Detecting Reflectional Symmetries in 3D Data Through Symmetrical Fitting

Aleksandrs Ecins
aecins@umiacs.umd.edu

Cornelia Fermüller
fer@umiacs.umd.edu

Yiannis Aloimonos
yiannis@umiacs.umd.edu

Department of Computer Science, University of Maryland, College Park

Abstract

Symmetry is ubiquitous in both natural and man-made environments. It reveals redundancies in the structure of the world around us and thus can be used in a variety of visual processing tasks. This paper presents a simple and robust approach to detecting symmetric objects and extracting their symmetries from three-dimensional data. Given a 3D mesh of an object, a set of candidate symmetries are proposed first and are then refined, so that they reflect the complete mesh onto itself. We show how our method can be used to detect symmetric objects in scenes consisting of synthetic 3D models, as well as 3D scans of real environments.

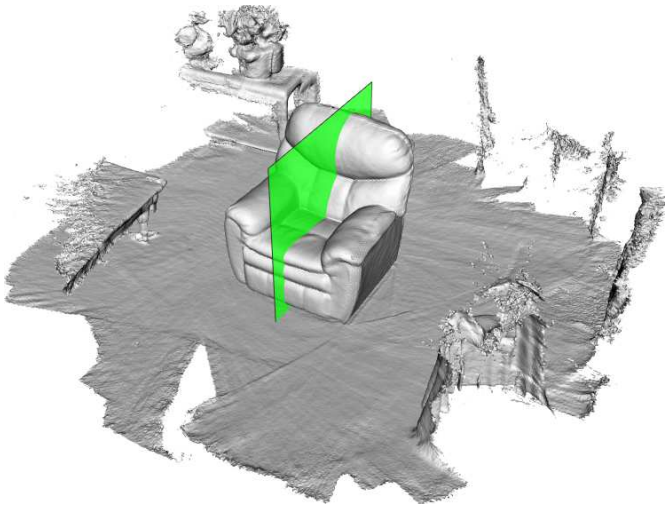


Figure 1: Output of our approach on a 3D reconstruction of a real scene.

1. Introduction

Symmetry is a common property shared by the majority of both natural and man-made objects and structures. Psychological studies suggest that symmetry plays a crucial role in the human visual perception system [13, 4]. In Computer Vision, symmetry detection was shown to serve as a useful preprocessing step for a variety of object-centric tasks such as object recognition [2], retrieval [8], 6DOF tracking [7], reconstruction [11], as well as robotic applications such as grasping [9].

Reflectional symmetry detection in 3D data has been studied extensively in the fields of Computer Vision, Robotics and Graphics [6]. Most of the proposed methods rely on the idea of finding symmetric correspondences between oriented points. Podolak *et al*. [8] defined the "Planar Reflective Symmetry Transform" that captured the degree of symmetry of an object with respect to all possible planes passing through it. Exact object symmetries were then detected by extracting the transform maxima. In [5], Mitra *et al*. relied on orientable feature matching to detect partial symmetries in complete 3D meshes of single objects. Symmetry hypotheses were found by matching uniquely orientable keypoints on the surface of the mesh and filtering out the dominant ones using mean shift clustering. Thrun and Wegbeit reasoned about occlusions of the scene to detect symmetries in partial 3D pointclouds [12]. Symmetries were found by searching through the space of possible transformations and finding the one that minimizes the number of points that reflect into the unoccluded space and maximizes the match between the original and the reflected clouds.

Unlike the majority of previous methods that use sparse feature matching, our approach uses dense symmetric correspondences to fit a reflectional symmetry plane to the surface of an object. We use an ICP-like technique that alternates between finding correspondences between symmetric points and refining the candidate symmetry plane given the correspondences. To ensure convergence we initialize multiple candidate symmetries based on the principle axes of the object. Candidate symmetries that did not converge to the true symmetries of an object are discarded using a set of metrics that measure how well a symmetry fits a given object. Finally, to find symmetric objects in a scene, we first segment out indivudial objects and then apply our fitting approach on each of the segments. Objects are segmented

(a) (b)

Figure 2: Visualization of the reflectional symmetry fitting approach (a) Input object pointcloud and candidate symmetry. Black curve denotes the observed points of the object. (b) Grey curve denotes the reflected pointcloud and blue lines show the estimated symmetric correspondences.

by extracting connected components lying above a common support plane (such as a table). While this object segmentation strategy is adequate for simple scenes where objects stand apart from each other, it would fail for heavily cluttered scenes where multiple objects are in physical contact. A more sophisticated object segmentation approach is required to deal with such cases [3].

The rest of the paper is organized as follows: Section 2 explains the process of fitting a symmetry plane to a pointcloud of an object. Section 3 describes the filtering metrics used to measure the fitness error between a symmetry plane and an object. Qualitative evaluation of our approach on synthetic and real scenes is presented section 4.

2. Symmetry fitting

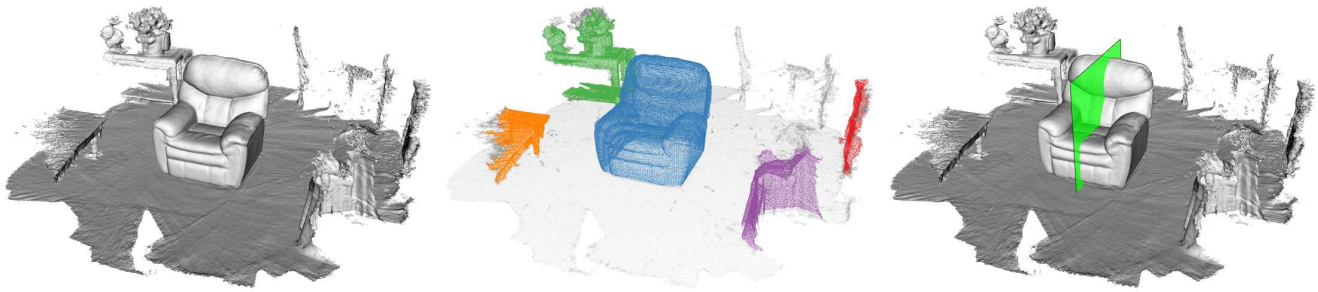
In order to detect scene symmetries, we employ a geometric fitting approach. Consider a pointcloud of an object consisting of oriented points $P_i = \{p_i, n_i\}$, where p_i is the the position of a point and n_i is its normal. A reflectional symmetry S imposes a constraint on the position and orientation of pairs of "symmetric points". We say that two points P_i and P_j form a symmetric correspondence under the reflectional symmetry S if the two points reflect onto each other i.e. P_j is close to $S(P_i)$ - the reflection of P_i by S. This motivates our fitting approach that alternates between symmetric correspondence estimation and symmetry plane refinement.

We represent a symmetry plane $S = \{p^S, n^S\}$ by a point lying in the plane p^S and the plane's normal n^S. For each point P_i of an object pointcloud we compute its reflection $S(P_i)$ and search for its nearest neighbor P_j. Since the object pointcloud may not be perfectly symmetric, and our initial symmetry plane may not be correct, we do not expect every point to have a valid symmetric match. Thus we only establish a correspondence if the distance between the reflected point and its nearest neighbor is less than a given distance threshold. In our experiments we calculate this threshold as twice the estimated spatial resolution of the pointcloud. Additionally, we reject all correspondences for which the angle between the reflected source normal $S(n_i)$ and the corresponding normal n_j is greater than a threshold (we used 45° in our experiments). After establishing the correspondences, we minimize the following function for the parameters of the symmetry plane:

$$\sum_{\{i,j\}} d_{p,pl}(S(P_i), P_j) \tag{1}$$

where $d_{p,pl}$ stands for point to plane distance and $\{i, j\}$ is the set of point correspondences. Levenberg-Marquardt non-linear solver is used to perform the optimization. This process is repeated until convergence or until a maximum number of 20 iterations is reached. This procedure results in a very accurate alignment between a symmetry and a pointcloud. However, similar to ICP, it only converges to a valid symmetry if the initial symmetry plane estimate is sufficiently close to the correct solution [10].

One of the ways to generate initial symmetry candidates is by jointly sampling plane positions and orientations. This is computationally prohibitive, since sampling from an $n \times n \times n$ grid is $\mathcal{O}(n^3)$ and densely sampling from a unit sphere incurs a large constant factor [8]. We avoid the costly position sampling by only sampling orientations and then refining plane positions along their normals based on the pointcloud structure. We start by sampling points from the surface of a unit sphere aligned to the principal axes of the object pointcloud. Specifically, we uniformly sample symmetry plane orientations in azimuth and elevation angles using a step size of of 45°. For each orientation, we construct a symmetry plane that passes through the segment's center of mass. To refine the position of a plane, we first find all points that could potentially form symmetric correspondences if the symmetry plane was shifted appropriately. Given a point P_i, its potential symmetric neighbors



(a) (b) (c)

Figure 3: Symmetry detection pipeline. Note that although several potential object segments were extracted, our approach returns the symmetry only for the symmetric object(s). (a) Input mesh. (b) Extracted object segments. (c) Detected symmetries.

P_j are enclosed in a cylinder centered at P_i with an axis parallel to the plane normal n^S. These "cylindrical" neighbors can be found efficiently by projecting the pointcloud onto the symmetry plane and performing a radius search. We filter out correspondences for which the angle between the reflected source normal $S(n_i)$ and the neighbor normal n_j is greater than $10°$. For each of the remaining correspondences, we calculate the shift distance $d_{i,j}$ as the distance between the symmetry plane and the midpoint between P_i and P_j. We compute the median shift d_{median} from all correspondences and use it to shift the symmetry candidate's point along its normal:

$$S = \{p^S + d_{median} * n^S, n^S\} \tag{2}$$

After generating the initial symmetries, we apply the global iterative refinement described above to fit the candidate symmetries to the pointcloud. Since some of the candidates are expected to converge to the same symmetry, we merge similar refined symmetry candidates. Note that this approach allows extracting multiple symmetries for the same object, since different initial symmetries may converge to different true symmetries of the object.

3. Symmetry filtering

The output of the fitting process described above is a set of candidate symmetries aligned to the object pointcloud. While some of the candidates will converge to the true symmetries of the object, some will converge to local minima that do not correspond to the global object symmetries. To filter out these incorrect detections we define two measures of fitness of a reflectional symmetry plane to a pointcloud.

Inlier score. Inlier score $Inlier(S)$ measures the proportion of the points in the pointcloud that have symmetric correspondences. It is calculated as the number of points with symmetric correspondences divided by the total number of points in the pointcloud.

Fitness score. Symmetry score $Fit(S)$ measures the average quality of the symmetric correspondences of a pointcloud. The quality of the symmetric correspondence is defined as the angle between the reflected source normal and the corresponding normal $\angle(S(n_i), n_j)$. Fitness score is calculated as:

$$Fit(S) = \frac{1}{|Crsp_S|} \sum_{\{P_i,P_j\} \in Crsp_S} 1 - \frac{\angle(S(n_i), n_j)}{\pi} \tag{3}$$

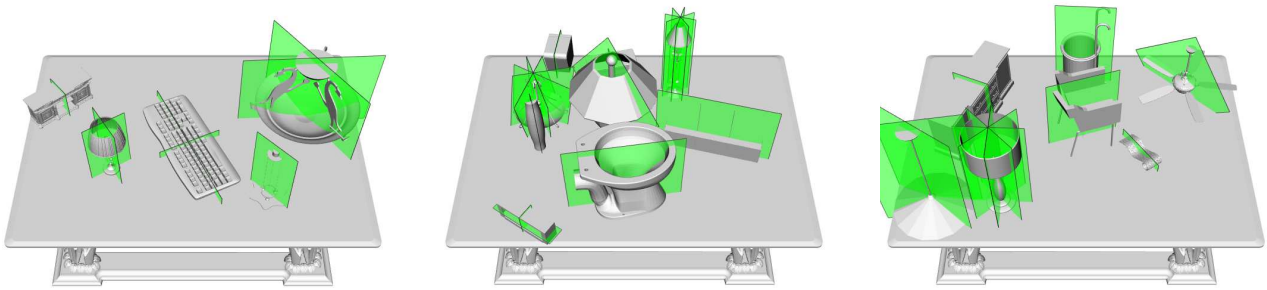
where $|Crsp_S|$ is the set of symmetric correspondences under symmetry S.

Both scores are defined in the $[0, 1]$ range, with higher values indicating a better fit to the pointcloud. A symmetry hypothesis S is considered valid if it satisfies both measures i.e. $Inlier(S) > \tau_{inlier}$ and $Fit(S) > \tau_{fit}$. Varying the filtering thresholds allows us to control the tolerance of imperfect symmetries, as well as sensitivity to noise.

4. Evaluation

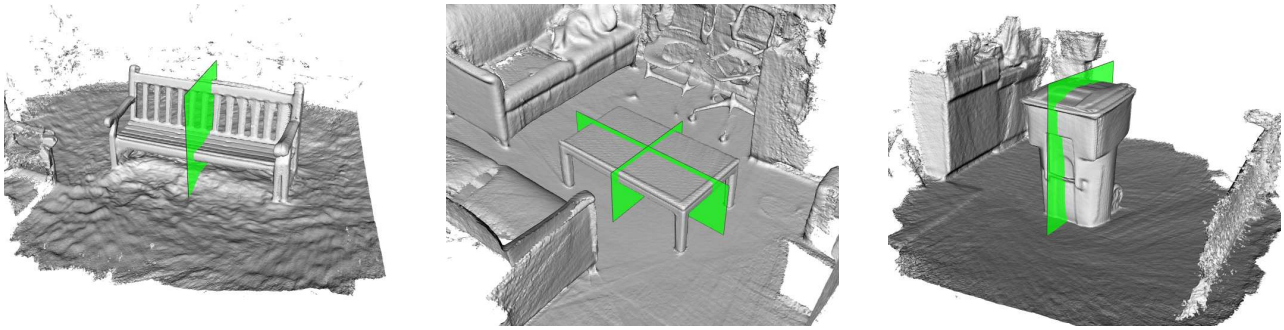
We evaluate our method on the 3D datasets released for the ICCV 2017 Symmetry Workshop Challenge [1]. Here we show results for local symmetry detection on synthetic and real scenes. Synthetic scenes are constructed by placing multiple 3D object models on a table, while real scenes are 3D room reconstructions. In both cases the first processing step is to detect the dominant plane in the scene (table plane for synthetic scenes and floor plane for real scenes). Individual object pointclouds are then obtained by extracting the connected components of the mesh lying above the support plane. Finally symmetries are fitted to the extracted object segments. The steps of the pipeline are shown in Figure 3.

For synthetic scenes we use agressive filtering thresholds of $\tau_{inlier} = 0.99$ and $\tau_{fit} = 0.95$. These are motivated by the fact that 3D models are noise-free, hence all of points in



(a) (b) (c)

Figure 4: Symmetry detection results on the synthetic dataset.



(a) (b) (c)

Figure 5: Symmetry detection results on the real dataset.

the model should have a symmetric correspondence (except for a few points in objects that are not perfectly symmetric). For the real scenes we use a looser filtering criteria of $\tau_{inlier} = 0.90$ and $\tau_{fit} = 0.90$. This is motivated by the fact that real data is noisy and may not capture the complete object's shape.

Figures 4 and 5 show detection results for sample scenes in synthetic and real datasets. Our method correctly predicts the symmetries for most of the objects. Note that it returns very few false positives. This demonstrates that our filtering stage is very efficient at estimating the consistency of a symmetry with a given object. For objects containing multiple symmetries, our method may detect some but not all of the symmetries. For example in Figure 4c, the fan has 5 correct symmetries (each passing through one of the blades), while our method only returns a single one. This is due to the fact that our initialization procedure produced only one symmetry that was close enough to the true symmetry. This problem can be alleviated by sampling the direction sphere more densely during the initialization stage. However, this comes at the cost of increased runtime. Another case of failure occurs when the distribution of points in the object is not symmetric. This happens for objects that are not perfectly symmetric, or have an additional "internal" structure that does not have the same symmetries as the external shape of the object e.g. a cupboard with compartments inside. While our approach can tolerate small deviations, larger ones tend to drive the alignment process away from the true symmetries of the object.

5. Conclusions

In this paper we presented a novel method for detecting reflectional symmetries in 3D data. The combination of a robust symmetry fitting approach with a precise filtering stage allows our method to return accurate symmetries for objects in both synthetic and real scenes. A promising direction of future research is to apply our approach on more cluttered scenes, where objects are piled on top of each other and are not fully visible.

References

[1] Iccv'17 symmetry workshop. https://sites.google.com/view/symcomp17/home, (accessed July 23, 2017). 3

[2] E. Barnea and O. Ben-Shahar. Depth based object detection from partial pose estimation of symmetric objects. In *European Conference on Computer Vision*, pages 377–390. Springer, 2014. 1

[3] A. Ecins, C. Fermüller, and Y. Aloimonos. Cluttered scene segmentation using the symmetry constraint. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 2271–2278. IEEE, 2016. 2

[4] P. Locher and C. Nodine. The perceptual value of symmetry. *Computers & Mathematics with Applications*, 17(4):475–484, 1989. 1

[5] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (TOG)*, 25(3):560–568, 2006. 1

[6] N. J. Mitra, M. Pauly, M. Wand, and D. Ceylan. Symmetry in 3d geometry: Extraction and applications. In *EUROGRAPHICS State-of-the-art Report*, 2012. 1

[7] K. Pauwels and D. Kragic. Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 1300–1307. IEEE, 2015. 1

[8] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser. A planar-reflective symmetry transform for 3d shapes. *ACM Transactions on Graphics (TOG)*, 25(3):549–559, 2006. 1, 2

[9] A. H. Quispe, B. Milville, M. A. Gutiérrez, C. Erdogan, M. Stilman, H. Christensen, and H. B. Amor. Exploiting symmetries and extrusions for grasping household objects. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 3702–3708. IEEE, 2015. 1

[10] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001. 2

[11] P. Speciale, M. R. Oswald, A. Cohen, and M. Pollefeys. A symmetry prior for convex variational 3d reconstruction. In *European Conference on Computer Vision*, pages 313–328. Springer, 2016. 1

[12] S. Thrun and B. Wegbreit. Shape from symmetry. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1824–1831. IEEE, 2005. 1

[13] C. W. Tyler. *Human symmetry perception and its computational analysis*. Psychology Press, 2003. 1