique, requiring non-canonical choices. We therefore build graphs that reflect algebraic symmetry properties and topological aspects of the shape. This does include relations of hierarchical containment. However, our representation does not compress detected hierarchies. This maintains a richer set of structural relations, making it better suited for correspondence problems. Van Kaik et al. [2013] obtain information from optimizing hierarchies for shape collections; our paper restricts itself to pairwise matching, where this is not possible. Graph-based matching has previously been studied by Fisher et al. [2011] with restriction to local spatial proximity relations while we use non-local invariants. Liu et al. [2012] also use symmetries to aid correspondence computation; a (single) dominant intrinsic involution is computed [Xu et al. 2009] as reference for dense correspondences; our method obtains coarse correspondences from a large set of different extrinsic symmetries.

Symmetry groups have been used for matching in the image domain by [Hauagge and Snavely 2012; Henderson et al. 2012], but the approaches are restricted to local descriptors while we seek to form global invariants that consist of complex arrangements of regularity patterns. Our approach is motivated by algebraic regularity models proposed by Pauly et al. [2008] and Bokeloh et al. [2012] for characterizing families of shapes. Our representation is more general: It is not restricted to lattices but captures all possible types of symmetry groups, and we can handle general geometric and topological relations of symmetric parts. However, we do not obtain generative models; we cannot create shape variations but just aim at identifying correspondences.

## 3 Regularity Graphs

In this section, we introduce *regularity graphs*, the shape abstraction employed in our approach.

**Graph nodes:** The nodes in the graph correspond to subsets of the geometry that each have a fixed regularity structure. Each node is annotated with its symmetry group and represents the maximal connected subset of the input model with that symmetry; multiple nodes might overlap. To facilitate matching of related but not fully identical structures (such as cubes and cuboids), we store symmetry groups that characterize these objects in a factorized graph, and represent each group by a small set of overlapping nodes. We recap algebraic symmetry in Sections 3.1/3.2 and discuss factorization in Section 3.3.

Class	Type	Comment
Involutorial (reflections, 2fold rotations)	$E = C_1$	identity group, no transformation
	$C_i$	inversion, i.e. reflecting by center point
	$C_s = \sigma$	reflection on a single plane
	$C_2$	2-fold rotation
Cyclic (one rotational generator)	$C_n$	rotational symmetry group with $n$ -fold rotation by $\frac{360}{n}$ degrees
	$C_{nh}$	combination of $C_n$ and $C_s$ with perpendicular reflection plane
	$C_{nv}$	$n$ -fold rotation with $n$ -reflection planes colinear with rotation axis
	$S_{2n}$	$n$ -fold rotation around improper axis, i.e. a rotation combined with a reflection
Dihedral (two rotational generators)	$D_n$	$n$ rotations by $\frac{360}{n}$ degrees (main axis) with perpendicular $180^\circ$ rotations
	$D_{nh}$	$D_n$ with perpendicular reflection plane. Coinciding $v$ -type reflections included.
	$D_{nd}$	$D_n$ group with $v$ -type reflections perpendicular to both rotation axes.
Polyhedral (more than two rot. gen.)	$T, T_h, T_v$	tetrahedral symmetry group
	$O, O_h$	octahedral symmetry group
	$I, I_h$	icosahedral symmetry group
Lattice (translational)	$\mathcal{L}^1, \mathcal{L}^2, \mathcal{L}^3$	lattice structure with 1,2,3 translational generators.

**Table 1:** Overview of all Euclidean point symmetry groups and lattices [Hahn 2002], see also Fig. 3 for the notation.

**Graph edges:** Edges represent pairwise relations, encoding topology (adjacency, containment) and symmetry relations that relate rotation axes and reflection planes, for example by coincidence or angles. Pairwise relations are described in Section 3.3.2.

### 3.1 Symmetry and Regularity

We build our approach on the classical model of groups of transformations for characterizing symmetry.

**Euclidean 3D geometry:** We consider the Euclidean space  $\mathbb{R}^3$  and pieces of geometry  $\mathcal{S} \subset \mathbb{R}^3$ . Further, we consider *transformations*  $\mathbf{T} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ . We restrict ourselves to the group  $E(3)$  of rigid motions, i.e., all combinations of translations, rotations, and reflections. This is a reasonable choice for man-made shapes, as their manufacturing and operation frequently involve rigid motions (assembly of parts, lathing, gears, axles, etc).

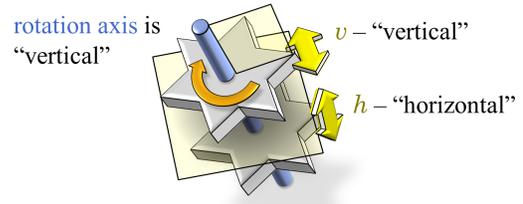
**Symmetry groups:** Symmetries are formally described by groups of transformations:  $\mathcal{S} \subset \mathbb{R}^3$  is symmetric with respect to one of its *symmetry groups*  $G \subset E(3)$  if it is invariant under the group actions of  $G$ , i.e., any  $\mathbf{T} \in G$  can be applied to  $\mathcal{S}$  unnoticed. We use  $\langle P \rangle$  to denote the group *generated* by all products of elements of  $P \subset E(3)$ . Elements  $p \in P$  are called *generators*.

### 3.2 Euclidean Symmetry Groups

Euclidean symmetries are well understood; there exists a complete classification of all possible types [Hahn 2002]. The understanding of the structure of Euclidean symmetry is essential for our approach; we therefore give a brief summary of the theory below. We organize this overview along the properties of discrete vs. continuous groups, and point groups vs. translational lattices.

#### Discrete Point Symmetry Groups

Discrete subgroup  $G \subset E(3)$  consist of isolated transformations: For each  $\mathbf{T} \in G$  there is an empty  $\epsilon$ -ball with  $\epsilon > 0$  around  $\mathbf{T}$  in  $E(3)$ . Point symmetries have at least one point  $\mathbf{x}_0$  that is invariant under all transformations. They are combinations of rotations and reflections, excluding translations. All discrete point groups are listed in Table 1. We now count the number of rotational generators:



**Figure 3:** We use Schönflies notation. Mirror planes are denoted with  $v$  (vertical) and  $h$  (horizontal), assuming that the main rotational symmetry axis is “vertical”. Combining  $v$  and  $h$  yields  $D_{nh}$ .

**Reflections (no rotation):** In 3D, there are two purely reflective groups: reflections at a point (denoted by  $C_i$ ) and at a plane ( $C_s$ ). Both consist of two elements, the identity and a reflection that is an *involution*, i.e., the reflection is its own inverse.

**Cyclic symmetry (1 rotation + reflections):** *cyclic symmetry groups*  $C_n, n \in \mathbb{N}$  have one rotational generator; they consist of repeated rotation by  $360/n$  degree (Fig. 2a).  $C_2$  is the third possible involutorial group ( $180^\circ$ -rotations are self-inverse). Cyclic groups can be combined with additional reflections with the reflection plane either containing the rotation axis (“ $v$ ”-type  $C_{nv}$ ) (Fig. 2b) or being perpendicular to it (“ $h$ ”-type  $C_{nh}$ ) (Fig. 3).  $C_{nv}$  groups have  $n$  reflection planes, replicated by the rotational symmetry. An additional type is  $S_{2n}$ , generated by a combination of reflection and rotation (*improper rotation*, Fig. 2c).

**Dihedral symmetry (2 rotations + reflections):** Rotational  $C_n$  symmetry groups can also be combined with rotation by  $180^\circ$  as additional involutions. This yields *dihedral symmetry groups*, denoted by  $D_n$ ; they are generated by two perpendicular rotations. The  $180^\circ$ -rotation is multiplied  $n$ -fold by the  $C_n$  symmetry (Fig. 2d).  $D_n$  groups can still be combined with further reflection planes. A perpendicular “ $h$ ”-type reflection yields  $D_{nh}$  (Fig. 2d). Because the  $180^\circ$  rotation is the concatenation of the “ $h$ ” and “ $v$ ” reflections,  $D_{nh}$  also contains the  $v$ -type reflections such that the involutorial rotation axes lies within the  $v$ -type reflection plane (Fig. 3). A different option is to add a  $v$ -axis that is orthogonal to the involutorial rotation, denoted by  $D_{nd}$ . Here, the  $180^\circ$ -rotation axes lie in between the  $v$ -planes, bisecting the angles in between them.

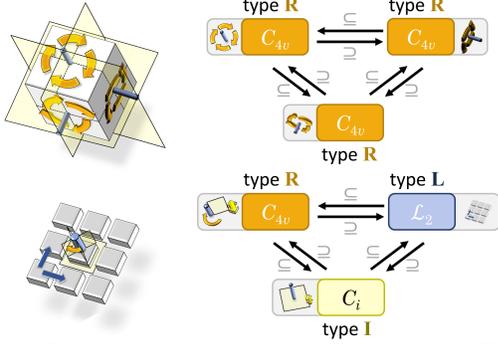
**Platonic solids (more than two rotational generators):** The remaining cases are the symmetry groups of the platonic solids: *Tetrahedral symmetry* ( $T$ , combined with reflections as  $T_h$  and  $T_v$ ), *octahedral symmetry* (same symmetry as the cube, 24 rotations denoted by  $O$ , or 48 rotations and reflections, denoted by  $O_h$ , see Fig. 2e), and *icosahedral symmetry* ( $I, I_h$ , same as dodecahedra).

#### Discrete Lattices

Point groups cannot contain global translations. If we choose  $k = 1, 2, \text{ or } 3$  linearly independent translations as generators, we obtain a translational lattice group  $\mathcal{L}_k$  (commutative  $k$ -parameter groups [Pauly et al. 2008]). We call them *lattices* and the geometry generated by their group action *grids*. Naturally, grids must always be infinite in extent. A helix is a 1-parameter lattice of a combined rotation and translation. In combination with point symmetry groups we obtain *crystallographic lattices* [Hahn 2002] (Fig. 2f).

#### Continuous Symmetry

The fine-grained limit yields continuous symmetries [Gelfand and Guibas 2004]. Continuous and discrete symmetry can also be mixed: Except from the sphere, any of the continuous degrees of freedom can be replaced by a discretized version. Our current implementation detects only discrete symmetry.



**Figure 4:** Graph representations for symmetric objects. We factorize symmetry groups in graphs of overlapping nodes.

### 3.3 Factorization Model

We now encode symmetry properties to be suitable for comparison by graph matching. *Partial matching* is particularly important: regularity structures should have similar graphs even if they are not identical in all aspects. For example, a cube should be similar to a rectangular cuboid in its graph representation. We therefore propose a representation that factors all of the symmetry groups listed above into combinations of elementary 1-parameter groups.

**Factorization model:** Point groups can be represented by combining involutions and/or 1-parameter rotation groups (including improper rotations and helices). Lattices add translational degrees of freedom. We therefore use three corresponding basic node types in our regularity graph:

- **Involutions (I):** point/plane reflections, rotations by  $180^\circ$ .
- **Rotations (R):** 1-parameter groups generated by a rotation, including improper and helical cases.
- **Lattices (L):** 1, 2, 3-parameter groups generated by 1-3 linearly independent translations.

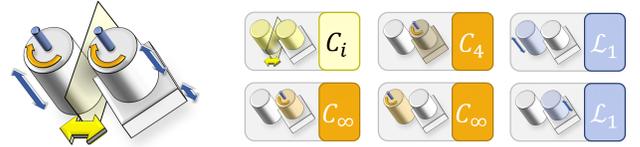
While this makes complex structures more comparable at the graph level, we nonetheless need to take a few precautions to avoid creating an overly redundant representation:

Recording all 1-parameter groups would create redundancy that obscures the structure for the later graph matching. For example, in a checker-board grid (2-parameter lattice), any regular line (for example, a diagonal) would create a separate 1-parameter lattice. Therefore, higher-order lattices are not factored into 1-parameter groups but keep a single L-node with  $k = 1$ -, 2-, or 3. Rotations use a similar compression: The  $C_{nv}$ ,  $D_n$ ,  $D_{nh}$ , and  $D_{nd}$  symmetries replicate their additional involutions  $n$ -fold by the main rotational symmetry, again creating redundant nodes. We therefore store additional involutions directly within the main R-node. Additionally, in case of rotations, lower order groups are replaced by higher order if they are subgroups, e.g. a group  $C/D_{n[h/d/v]}$  is a subgroup of  $C/D_{kn[h/d/v]}$  for any  $k \in \mathbb{N}$ . Examples of factored representations are shown in Fig. 4. Our representation encodes all original symmetry properties; while different choices of rules would be possible at this point, no information is discarded.

#### 3.3.1 Building Graph Nodes

So far, we have only consider globally symmetric objects. We now relax the model towards partial symmetry: we permit incomplete groups and perform symmetry-based segmentation into graph nodes.

**Incomplete groups:** We accept incomplete “excerpts” of a full symmetry group. For lattice structures and for rotational regularity,



**Figure 5:** We decompose the scene into overlapping sets of maximal symmetry groups. These sets form nodes in a graph (for clarity, not all possible nodes are shown).

we require at least three consecutive symmetric objects; involutions (naturally) require only two.

**Graph nodes:** We define graph nodes as maximal possible subsets of the geometry with constant symmetry properties. We tag each surface point  $\mathbf{x} \in \mathcal{S}$  with the maximal symmetry group  $G \subset E(3)$  that it can be part of, i.e.,  $G(\mathbf{x}) \subseteq \mathcal{S}$ . The geometry of the nodes obtained can and usually does overlap; we do not aim at a disjoint segmentation yet (see Fig. 5). In the following, we denote these by  $\mathcal{V} = \{v_1, \dots, v_n\}$ . Each graph node represents the geometry  $\text{geo}(v_i) \subseteq \mathcal{S}$ . We store its type (I/R/L), the generators ( $\mathbf{T} \in E(3)$ ) and the repetitions as an integer interval  $\mathcal{I}$ .

#### 3.3.2 Relations (Graph edges)

Given the nodes in the regularity graph, we build edges, denoted by  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ , that encode pairwise relations between graph nodes. Two directed edges  $(v_i, v_j), (v_j, v_i) \in \mathcal{E}$  between the nodes  $v_i$  and  $v_j$  are added to the graph when for a given threshold  $\epsilon_{prox}$ , which is typically set to 5% of the bounding box size, the shortest distance between any two points of both nodes is below the threshold, i.e.:

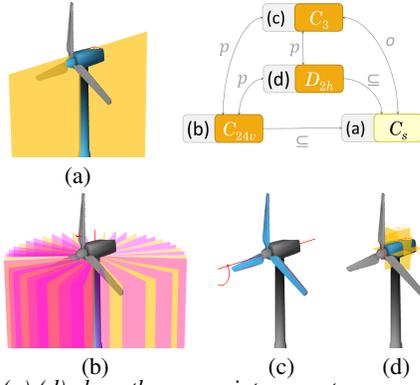
$$\min_{x \in \text{geo}(v_i)} \min_{y \in \text{geo}(v_j)} (\|x - y\|) \leq \epsilon_{prox}.$$

We extend our regularity graph model by including flags for each of the edge representing a set of discrete relations between two nodes in our model. Each of these flags describes special properties of the topology as well as of the symmetric relation between two nodes. We distinguish four types of discrete relations and encode them into a binary functions  $R_p : \mathcal{E} \rightarrow 0, 1$ , with  $p = prox, over, sub, sym$ :

- **Proximity relation:** Any edge connecting two adjacent nodes, i.e.,  $\text{geo}(v_i) \cap \text{geo}(v_j) = \emptyset$  is marked with a *proximity flag*, i.e.  $R_{prox}(\{v_i, v_j\}) = 1$  and  $R_{prox}(\{v_j, v_i\}) = 1$ .
- **Subset and overlap relations:** If two nodes are overlapping, i.e.  $\text{geo}(v_i) \cap \text{geo}(v_j) \neq \emptyset$ , both edges  $(v_i, v_j)$  and  $(v_j, v_i)$  are marked with an *overlap flag*, i.e.  $R_{over}(\{v_i, v_j\}) = 1$  and  $R_{over}(\{v_j, v_i\}) = 1$ . Additionally in case if the geometry of node  $v_i$  is covered by the geometry of  $v_j$ , i.e.  $\text{geo}(v_i) \subseteq \text{geo}(v_j)$ , we add a *subset flag*, i.e.  $R_{sub}(\{v_i, v_j\}) = 1$ , encoding hierarchical containment. Equality is given by two opposing subset edges.
- **Symmetry relation:** We add a *symmetry relation flag* to an edge, i.e.  $R_{sym}(\{v_i, v_j\}) = 1$ , if the vectors characterizing the invariant sets (i.e. rotation axis of a rotational group and reflection plane normal of the  $C_s$  group, respectively) are approximately the same. For edges connecting lattice structures, we interpret the generating translation as axis or two of them as a plane and apply the same criterion.

In our implementation, “approximately the same” means that we allow deviations of up to 5% of the bounding box size for distances ( $\epsilon_{dist}$ ), and  $15^\circ$  for angles ( $\epsilon_{angle}$ ); see Table 3.

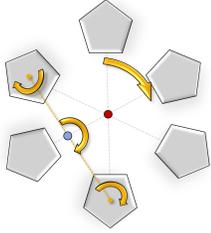
The proposed binary relations are not exhaustive; further invariants such as co-occurrence could be added. Fig. 6 shows an example of a regularity graph of a windmill as has been detected with our approach.



**Figure 6:** (a)-(d) show the one-point symmetry groups of a windmill (purple colored reflection planes are missing in the data). Tags  $p$ ,  $o$  and  $\subseteq$  indicate respectively proximity, overlap and subset relationships as described in Section 3.3.2.

### Hierarchies and Redundant Nodes

The graph model outlined above captures hierarchically nested structures through subset relations. In contrast to previous work, such as Wang et al. [2011; 2013], it does not perform *compression*. This means, all children appear as separate nodes in the graph. This allows the graph to encode further relations of subsets of the children to the rest of the scene. This is beneficial for our target application because we do not need to decide a priori which relations will be meaningful for shape matching.



**Figure 7:** Redundant  $C_2$  nodes emerge from hierarchical nesting of  $C_{5v}$  within  $C_{6v}$  nodes.

However, hierarchically nested symmetries can induce further symmetries, as shown in Fig. 7. In this example, additional involutions arise from the repeated sub-element being symmetric by itself. An additional filter can remove this redundancy: For any pair of detected nodes  $v_i$  and  $v_j$ , where  $\text{geo}(v_i) \subseteq \text{geo}(v_j)$  we check if applying the transformation  $\mathbf{T}_j$  of node  $v_j$  to the geometry of node  $v_i$  yields matching geometry, i.e., if  $\mathbf{T}_j(\text{geo}(v_i)) = \text{geo}(v_j)$ . If this is the case, node  $v_i$  is removed from the graph.

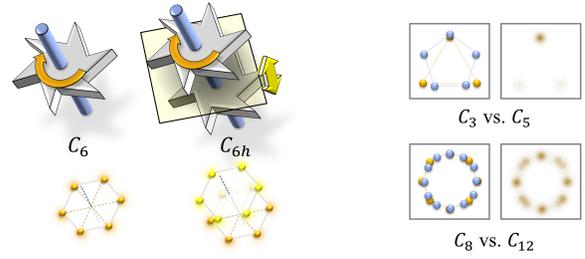
In our experiments, filtering leads to slightly smaller graphs (between 5-20%) but it had only a small influence on correspondence estimation.

## 4 Graph Matching

Given regularity graphs of multiple shapes, our task is now to find a matching between these graphs. Hence, the last ingredient of our model is a similarity function that gives us scores for comparing pairs of nodes and edges (*unary* and *pairwise* scores). We use positive scores, where zero means no match and one means that a single structural aspect matches perfectly.

### 4.1 Comparing Nodes (Unary Scores)

We compare pairs of nodes by quantifying how similar their symmetries are. For subgroups  $G_1, G_2 \subset E(3)$ , we define  $\text{sim}(G_1, G_2) \geq 0$  to quantify their similarity. Kazhdan et al. [2003] propose a continuous measure for how much a piece of geometry is symmetric according to a given symmetry group, capturing  $C_n$  and  $C_s$  groups. In contrast, we compare more complex groups. For example, a  $C_{nh}$  and  $C_n$  node might be corresponding, having only the rotation axis in common, while the additional reflection is only



(a) In this example, only half of the group elements match. (b) Increasing number of rotations approach a continuous circle.

**Figure 8:** Conceptually, we compare groups by measuring their overlap, shown schematically by blurred points. (a)  $C_6$  scores one against another  $C_6$ , and  $\frac{1}{2}$  against  $C_{6h}$ . (b) A  $C_8$  group is more similar to a  $C_{12}$  than a  $C_3$  to a  $C_5$ .

a coincidence in the first model. Nonetheless, a match to a second  $C_{nh}$  would still make a stronger point.

We therefore base the rating on the *amount of common structure*: The more complex structures are matching, the higher the likelihood that this is not a coincidence. We first describe a geometric intuition: We can think of point groups  $G \subseteq SO(3)$  as a set of points on the manifold  $SO(3)$  (centering rotation axes / reflection planes). We then compare the point sets by measuring a suitable notion of distances between them (see Fig. 8). We implement an approximation of this conceptual idea, as detailed below.

**Involutions:** For pairs of involutions (I), we set a fixed score of one if the subtype ( $C_i, C_s, C_2$ ) matches, and zero otherwise.

**Rotations and complex point groups:** For a pair of cyclic and dihedral groups  $G_i, G_j$ , we approximate the overlap by

$$\text{sim}(G_i, G_j) = w \cdot \exp\left(-\left(\frac{2\pi/n_i - 2\pi/n_j}{\sigma_{rot}}\right)^2\right), \quad (1)$$

where  $\sigma_{rot}$  is a parameter that controls how sensitive the comparison should operate. We use  $\sigma_{rot} = 0.15 \approx 8.5^\circ$  for all our examples as this have been found empirically to be a good value. The numbers  $n_i$  and  $n_j$  are the number of rotations around the main axis of the group, for example  $C_6$  and  $D_{6h}$  both perform 6 rotations. The factor  $w$  models the *structural overlap* of both groups: For example,  $C_n$  against  $C_{nh}$  obtains half the score of matching  $C_n$  against itself. Table 2 lists the weights between all one-point group pairs and the involution group  $C_s$  as used in our implementation.

The exponential kernel (Eq. 1) measures similarity as a cross-correlation: The kernel is a Mercer kernel that acts as a scalar product in an implicitly defined feature space, which yields a consistent metric. Similarly, we use exponential kernels for the further comparisons. An alternative design would be to use subgroups for classification, but this would rate circles approximated by meshes with discrete  $C_{23}$  and  $C_{24}$  symmetry as totally distinct, which does not match their visual impression.

**Lattices:** When comparing two 1-parameter grids  $\mathbf{L}_a$  and  $\mathbf{L}_b$  we compare the lengths of their minimal generators  $\mathbf{u}_a$  and  $\mathbf{u}_b$  respectively:

$$\text{sim}(\mathbf{L}_a, \mathbf{L}_b) = \exp\left(-\frac{(\|\mathbf{u}_a\| - \|\mathbf{u}_b\|)^2}{\sigma_d^2}\right), \quad (2)$$

where  $\sigma_d$  controls the sensitivity of the comparison and is set to 0.3 in all examples used. 2-parameter grids obtain scores in the same way, however, taking the squared difference of the angles of both generators in the exponent. This model could in principle be refined by taking the rotational component (torsion for 1-parameter

	$C_s$	$C_n$	$C_{nh}$	$C_{nv}$	$D_n$	$D_{nh}$
$C_s$	[1]	[0]	[1/n]	[1/n]	[1/n]	[1/2n]
$C_n$	[0]	1	0.5	0.5	0.5	0.25
$C_{nh}$	[1/n]	0.5	1	0.5	0.5	0.5
$C_{nv}$	[1/n]	0.5	0.5	1	0.5	0.5
$D_n$	[1/n]	0.5	0.5	0.5	1	0.5
$D_{nh}$	[1/2n]	0.25	0.5	0.5	0.5	1

**Table 2:** Weight  $w$  in Equation 1. The weight describes the structural overlap between the complex rotational groups. For the combination of one-point group  $G$  with the involution group  $C_s$  the numbers in the table show the actual score  $\text{sim}(C_s, G)$ .

grids, curvature of the cylinder for 2 parameters) into account. Our implementation currently omits 3-parameter lattices.

## 4.2 Comparing Edges (Pairwise Scores)

The unary scores do not yet model the the arrangement of different parts. We therefore also employ edge scores that capture how nodes are arranged relative to each other. As discussed in Section 3.3.2, these edges in the regularity graph contain both discrete (subset, adjacency) as well as continuous relations (angles between direction vectors of invariant sets, i.e., normal of reflection planes and direction of rotation axes), which now need to be compared:

Let  $\alpha_1, \alpha_2$  be the angles measured between two such direction vectors of two pairs of nodes connected by two graph edges  $e_1, e_2$  and  $R_p$  be the binary indicator of the according discrete property. We can now combine the discrete and continuous properties into an edge score using the Gaussian similarity again:

$$\text{sim}(e_1, e_2) = \exp\left(-\left(\frac{\alpha_1 - \alpha_2}{\sigma}\right)^2\right) \sum_{p \in \mathcal{P}} w_p R_p(e_1) R_p(e_2), \quad (3)$$

where  $\mathcal{P}$  is the set of all discrete properties, i.e.  $\mathcal{P} = \{\text{prox}, \text{over}, \text{sub}, \text{sym}\}$  and the normalized weights  $w_p$  are used to emphasize certain properties over the other. In our implementation we set them to  $w_{\text{prox}} = w_{\text{over}} = w_{\text{sub}} = 0.15$  and  $w_{\text{sym}} = 0.55$  to emphasize the importance of coinciding symmetry axes and planes compared to other topological properties.

## 4.3 Matching function

Equipped with the unary and pairwise similarity functions we now discuss the matching approach to find correspondences between a pair of shapes. Let  $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$  and  $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2)$  be two regularity graphs, with nodes denoted by  $\mathcal{V}_1 = \{v_1, \dots, v_{n_1}\}$  and  $\mathcal{V}_2 = \{w_1, \dots, w_{n_2}\}$ . We compute correspondences as relations between nodes: Let

$$m : \{1, \dots, n_1\} \times \{1, \dots, n_2\} \rightarrow 0, 1 \quad (4)$$

be a binary indicator function that represents this relation:  $m_{ij}$  is one if and only if nodes  $v_i, w_j$  correspond. We determine the indicators by maximizing the quadratic assignment score  $Q(m)$ :

$$Q(m) = \sum_{\substack{v_i \in \mathcal{V}_1, \\ w_j \in \mathcal{V}_2}} m_{ij} \cdot \text{sim}(v_i, w_j) + \sum_{\substack{\{v_i, v_j\} \in \mathcal{E}_1, \\ \{w_k, w_l\} \in \mathcal{E}_2}} \omega_{ijkl} \cdot m_{ik} \cdot m_{jl} \cdot \text{sim}(\{v_i, v_j\}, \{w_k, w_l\}) \quad (5)$$

The pairwise terms are weighted by the similarity of their nodes associated with their edges:

$$\omega_{ijkl} = \sqrt{\text{sim}(v_i, w_k) \cdot \text{sim}(v_j, w_l)}. \quad (6)$$

Because the weights and the score function  $\text{sim}(\cdot, \cdot)$  are strictly positive the energy is trivially maximized by setting all  $m_{ij}$  to one. We therefore in addition demand that the mapping encoded in  $m$  is injective, i.e., we never map two nodes to a single node. Formally, this means:

$$\forall i \in 1..n_1 : \sum_{j=1}^{n_2} m_{ij} \leq 1 \quad \text{and} \quad \forall j \in 1..n_2 : \sum_{i=1}^{n_1} m_{ij} \leq 1 \quad (7)$$

We also do not want bad matches that connect two pairs of nodes such that the edge between them is incompatible. Assume that node  $v_i \in \mathcal{V}_1$  is mapped to  $w_k \in \mathcal{V}_2$  and  $v_j \in \mathcal{V}_1$  is mapped to  $w_l \in \mathcal{V}_2$ , i.e.,  $m_{ik} m_{jl} = 1$ . Then, the edges  $\{v_i, v_j\}, \{w_k, w_l\}$  must either both exist and be compatible, i.e.:

$$\text{sim}(\{v_i, v_j\}, \{w_k, w_l\}) > 0 \quad (8)$$

or both must not exist. The resulting optimization problem is a standard quadratic assignment problem with compatibility and injectivity constraints. We solve it using a spectral relaxation [Leordeanu and Hebert 2006].

**Additional geometric information:** The model so far does not consider the actual geometry inside a node aside from its symmetry, as encoded in the unary scores. We can optionally add further geometrical descriptors to make the matching more sensitive. In the evaluation of our approach (Section 6), we extended the model by a simple volume ratio of two nodes, providing local cues about the geometry. With  $\text{vol}(v)$  and  $\text{vol}(\mathcal{G})$  denoting the volume of PCA-aligned bounding box for nodes and the whole object, respectively, we create weights of

$$\omega_{ij} = \frac{\min(\text{vol}(v_i)/\text{vol}(\mathcal{G}_1), \text{vol}(v_j)/\text{vol}(\mathcal{G}_2))}{\max(\text{vol}(v_i)/\text{vol}(\mathcal{G}_1), \text{vol}(v_j)/\text{vol}(\mathcal{G}_2))}, \quad (9)$$

and multiply them with the unary scores (in Equation 5:  $\text{sim}(v_i, w_j) = \omega_{ij} \cdot \text{sim}(v_i, w_j)$ ). Depending on the geometric variability of the data, this can both improve or reduce the matching quality (Figure 10). This could be extended with more sophisticated geometry descriptors (which is out of scope of this paper).

## 4.4 Qualitative Geometry Model

The regularity graph constructed above encodes only the topology of the symmetric nodes and qualitative relative orientation information. The strength of this view is its strong invariance with respect to geometric variations. In addition, we can also add geometric information; this is in particular important to study the trade-off between geometry and regularity-based information.

We now describe an extension to introduce more geometry: First, we modify the regularity graph so that it partitions the geometry disjointly. Second, we describe a qualitative model of geometry matching that is compatible with our graph-based approach. Finally, we combine the two scores and compute a solution.

**Modified graph:** The regularity graph defined above consists of overlapping nodes that enumerate maximal symmetry groups. In order to uniquely associate regularity properties with geometry, we segment the geometry further into *disjoint* pieces by splitting at node boundaries: For each point on the geometry, we consider the list of all nodes in the regularity graph that overlap the point. We then segment the geometry into maximal connected fragments that are associated with the same set of graph nodes. In other words, we determine pieces that are overlapped by the same partial symmetries of the shape. This gives a unique, disjoint partition of the original geometry into *segments*, denoted by  $\mathcal{V}' = \{v'_1, \dots, v'_{n'}\}$ .

**Qualitative geometry model:** We include the extrinsic geometry of the shapes. Gelfand et al. [Gelfand et al. 2005] model this

parameter	default value	decription
building regularity graphs		
$\epsilon_{prox}$	5% bb.	max. distance for proximity
$\epsilon_{angle}$	15°	max. angular deviation
similarity		
$\sigma_{rot}$	8.5°	angular kernel width
$\sigma_d$	0.3	lattice kernel width
$\sigma'$	30% bb.	geometry resolution (if used)
graph matching		
$\omega_{prox}, \omega_{over}, \omega_{sub}$	0.15	topology weights (proximity, overlap, subsets)
$\omega_{sym}$	0.55	symmetry weight
symmetry detection		
$\epsilon_{match}$	2.5% bb.	matching threshold

**Table 3:** Parameter overview (bb. = bounding box).

as a graph matching problem: The pairwise Euclidean distances between points of an object are frame independent invariants of the extrinsic geometry. We compute the closest distances between all segments  $v'_i, v'_j \in \mathcal{V}'$  and create pairwise constraints that the Euclidean distances  $\text{dist}(v'_i, v'_j)$  should be preserved by the mapping: Let  $\mathcal{V}'_1, \mathcal{V}'_2$  be the sets of segments of two shapes, and let  $v'_i, v'_j \in \mathcal{V}'_1, w'_k, w'_l \in \mathcal{V}'_2$ . We create the (purely quadratic) energy in terms of variables  $m'_{ij}$  that indicate correspondence between segments (rather than graph nodes):

$$Q_g(m') = \sum_{\substack{v'_i, v'_j \in \mathcal{V}'_1, \\ w'_k, w'_l \in \mathcal{V}'_2}} m'_{ij} \cdot m'_{kl} \cdot \text{sim}'_{ijkl} \quad (10)$$

with

$$\text{sim}'_{ijkl} = \exp\left(-\frac{1}{2\sigma'}(\text{dist}(v'_i, v'_j) - \text{dist}(w'_k, w'_l))^2\right) \quad (11)$$

The Gaussian kernel in Eq. 11 controls the level-of-detail. Because we are only interested in the coarse, low-frequency geometry (expecting details to vary), we use a large variance, effectively blurring the extrinsic geometry. In our experiments, normalize each of the two input shapes separately to unit bounding box and then use  $\sigma' = 0.3$  as standard deviation. This means, any geometry smaller than about 30% of the object size is “blurred out” and does not affect the geometric matching.

**Combining regularity and qualitative geometry:** We combine the two sources of information by linear mixing. First, we setup a quadratic assignment problem on level of segments, as described in the previous paragraph and compute the geometric energy (Eq. 10). We then add the energy of the regularity graph (Eq. 5) by *scattering* it to its corresponding segments. Let  $R(v'_i)$  denote the set of nodes in the regularity graph that overlap segment  $v'_i$ . We can then express the additional energy as:

$$Q_r(m') = \sum_{\substack{v'_i \in \mathcal{V}'_1, \\ w'_j \in \mathcal{V}'_2}} m'_{ij} \left[ \sum_{v \in R(v'_i)} \left( \max_{w \in R(w'_j)} \text{sim}(v, w) \right) \right] \\ + \sum_{\substack{\{v'_i, v'_j\} \in \mathcal{E}_1, \\ \{w'_k, w'_l\} \in \mathcal{E}_2}} m'_{ik} \cdot m'_{jl} \cdot \sum_{\substack{v_1 \in R(v'_i) \\ v_2 \in R(v'_j)}} \left[ \max_{\substack{w_1 \in R(w'_k) \\ w_2 \in R(w'_l)}} (\omega_{ijkl} \cdot \text{sim}(\{v_1, v_2\}, \{w_1, w_2\})) \right] \quad (12)$$

Scattering implements a simple idea: Due to multiple nodes of the regularity graph overlapping each segment, we have to summarize matching scores of  $n$  sources against  $m$  targets. Equation 12 always picks the best target and then averages over all sources. The sum-of-maxima strategy is important. A simple double sum blurs out information because it introduces a quadratic number of mismatches, leading to much worse results in practice. The pairwise case is handled analogously.

The geometry score  $Q_g$  and the regularity score  $Q_r$  are combined

by an affine combination

$$Q(m') = \lambda Q_g(m') + (1 - \lambda) Q_r(m') \quad (13)$$

with a weight parameter  $\lambda \in [0, 1]$  (the value is discussed in Section 6). We then use the same spectral graph matching on the resulting quadratic energy algorithm to compute correspondences.

## 5 Symmetry Detection

Having discussed all the components of our matching model we finally address the algorithm for extracting the symmetry groups  $G \subset E(3)$ , their actions, i.e. transformation  $\mathbf{T}(G)$ , and the corresponding geometry  $\text{geo} \subseteq \mathcal{S}$ . The algorithm is heavily inspired by Pauly et al. [2008] and Bokeloh et al. [2009; 2012]; hence, we focus on the main differences.

**Input:** We assume that our input is a triangle mesh  $\mathcal{S} \subset \mathbb{R}^3$ . Further, let  $l$  denote the longest side length of an axis aligned bounding box of  $\mathcal{S}$ ; the quantity is used to scale thresholds.

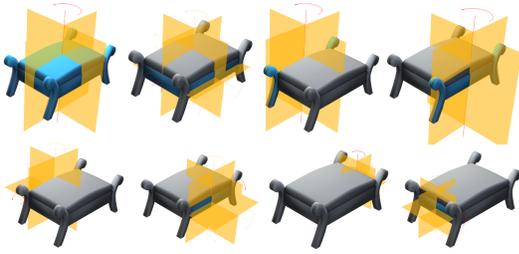
We target meshes that have been modeled by artists. Thus, we expect clean features but regularity might be imperfect because artists often just coarsely place repetitive elements. An extension to scanner data with noisy features requires robust feature detection, which is possible but out of scope of this paper. In order to deal with “artist noise” we incorporate ICP alignment when comparing geometries.

Detecting symmetries in the sense of regularities involves two steps: (i) finding candidate correspondences [Mitra et al. 2006] and (ii) finding *regular geometry*, i.e., grouping transformations to larger sets that form groups acting on a common piece of geometry [Pauly et al. 2008].

**Finding transformations:** First we find sharp edges directly in the mesh, gathering boundaries and non-coplanar edge-adjacent triangles (we use a threshold of  $\leq 150^\circ$ ). A feature point is associated with the center of such an edge, representing the *feature line*. We attach a local frame build from the direction of the edge and the sum of the normals of both adjacent triangles. For point based geometry representation a more sophisticated line detection approach would be required, however, it is out of the scope of this paper. The search of candidate symmetries is accelerated over multiple threads by clustering of feature lines by their length and angle between the normals.

**Detecting 1-parameter groups:** We now detect 1-parameter groups (type I, L, R) via randomized sampling and afterwards combine these, as discussed in Section 3.3. Involutions are given by matching a single pair of features that are related by such a transformation. We avoid spurious matches by requiring at least three more feature matches to confirm the match. For rotations and translations, we compute a continuous orbit: Again, we start with a pair match obtained via random sampling. Let  $\mathbf{x}_1$  be the first feature point, and  $\mathbf{T} \in SE(3)$  be a proper rigid motion (no reflection) that maps the local neighboring feature lines at  $\mathbf{x}_1$  to the second feature point  $\mathbf{x}_2$ . We then consider the continuous orbit  $\mathbf{T}^t(\mathbf{x}_1), t \in \mathbb{R}$ . We find all further feature points that are close to the orbit. To be more robust to modeling inaccuracies (“artist noise”), we still accept matches that are not integer multiples of the generator but could deviate by up to  $\pm 25\%$ . An ICP-based validation, discussed below, removes false positives. Fig. 9 shows an example of automatically detected symmetries.

**Complex groups and scheduling:** We identify the more complex cases by detecting overlaps of one-parameter groups. Scheduling different symmetry detections is important to avoid computational costs: We detect lattices first. When a 1-parameter lattice is detected, we immediately try to extend it to 2- and 3-parameter lattices by searching for candidate correspondences. We use the



**Figure 9:** Symmetry groups found within a bench. The nodes are colored blue. There are only  $C_{2v}$  and  $D_{2h}$  groups present.

RANSAC grid detection algorithm from [Pauly et al. 2008]. From each detected lattice, we keep only one cell for further analysis. The boundaries of the generating cell are uniquely determined by the boundaries of the lattice. Only after handling all lattices, we start detecting rotational 1-parameter groups and involutions.

**Segmentation:** Next we extract the geometry the symmetry group acts upon, i.e.  $\text{geo}(v_i)$ . Here we sample the triangle mesh  $S$  with a Poisson disc sampler, creating a discrete point cloud  $S \subset \mathcal{S}$  with sample spacing  $\epsilon_{\text{sample}} > 0$  (we use  $\epsilon_{\text{sample}} = \frac{1}{200}l$ ; the concrete value has no influence unless it is undersampling the result). Let  $\{\mathbf{I}, \mathbf{T}_1, \dots, \mathbf{T}_k\}$  be an (excerpt of a) group of transformations and  $\mathbf{x}$  the start feature (associated with  $\mathbf{T} = \mathbf{I}$ ; this is only relevant for incomplete excerpts of groups). We start simultaneous region growing in the point cloud starting at  $\mathbf{T}_1\mathbf{x}, \dots, \mathbf{T}_k\mathbf{x}$ , comparing the point-to-plane distance of the input geometry  $S$  and the transformed point clouds  $\mathbf{T}_1(S), \dots, \mathbf{T}_k(S)$  to  $S$ ; region growing is stopped when the distance is larger than a fixed threshold  $\epsilon_{\text{match}} > 0$  (we use  $\epsilon_{\text{match}} = \frac{1}{40}l$  the point-to-plane distance allows us to measure distances below sample spacing accurately). We also stop region growing at boundaries of the connected components of the input mesh. Two more measures make this step more robust: For each continuous orbit  $\mathbf{T}_1^z$ , we try small shifts to the integer exponents  $z$  and use ICP to “snap” in. We accept the match if the matched region becomes larger. Points are considered non-symmetric if they are matched in less than 75% (rather than 100%) of the total number of transformations in the group. All the points of the detected region  $\text{geo} = \mathbf{T}_1\mathbf{x} \cup \dots \cup \mathbf{T}_k\mathbf{x}$  are then marked with the identified symmetry group for complex group combination, see Section 3.3. The exact shape of the segments play a minor role in our model; they affect proximity relations, for volume-estimates (weights in Equation 9) and for removing spurious subgroups Section 3.3.2. The segmentation is also used for rendering matched regions in the visualizations of this paper.

**Avoiding continuous ambiguities:** The segmentation described above (which has been similarly employed in most previous work) often yields unintuitive results due to continuous symmetry. For example, large planar polygons are common and they are often aligned with the transformations of other discrete symmetries. This creates “shadow” boundaries from aligning discrete elements with continuously symmetric area. Although this is formally correct, it is not very useful for our purpose. We therefore only include area that is bounded by symmetric line features as defined above; symmetry ending within a flat region is not allowed.

## 6 Implementation and Results

We have implemented the algorithm in C++ and tested it on a desktop computer equipped with a  $2 \times$  hexa-core Intel Xeon X5650 at 2.6GHz processor and 96GB RAM. The current implementation of the symmetry detection does not implement a few cases of rare symmetry groups: helices and 3-parameter lattices are omitted, and  $S_{2n}$  groups are recognized as  $C_n$ .

We apply our implementation to a number of test data sets, taken from Trimble 3DWarehouse™ and The Digation Archive™. We evaluate two different matching pipelines:

**Pipeline 1:** We match at the level of overlapping graph nodes  $v \in \mathcal{V}$ . We omit the segmentation and the coarse geometry model and use only volume weights.

**Pipeline 2:** We use the qualitative pairwise geometry model of Section 4.4 in favor of the unary volume term. Matching is performed at the level of disjoint segments  $v' \in \mathcal{V}'$ .

The results are shown in Fig. 1 and Figs. 11-18. We show the match of one graph node/segment (blue) to all other node/segments in each other shape (orange), after pairwise injective matching. We use multiple images with selected source nodes, for clarity. A full list of the matching results of all graph nodes (i.e., pipeline 1) is provided as additional material. The corresponding statistics are shown in Table 4. We now first look at results obtained with pipeline 1.

**Wheels:** We first examine a basic case. Fig. 11 shows wheels with 5-, 6- and 12-fold rotational symmetries of various types ( $C_n, C_{nv}, D_n$ ). We already obtain moderately complex graphs (15-28 nodes each, see Table 2), which allow us to disambiguate different components of the wheels.

**Chandeliers:** We add complexity using a collection of five chandeliers with very strong geometric variations. We detect corresponding elements as shown in Fig. 12. The central column of the objects cannot be matched because of the influence of the volume term. In this data set, one input model required manual cleanup: The thin elements in the middle chandelier (Fig. 1) had been placed very asymmetrically by the artist and had to be rearranged.

**Fences:** The fences shown in Fig. 14 utilize the relations between 1-parameter grids and rotational/reflective symmetries. Our symmetry detection algorithm is able to compensate the rather irregular placement (without user intervention) in the original models to some extent and obtain plausible matching results. For the top hand-rail, the algorithm confuses it with the bottom parts; from the perspective of symmetry, these elements are equivalent. Although the fences look like a single grid to casual observer the graph actually contains all node types (lattice, rotations, reflections, dihedral symmetry; leading to graphs with more than 300 edges). Therefore, it is necessary to use a global graph matching model to find the consistent structures that indicate useful matches.

**Cars:** Fig. 18 shows 12 shapes of mixed two-, four- and six-wheel drives representing the most challenging test case that demonstrates the limitations of purely symmetry-based matching. Here, many symmetry relations vary strongly themselves and the topological relations also only give insufficient cues, e.g. symmetric wheels in proximity to symmetric body. While the wheels are mostly matched consistently, other structures are not matched reliably. Fig. 18(left) shows many false-positives while detecting similar windows.

### Influence of Geometry Matching

In the next three examples, we utilize pipeline 2 and study the influence of the geometry term. Fig. 13, 15, and 17 show segment level matches; the results for pipeline 1 are provided in the supplementary material.

**Wind mills:** Fig. 13 shows correspondences among windmills where the proller, balcony and the base part of the first shape were matched properly to other shapes.

**Beds & bicycles:** Fig. 15 and Fig. 17 show objects of average complexity with five objects from the same class and one from a related class (the respective top left images show the disjoint segmentation

data set	$\varnothing \mathcal{V} $	$\min \mathcal{V} $	$\max \mathcal{V} $	$\varnothing \mathcal{E} $	$t_d$	$t_m$
Chandeliers	85	55	168	1369	334	17
Wheels	20	15	28	184	98	0.6
Fences	29	11	47	326	172	1.2
Cars	38	15	89	522	620	16
Beds	22	9	38	273	158	1.6
Bicycles	19	11	32	135	103	0.9
Wind mills	20	4	48	301	105	0.2

**Table 4:**  $\varnothing|\mathcal{V}|$ ,  $\min|\mathcal{V}|$  and  $\max|\mathcal{V}|$  are the average, minimum and maximum number of detected symmetries in a data set;  $\varnothing|\mathcal{E}|$  is the average number of edges in a data set, whereby undirected edges are counted as two;  $t_d$  and  $t_m$  is the average time in seconds for detecting and matching symmetries.

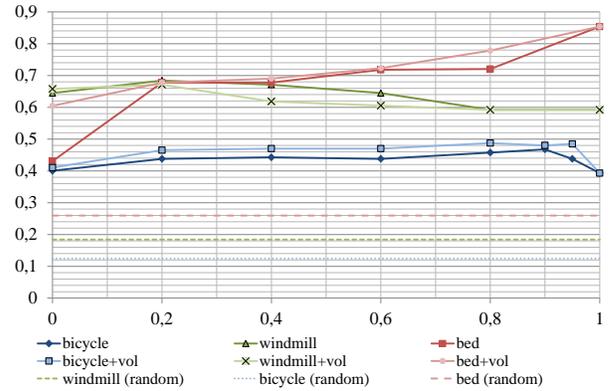
computed by pipeline 2). Front/back parts of the beds were sometimes mismatched as well as left/right, which is the consequence of symmetries in the regularity graph itself. From the perspective of symmetries the found matches are plausible. In case of the two-wheel vehicles the rather unique topology helps to identify corresponding parts, such as frame, sprocket, fork and wheels. In one case, a sprocket matches a wheel; as seen in the segmentation figure, no segments besides of the wheels were detected in the target bicycle. The graph matcher did not have any suitable choice for the correspondence and had to make the only possible assignment.

**Geometric vs. structural symmetry:** We now consider the question of the relative influence of geometry versus symmetry cues. For this, we vary the parameter  $\lambda$  between zero (only symmetry) and one (only geometry). We evaluate the results quantitatively by comparing against user annotated ground-truth for the three data sets (beds, bicycles, windmills). Pairs of segments are tagged as valid correspondences if the assignment seem intuitively plausible. Global symmetries, such as reflections and  $180^\circ$  rotations, are permitted if plausible (beds, windmills). We measure the percentage of correctly matched pairs; the absolute percentages have limited meaning due to the coarse user annotation; however, the relative behavior for each model reveals some interesting properties:

Fig. 10 shows the results. All three data sets benefit from including the coarse geometry model. For bicycles and windmills, a weighted combination gives the best results. For bicycles, higher geometry weights are useful because their coarse-scale geometry varies less. The beds work best with pure geometry matching. The reason for this is that the shapes are quite simple and structurally relevant symmetries are global cuboid symmetries, while symmetry properties of individual elements (bars, handles) vary in an unrelated way. The global symmetries is already captured by the Euclidean distance constraint. The results nonetheless implicitly benefit from symmetry-based segmentation. We also study the influence of local geometry descriptors by reactivating the volume term (lighter-color curves in Fig. 10). This improves the results for shapes with less geometric variability (bikes, beds) but reduces performance for strongly varying node geometry (windmills), as expected.

**Limitations and future work:** The strength of the usage of regularity-based matching is its strong invariance, but this also always implies a trade-off with specificity. Our method captures only coarse correspondences (no dense point-to-point correspondences) and the false-positive always remain. The comparison with geometry-based matching shows that the regularity model provides new, complementary information. Quantitative improvements are in the range of up to 20% (bicycle improve to 0.46 from 0.38; windmills: 0.68 from 0.58); while this does not seem much at first sight, the problem is very difficult and additional information beyond a coarse shape model is hard to obtain.

The a priori model for the matching scores is a compromise for general object classes and it has, despite our efforts to derive it from a few basic principles, still several ad-hoc parameters (see Table 3).



**Figure 10:** Recall plot for the bed, windmill and bicycle datasets. The x-axis represents different weights  $\lambda$  for the qualitative geometry model, and the y-axis is the percentage of correctly predicted annotations. “+vol” (light color) adds a local geometry descriptor. For reference of scale, solutions obtained from random matching scores are given with dotted lines.

In practice, learning these parameters from data might lead to better results than using fixed values. Similarly, the pairwise matching model has still limitations: After quantization, it yields 1:1 matches between graph nodes but cannot recognize symmetries in the solution (such as the individual elements of a 5-fold and 7-fold replication). Spectral embeddings of multiple pairwise matches could address this issue [Lipman et al. 2010].

Further, there are parameter choices for symmetry detection: the choice of spatial and angle tolerance thresholds affects matching results as too generous threshold can lead to clutter. The robustness of symmetry detection (which is not the focus of our paper) is limited, in particular for artist-made, inaccurate meshes. Common artifacts include missing detections and duplicate, non-merged nodes of different symmetry for nearly overlapping geometry.

Overall, our approach should not be regarded as a stand-alone shape matching system but rather as a new type of shape descriptor that yields information complementary to existing work. For generic results, it should be used along with (probably multiple) further geometric descriptors, probably with a discriminative learning method for optimizing the fusion of information. However, such a system is beyond the scope of our paper.

## 7 Conclusion

We have presented a new method for finding correspondences between shapes by relating their symmetries. The method uses symmetry groups to identify canonical building blocks of geometry and relates these parts by comparing the overlap in their regularity structure as well as the relation of symmetry axes and planes, as well as translational generators. The method allows matching of man-made shapes that are related in function but might have very different geometry. The method is complementary to existing geometric approaches: While it gives only coarse and partial matches, it can perform matching between very dissimilar geometries, where no traditional methods are available.

In future work, in addition to addressing limitations discussed above, we would like to extend the idea towards shape classification as well as generation of shapes by incorporating symmetry regularities. A key tool could be the estimation of a latent regularity graph of symmetry properties common to a larger collection of objects. We think such a data-driven approach could address an important aspect of characterizing the structure of shapes.

## Acknowledgements

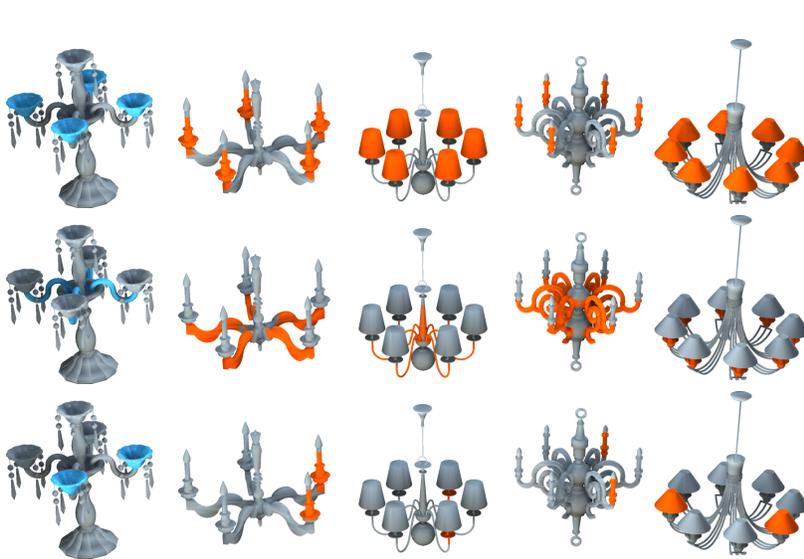
The authors would like to thank Martin Bokeloh, Bernt Schiele, and Sharath Raghvendra for insightful discussions, and the anonymous reviewers for their helpful comments and suggestions. This work has been supported by NSF grants FODAVA 808515 and DMS 1228304, a Google research award, the Max Plack Center for Visual Computing and Communications, and the Cluster of Excellence “Multimodal Computing and Interaction”. The 3D models in our experiments were taken from The Digimation™ Model Bank Library and Trimble 3D Warehouse™.

## References

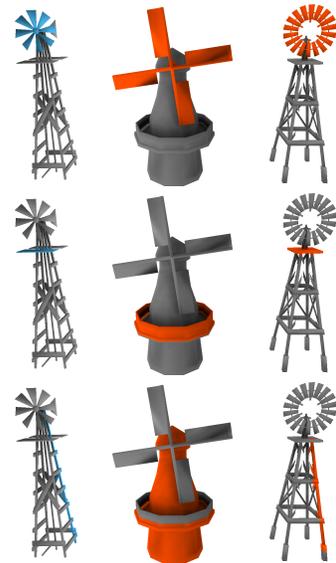
- ALLEN, B., CURLESS, B., AND POPOVIĆ, Z. 2003. The space of human body shapes: Reconstruction and parameterization from range scans. *ACM Trans. Graph.* 22, 3, 587–594.
- BOKELOH, M., BERNER, A., WAND, M., SEIDEL, H.-P., AND SCHILLING, A. 2009. Symmetry detection using line features. *Computer Graphics Forum (Proc. Eurographics)*.
- BOKELOH, M., WAND, M., SEIDEL, H.-P., AND KOLTUN, V. 2012. An algebraic model for parameterized shape editing. *ACM Transactions on Graphics* 31, 4.
- CULLEN, B., AND O’SULLIVAN, C. 2011. Symmetry hybrids. In *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, ACM, New York, NY, USA, CAe ’11, 33–38.
- FISHER, M., SAVVA, M., AND HANRAHAN, P. 2011. Characterizing structural relationships in scenes using graph kernels. *ACM Trans. Graph.* 30, 4 (July), 34:1–34:12.
- GAL, R., SORKINE, O., MITRA, N., AND COHEN-OR, D. 2009. iwires: An analyze-and-edit approach to shape manipulation. *ACM Trans. Graph.* 28, 3.
- GELFAND, N., AND GUIBAS, L. J. 2004. Shape segmentation using local slippage analysis. In *Proc. SGP*, 214–223.
- GELFAND, N., MITRA, N. J., GUIBAS, L. J., AND POTTMANN, H. 2005. Robust global registration. In *Proc. SGP*, 197–206.
- HAHN, T. 2002. *International Tables for Crystallography, Volume A: Space Group Symmetry*. Springer Verlag, Berlin.
- HAUAGGE, D. C., AND SNAVELY, N. 2012. Image matching using local symmetry features. In *Proc. CVPR*, 206–213.
- HENDERSON, T. C., COHEN, E., JOSHI, A., GRANT, E., DRAELOS, M., AND DESHPANDE, N. 2012. Symmetry as a basis for perceptual fusion. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012 IEEE Conference on*, 101–107.
- HUANG, Q., ZHANG, G., GAO, L., HU, S., BUSTCHER, A., AND GUIBAS, L. 2012. An optimization approach for extracting and encoding consistent maps in a shape collection. *ACM Transactions on Graphics* 31, 125:1–125:11.
- KALOGERAKIS, E., HERTZMANN, A., AND SINGH, K. 2010. Learning 3d mesh segmentation and labeling. *ACM Trans. Graph.* 29, 3.
- KAZHDAN, M., FUNKHOUSER, T., AND RUSINKIEWICZ, S. 2003. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Symposium on Geometry Processing*.
- KAZHDAN, M., FUNKHOUSER, T., AND RUSINKIEWICZ, S. 2004. Symmetry descriptors and 3d shape matching. In *Proc. Symposium on Geometry Processing (SGP)*.
- KIM, V. G., LI, W., MITRA, N., DIVERDI, S., AND FUNKHOUSER, T. 2012. Exploring collections of 3d models using fuzzy correspondences. In *ACM SIGGRAPH 2012 papers*.
- LEORDEANU, M., AND HEBERT, M. 2006. Efficient map approximation for dense energy functions. *ICML ’06*, 545–552.
- LIPMAN, Y., CHEN, X., DAUBECHIES, I., AND FUNKHOUSER, T. 2010. Symmetry factored embedding and distance. *ACM Trans. Graph.* 29 (July), 103:1–103:12.
- LIU, T., KIM, V. G., AND FUNKHOUSER, T. 2012. Finding surface correspondences using symmetry axis curves. *Computer Graphics Forum (Proc. SGP)* (July).
- MARTINET, A. 2007. *Structuring 3D Geometry based on Symmetry and Instancing Information*. PhD thesis, INP Grenoble.
- MITRA, N. J., GUIBAS, L. J., AND PAULY, M. 2006. Partial and approximate symmetry detection for 3d geometry. *ACM Trans. Graph.* 25, 3, 560–568.
- MITRA, N. J., YANG, Y.-L., YAN, D.-M., LI, W., AND AGRAWALA, M. 2010. Illustrating how mechanical assemblies work. *ACM Transactions on Graphics* 29, 3.
- NGUYEN, A., BEN-CHEN, M., WELNICKA, K., YE, Y., AND GUIBAS, L. 2011. An optimization approach to improving collections of shape maps. *Computer Graphics Forum (Proc. SGP)*, 1481–1491.
- OVSIJANIKOV, M., LI, W., GUIBAS, L., AND MITRA, N. 2011. Exploration of continuous variability in collections of 3d shapes. *ACM Trans. Graph. (Proc. Siggraph)*.
- PAULY, M., MITRA, N. J., WALLNER, J., POTTMANN, H., AND GUIBAS, L. 2008. Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics* 27, 3, #43, 1–11.
- SIDI, O., VAN KAICK, O., KLEIMAN, Y., ZHANG, H., AND COHEN-OR, D. 2011. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *ACM Trans. Graph.* 30, 6 (Dec.), 126:1–126:10.
- SIMARI, P., NOWROUZEZAHRAI, D., KALOGERAKIS, E., AND SINGH, K. 2009. Multi-objective shape segmentation and labeling. *Computer Graphics Forum* 28, 5.
- THRUN, S., AND WEGBREIT, B. 2005. Shape from symmetry. In *Proc. ICCV’05*, 1824–1831.
- VAN KAICK, O., ZHANG, H., HAMARNEH, G., AND COHEN-OR, D. 2011. A survey on shape correspondence. *Computer Graphics Forum* 30, 6, 1681–1707.
- VAN KAICK, O., XU, K., ZHANG, H., WANG, Y., SUN, S., SHAMIR, A., AND COHEN-OR, D. 2013. Co-hierarchical analysis of shape structures. *ACM Trans. Graph. (Proc. Siggraph)* 32, 4 (July), 69:1–69:10.
- WANG, Y., XU, K., LI, J., ZHANG, H., SHAMIR, A., LIU, L., CHENG, Z., AND XIONG, Y. 2011. Symmetry hierarchy of man-made objects. *Computer Graphics Forum (Proc. EUROGRAPHICS)* 30, 2.
- WANG, Y., XU, K., ZHANG, H., COHEN-OR, D., AND SHAMIR, A. 2012. Structural co-hierarchy of a set of shapes. Tech. rep.
- XU, K., ZHANG, H., TAGLIASACCHI, A., LIU, L., LI, G., MENG, M., AND XIONG, Y. 2009. Partial intrinsic reflectional symmetry of 3d shapes. *ACM Transactions on Graphics, (Proceedings SIGGRAPH Asia 2009)* 28, 5, 138:1–138:10.



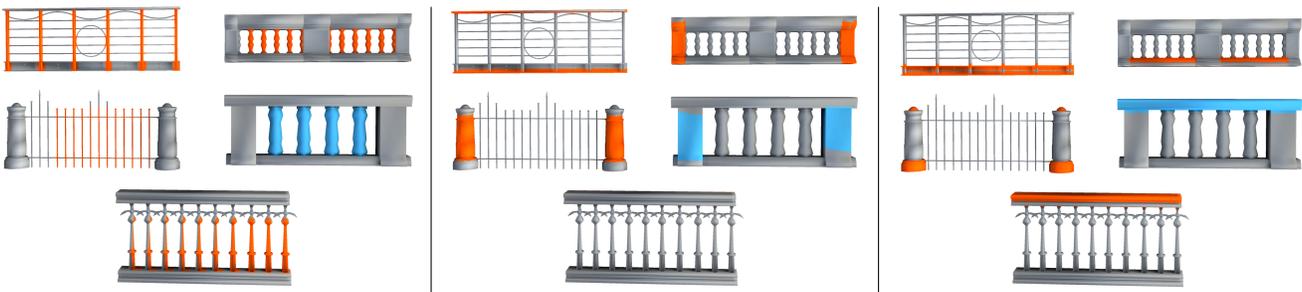
**Figure 11:** Results for the “wheels”. We show the matches for several nodes; the selected node is marked blue, the corresponding nodes are marked orange. The examples includes various rotational symmetries as well as dihedral groups. The lower right example shows a mismatch, where the spokes forming 12-fold rotational symmetry matches the outer tire best.



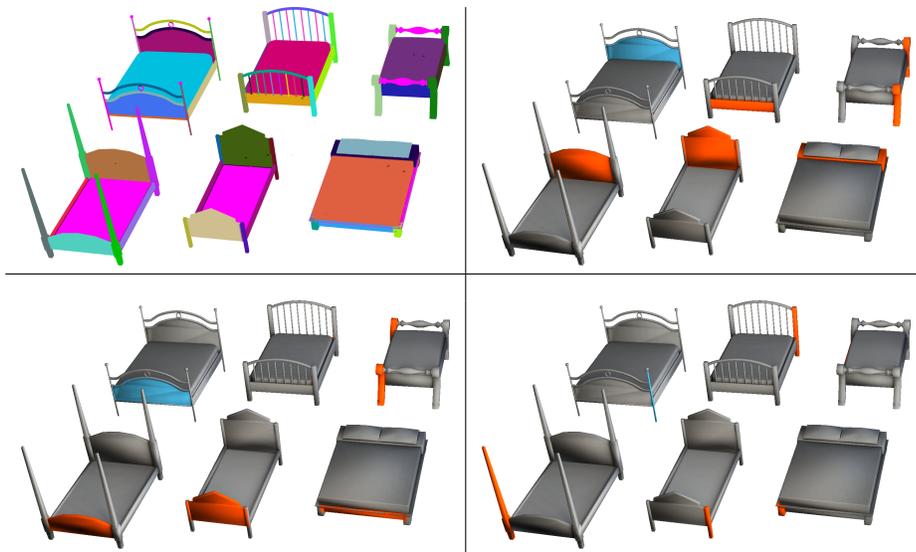
**Figure 12:** Upper row: Selecting a single lamp (blue) plausibly identifies most related parts (orange) on other shapes. Center: The support structure was identified. Bottom: Our graph contains further nodes; we show as an example a node capturing a  $C_{2v}$  involution between a pair of lamps. Given the differences in rotations, we obtain a plausible assignment.



**Figure 13:** Windmills: The main structural elements as well as propellers are identified properly.



**Figure 14:** The fences models suffer from imprecise modeling (there is remaining shift in the lower fence on the right). The matches are qualitatively correct, except from the handrail, which appears fully symmetric to the bottom rail to our algorithm; this symmetry is broken by the pairwise matching.



**Figure 15:** “Bed” data set with 5 bed and one bench shape. Matching cannot determine the global orientation information, hence, front/back or left/right parts are frequently swapped by a global reflection or 180° rotation. Selected node is shown in blue. This example uses disjoint segments and the qualitative geometry model (pipeline 2). The top-left image shows the segments computed within pipeline 2 (Sect. 4.4). Purple parts have not been identified as symmetric.



**Figure 16:** “Church” data set. Lattice structures, e.g. windows, as well as the main towers were matched properly.



**Figure 17:** “Bicycle” data set: top-right image shows matching of the sprocket, the bottom-left of the fork, bottom-left of the frame and bottom-right of the frame. This example uses disjoint segments and the qualitative geometry model (pipeline 2). The top-left image shows the disjoint segments computed within pipeline 2 (Sect. 4.4). Purple parts have not been identified as symmetric.



**Figure 18:** “Cars” are a borderline case. While wheels are matched properly, windows, for example, suffer from various false-positives.