Efficient 3D Reflection Symmetry Detection: a View-Based Approach

Bo Li[a,*], Henry Johan[b], Yuxiang Ye[a], Yijuan Lu[a]

[a] *Department of Computer Science, Texas State University, San Marcos, USA*
[b] *Visual Computing, Fraunhofer IDM@NTU, Singapore*

Abstract

Symmetries exist in many 3D models while efficiently finding their symmetry planes is important and useful for many related applications. This paper presents a simple and efficient view-based reflection symmetry detection method based on the viewpoint entropy features of a set of sample views of a 3D model. Before symmetry detection, we align the 3D model based on the Continuous Principal Component Analysis (CPCA) method. To avoid the high computational load resulting from a directly combinatorial matching among the sample views, we develop a fast symmetry plane detection method by first generating a candidate symmetry plane based on a matching pair of sample views and then verifying whether the number of remaining matching pairs is within a minimum number. Experimental results and two related applications demonstrate better accuracy, efficiency, robustness and versatility of our algorithm than state-of-the-art approaches.

Keywords:
symmetry detection, reflection symmetry, view-based approach, viewpoint entropy, matching

1. Introduction

Symmetry is an important clue for geometry perception: it is not only in many man-made models, but also widely exists in the nature [1]. Symmetry has been used in many applications such as: 3D alignment [2], shape matching [3], remeshing [4], 3D model segmentation [5] and retrieval [6].

However, existing symmetry detection algorithms still have much room for improvement in terms of both simplicity and efficiency in detecting symmetry planes, as well as the degree of freedom to find approximate symmetry planes for a roughly symmetric 3D model. In addition, most of the existing symmetry detection methods are geometry-based, thus their computational efficiency will be tremendously influenced by the number of vertices of a model.Though sampling and simplification can be used to reduce the number of vertices, they also decrease the shape accuracy and cause deviations in geometry. Therefore, a symmetry detection algorithm often directly uses original models as its input, as can be found in many existing related papers.

Motivated by the symmetric patterns existing in the viewpoint entropy [7] distribution of a symmetric model, we propose a novel and efficient view-based symmetry detection algorithm (see Fig. 1) which finds symmetry plane(s) by matching the viewpoint entropy features of a set of sample views of a 3D model aligned beforehand using Continuous Principal Component Analysis (CPCA) [8]. Based on experimental results, we find that our symmetry detection algorithm is more accurate (in terms of both the positions of detected symmetry planes and sensitivity to minor symmetry differences), efficient, robust (e.g. to the number of vertices and parameter settings such as view sampling), and versatile in finding symmetry planes of diverse models.

In the rest of the paper, we first review the related work in Section 2. In Section 3, we present the viewpoint entropy distribution-based symmetry detection algorithm. Section 4 describes diverse experimental evaluation and comparison results of the detection algorithm. In Section 5, we show two interesting applications of our symmetry detection idea in 3D model alignment and best view selection. Section 6 concludes the paper and lists several future research directions. This paper is an extension of our prior publication [9].

2. Related Work

Symmetry Types. Though there are different types of symmetry, reflection symmetry is the most important and commonly studied. Chaouch and Verroust-Blondet [2] introduced four types of reflection symmetries, which are cyclic (several mirror planes passing through a fixed axis), dihedral (several mirror planes passing through a fixed axis with one perpendicular to the axis), rotational symmetry (looks similar after rotation, e.g., different platonic solids, like tetrahedron, octahedron, icosahedron and dodecahedron) and unique symmetry (only one mirror plane, for instance, many natural and most man-made objects). Most symmetric objects are mirror rather than rotational symmetric [10].

Symmetry Detection. Symmetry detection is to search the (partial or full) symmetry planes of a 3D object. The latest review on symmetry detection is available in [11]. We classify current

*Corresponding author at: 601 University Drive, Department of Computer Science, Texas State University, San Marcos, Texas 78666; E-mail: B_L58@txstate.edu, li.bo.ntu0@gmail.com; Tel: +001 512 245 6580; Fax: +001 512 245 8750.

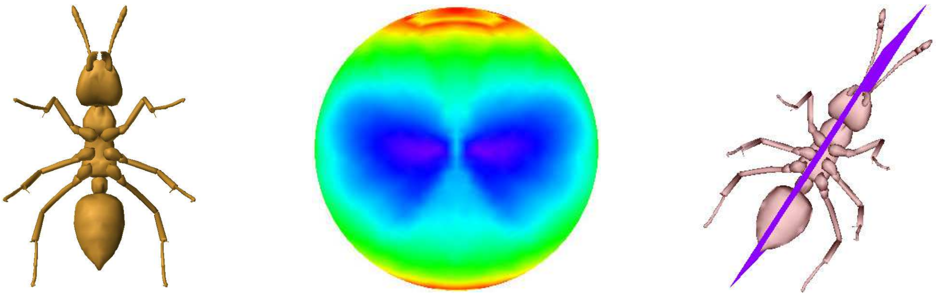


Figure 1: An overview of our view-based symmetry detection algorithm: an example of an ant model, its viewpoint entropy distribution, and the detected symmetry plane by matching the viewpoints.

symmetry detection techniques into the following four groups according to the features employed.

Symmetry detection based on pairing point features. This type of approach first samples points on the surface of a 3D model and then extracts their features. After that, it finds point pairs by matching the points. Based on the point pairs, symmetry evidences are accumulated to decide the symmetry plane. Two typical algorithms are [12] and [13]. To decide the symmetry plane, Mitra et al. [12] adopted a stochastic clustering and region-growing approach, while Calliere et al. [13] followed the same framework of pairing and clustering, but utilized 3D Hough transform to extract significant symmetries. In fact, the initial idea of this approach can be traced back to the symmetry distance defined in [14]. Podolak et al. [15] proposed a planar-reflective symmetry transform and based on the transform they defined two 3D features named center of symmetry and principal symmetry axes, which are useful for related applications such as 3D model alignment, segmentation, and viewpoint selection.

Symmetry detection based on pairing line features. Bokeloh et al. [16] targeted on the so-called rigid symmetries by matching feature lines. Rigid symmetries are the reoccurring components with differences only in rigid transformations (translation, rotation and mirror). They first extracted feature lines of a 3D model, then performed feature line matching, and finally validated the symmetry based on the feature correspondence information by adopting a region growing approach, as well.

Symmetry detection based on 2D image features. Sawada and Pizlo [10] [17] performed symmetry detection based on a single 2D image of a volumetric shape. First, a polyhedron is recovered from the single 2D image based on a set of constraints including 3D shape symmetry, minimum surface area, maximum 3D compactness and maximum planarity of contours. Then, they directly compared the two halves of the polyhedron to decide its symmetry degree. From a psychological perspective, Zou and Lee [18] [19] proposed one method to detect the skewed rotational and mirror symmetry respectively from a CAD line drawing based on a topological analysis of the edge connections.

Other symmetry detection approaches. Martinet et al. [20] proposed a 3D feature named generalized moments for symmetry detection. Rather than directly computing original moments features, they mapped them into another feature space by spherical harmonics transform and then searched for the global symmetry in the new feature space. Xu et al. [21] developed an algorithm to detect partial intrinsic reflectional symmetry based on an intrinsic reflectional symmetry axis transform. After that, a multi-scale partial intrinsic symmetry detection algorithm was proposed in [22]. There are also techniques to detect some other specific symmetries, such as curved symmetry [23] and symmetries of non-rigid models [24] [25], as well as symmetry hierarchy of a man-made 3D model [26]. Kim et al. [27] detected global intrinsic symmetries of a 3D model based on Möbius Transformations [28], a stereographic projection approach in geometry. Recently, Wang et al. [29] proposed Spectral Global Intrinsic Symmetry Invariant Functions (GISIFs), which are robust to local topological changes compared to the GISIFs obtained from geodesic distances. Their generality and flexibility outperform the two classical GISIFs: Heat Kernel Signature (HKS) [30] and Wave Kernel Signature (WKS) [31].

All above and existing symmetry detection techniques can be categorized into geometry-based approach. However, distinctively different from them, we adopt a view-based approach to accumulate the geometrical information of many vertices together into a view in order to more efficiently detect the reflection symmetry of a 3D model, which also serves as the novelty and main contribution of our method.

Symmetry Applications. As an important shape feature, symmetry is useful for many related applications. For example, they include symmetry plane detection for 3D MRI image [32], shape matching [3] [15], 3D model alignment [33] [6], shape processing and analysis [34] including remeshing [4], symmetrization [12], viewpoint selection [15], and subspace shape analysis [35], 3D segmentation [15] [5] [29], and curve skeleton extraction [36] [37].

3. Symmetry Detection Algorithm

3.1. 3D Model Normalization Based on CPCA

Properly normalizing a 3D model before symmetry detection can help us to minimize the searching space for symmetry planes to be some 2D planes that have certain common specific properties, i.e., passing the same 3D point. The process of 3D normalization includes three steps: 3D alignment (orientation normalization), translation (position normalization), and scaling (size normalization).

3D model alignment is to transform a model into a canonical coordinate frame, where the representation of the model is independent of its scale, orientation, and position. Two commonly used 3D model alignment methods are Principal Component Analysis (PCA) [38] and its descendant Continuous Principal Component Analysis (CPCA) [8] which considers the area of each face. They utilize the statistical information of vertex coordinates and extract three orthogonal components with largest extent to depict the principal axes of a 3D model. CPCA is generally regarded as a more stable PCA-based method. In addition, Johan et al. [39] proposed a 3D alignment algorithm based on Minimum Projection Area (MPA) motivated by the fact that many objects have normalized poses with minimum projection areas. That is, for many objects, one of their canonical views has a minimum projection area compared to the other arbitrary views of the objects. Therefore, they align a 3D model by successively selecting two perpendicular axes with minimum projection areas while the third axis is the cross product of the first two axes. It is shown in [39] that MPA can align most 3D models in terms of axes accuracy (the axes are parallel to the ideal canonical coordinate frame: front-back, left-right, or top-bottom view). It is also robust to model variations, noise, and initial poses. However, compared with the PCA-based approaches, MPA takes a longer time to align 3D models while for this research we want to detect symmetry fast.

After a comparison (see Section 4.3 for more details) of the influences of different 3D model alignment algorithms on the efficiency, accuracy and robustness of our view-based symmetry detection approach, we choose CPCA to align a model before performing symmetry detection. After the alignment with CPCA, we translate the model such that the center of its bounding sphere locates at the origin and scale the model such that its bounding sphere has a radius of 1. After this normalization, the symmetry plane(s) will pass the origin, which helps us to significantly reduce the searching space.

3.2. View Sampling and Viewpoint Entropy Distribution Generation

Vázquez et al. [7] proposed an information theory-related measurement named viewpoint entropy to depict the amount of information a view contains. It is formulated based on the Shannon entropy and incorporates both the projection area of each visible face and the number of visible faces into the definition. However, the original definition was developed based on perspective projection, thus we use its extended version defined in [40] for orthogonal projection.

For each model, we sample a set of viewpoints based on the Loop subdivision [41] on a regular icosahedron, denoted as L_0. We subdivide L_0 n times and denote the resulting mesh as L_n. Then, we set the cameras on its vertices, make them look at the origin (also the center of the model) and apply orthogonal projection for rendering. For a 3D model, to differentiate its different faces, we assign different color to each face during rendering. One example is shown in Fig. 2.

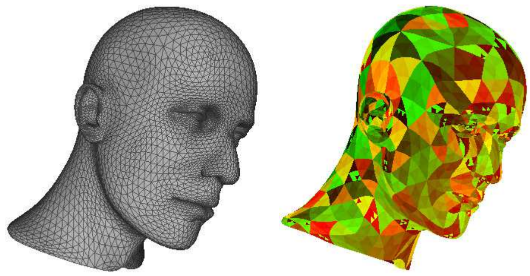


Figure 2: Face color coding example.

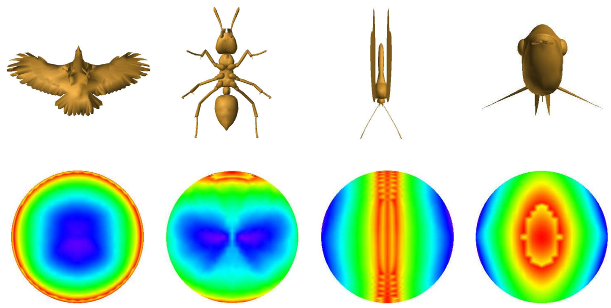


Figure 3: Viewpoint entropy distribution examples: 1^{st} row shows the models after alignment with CPCA; 2^{nd} row demonstrates their respective viewpoint entropy distribution. Blue: large entropy; green: mid-size entropy; red: small entropy.

The viewpoint entropy [40] of a view with m visible faces is defined as follows.

$$E = -\frac{1}{log_2(m+1)} \sum_{j=0}^{m} \frac{A_j}{S} log_2 \frac{A_j}{S} \tag{1}$$

where, A_j is the visible projection area of the j^{th} (j=1, 2, $\cdots$, m) face of a 3D model and A_0 is the background area. S is the total area of the window where the model is rendered: $S = A_0 + \sum_{j=1}^{m} A_j$. Projection area is computed by counting the total number of pixels inside a projected face.

Figure 3 shows the viewpoint entropy distributions of several models by using L_4 (2,562 sample viewpoints) for view sampling and mapping their entropy values as colors on the surface of the spheres based on the HSV color model. We can see there is a perfect correspondence between the symmetry of a model and that of its viewpoint entropy distribution sphere: their symmetry planes are the same. Therefore, the symmetry of a 3D model can be decided by finding the symmetry in the entropy distribution, thus avoiding the high computational cost of direct matching among its geometrical properties. What's more, since viewpoint entropy is computed based on the projection of each face, it is highly sensitive to small differences in the model. In addition, each viewpoint simultaneously captures the properties of many vertices and faces of a model as a whole, which also helps to significantly reduce the computational cost. We also find that it is already accurate enough based on a coarse view sampling, such as using L_1, as demonstrated in Section 4.2. Motivated by these findings, we propose to detect the symmetry of a 3D model based on its viewpoint entropy distribution.

3.3. *Symmetry Detection Based on Iterative Feature Pairing*

Even only using L_1 (42 viewpoints) for view sampling, if based on a naive matching approach by first directly selecting half of the total viewpoints and then matching them with the remaining half, it will result in $P(42, 21)$=2.75×10^{31} combinations. Thus, we develop a much more efficient symmetry detection method based on the following idea: iteratively select a matching pair of viewpoints to generate a symmetry plane and then verify all the rest matching pairs to see whether they are symmetric as well w.r.t the symmetry plane or at least in the symmetry plane. The method is listed in Algorithm 1.

Algorithm 1: Symmetry detection by iterative pairing

Input : N: number of viewpoints;
$Pos[N]$: positions of N viewpoints;
$E[N]$: entropy values of N viewpoints;
n: icosahedron subdivision level;
δ=0.015: entropy difference threshold;
ϵ=1e-5: small difference in double values
Output: Symmetry planes' equations, if applicable
begin
  `// loop symmetric viewpoint pairs (u, v)`
  for $u \leftarrow 0$ **to** $N-2$ **do**
    $P_u \longleftarrow Pos[u]$;
    for $v \leftarrow u+1$ **to** $N-1$ **do**
      if $|E[u]-E[v]| > \delta * \min\{E[u], E[v]\}$ **then**
        continue;
      $P_v \longleftarrow Pos[v]$, $T_1 \longleftarrow normalize(P_u - P_v)$;
      $matches \longleftarrow 2$;
      `// verify other matching pairs`
      for $i \leftarrow 0$ **to** $N-2$ **do**
        if $i == u$ OR $i == v$ **then**
          continue;
        $P_i \longleftarrow Pos[i]$;
        for $j \leftarrow i+1$ **to** $N-1$ **do**
          if $j == u$ OR $j == v$ OR $j == i$ **then**
            continue;
          if $|E[i]-E[j]| > \delta * \min\{E[i], E[j]\}$ **then**
            continue;
          $P_j \longleftarrow Pos[j]$, $P_m \longleftarrow \frac{P_i+P_j}{2}$;
          $T_2 = normalize(P_i - P_j)$;
          $CT = T_1 \times T_2$, $DT = T_1 \cdot T_2$;
          if $||CT|| > \epsilon$ AND $|DT| \neq 0$ **then**
            continue;
          if $|T1 \cdot P_m| > \epsilon$ **then**
            continue;
          $matches$=$matches$+2;
          break;
      `// output the symmetry plane`
      if $matches \geq N - 2^{n+2}$ **then**
        Output and visualize the symmetry plane:
        $T_1[0]*x + T_1[1]*y + T_1[2]*z = 0$

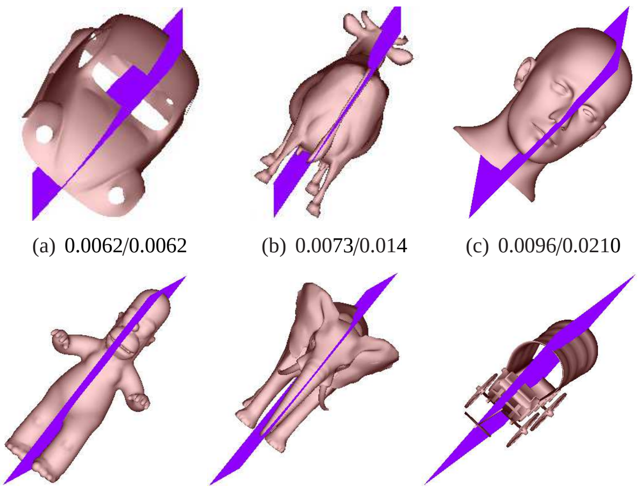
We need to mention the followings for the algorithm. The views corresponding to the viewpoints that are located on the symmetry plane do not need to match each other. While, according to the Loop rule [41], at most 2^{n+2} vertices of L_n are coplanar in a plane w.r.t a great circle. That is to say, at most 2^{n+2} viewpoints could be in the real symmetry plane. An ideal algorithm is to perfectly match w.r.t the symmetry plane all the viewpoint pairs that are not in the symmetry plane. However, we have found that usually there are numerical accuracy problems related to 3D model rendering (e.g. aliasing), viewpoint entropy computation (usually the entropy values of two symmetric viewpoints are not completely the same), as well as possible (either big or minor) differences in mesh triangulation. Therefore, we propose to partially solve this issue by relaxing some of the conditions though it sometimes causes certain false positive detections: if the total number (*matches*) of matched viewpoints w.r.t a candidate symmetry plane is at least $N-2^{n+2}$, then it is confirmed as a symmetry plane. δ is a threshold which can control the strictness of symmetry definition. For example, using a small threshold we detect more strictly defined symmetries while using a bigger threshold, we allow some minor differences and detect rough symmetry properties. T_1 and T_2 are the normals of the planes w.r.t two correspondence points (P_u and P_v; P_i and P_j). The condition $||CT|| > \epsilon$ AND $|DT| \neq 0$ means T_1 and T_2 is neither parallel nor perpendicular to each other. In another word, the line between P_i and P_j is not perpendicular to the candidate symmetry plane since T_1 and T_2 are not parallel (otherwise, $||CT|| = 0$); and P_i and P_j are also not in the symmetry plane (otherwise, $|DT| = 0$). P_m is the midpoint of the line segment connecting points P_i and P_j. It is used to further assert the vertical symmetry property of P_i and P_j about the candidate symmetry plane by finding out whether the midpoint is in the plane, that is $|T_1 \cdot P_m| = 0$. The computational complexity of the algorithm is $O(N^4)$, which is much faster than the combinatorial matching approach: e.g. there are only $N^2 \cdot (N-1)^2/4$=741,321 combinations based on L_1 (N=42), which is 3.71×10^{25} faster than the naive method. In experiments, we select n to be 1.

4. Experiments and Discussions

4.1. *Evaluation w.r.t to Dataset-Level Performance*

We have tested our algorithm on the NIST benchmark [42] and selected models from the AIM@SHAPE Shape Repository [43] to compare with state-of-the-art approaches like the Mean shift [12] and 3D Hough transform [13] based methods, which are among the few papers that deal with global symmetry detection and at the same time provide a quantitative evaluation based on a common set of 3D models. 3D Hough transform [13] can only deal with global symmetry, while Mean shift [12] can deal with partial and approximate symmetry as well. Experiments show that our approach can stably detect the symmetry planes of diverse symmetric models and it also can detect a symmetry plane for a rough symmetric model with a bigger threshold δ.

Figure 4 demonstrates several examples while Table 1 compares their timing information. We need to mention that due to the difference in the specifications of the CPUs used in the experiments, we do not directly compare the absolute running time, but rather we focus on the change of the running time with respect to the increase in the number of vertices of the 3D models. As can be seen, our method shows better computational efficiency property in terms of scalability to the number of vertices. This is mainly because the computational time does not increase linearly with the increase in the number of vertices of a 3D model since we just render the 3D model first and detect its symmetry only based on the rendered views. However, for the other two geometry-based approaches Mean shift and 3D Hough, their computational time is highly affected by the number of vertices of the model. This is because the computational complexity of Mean Shift (in the best case) and 3D Hough is $O(NlogN)$, where N is the number of pairs when only one iteration is needed [13]. Since both of them are geometry-based approach, the value of N as well as their complexity is highly dependent on the number of vertices that a 3D model has. For our case, though the computational complexity of the viewpoint matching step (Section 3.3) is $O(N^4)$, the number of viewpoints N (N=42 in our experiments) is a constant number. Therefore, this matching step has a constant running cost, that is, it is not dependent on the number of vertices.



(a) 0.0062/0.0062 (b) 0.0073/0.014 (c) 0.0096/0.0210

(d) 0.0013/0.0036 (e) 0.0003/0.0027 (f) 0.0005/0.0041

Figure 4: Example symmetry detection results with mean/max error measures [44].

Table 1: Timing information (in seconds) comparison of our methods and other two state-of-the-art approaches: Mean shift [12] and 3D Hough [13] are based on a Pentium M 1.7 GHz CPU according to [13]; while our method is using an Intel(R) Xeon(R) X5675 @ 3.07GHz CPU.

Models	**Cube**	**Beetle**	**Homer**	**Mannequin**
#Vertices	602	988	5,103	6,743
Mean shift	1.8	6.0	91.0	165.0
3D Hough	2.2	3.0	22.0	33.0
Our method	0.7	0.8	1.0	1.1

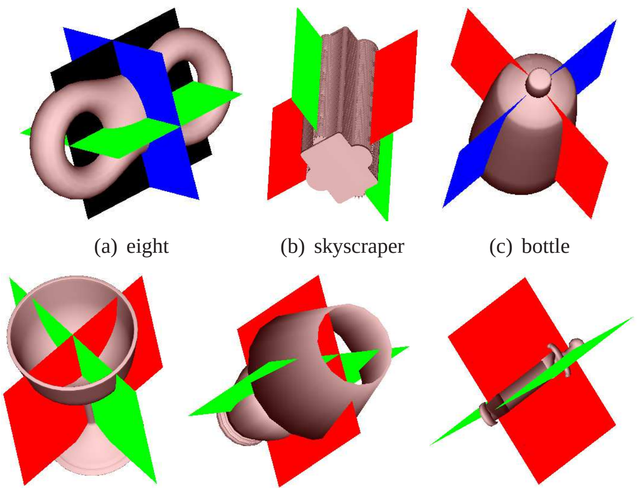
To measure the accuracy of the detected symmetry planes, we adopt the mean (normalized by the surface area) and maximum (w.r.t the bounding box diagonal) distance errors developed in Metro [44] which is based on surface sampling and point-to-surface distance computation. Table 2 compares the mean and max errors of the four models in Table 1 (see Fig. 4 for the errors of other models) with the Mean shift [12] and 3D Hough transform [13] based methods. The errors are computed based on the original mesh and its reflected 3D model w.r.t the detected symmetry plane. As can be seen, our approach achieves much (4∼6 times w.r.t 3D Hough transform and 11∼44 times w.r.t Mean shift) better overall accuracy (see the mean errors), in spite that a few points may not be the most accurate but they still maintain a moderate accuracy (indicated by the max errors).

In addition, it is also very convenient to detect different degrees of symmetries via control of the entropy difference threshold δ. As shown in Fig. 4, there is a minor asymmetry on the the tail part of the cow, while other parts are symmetric. If we want to obtain strict symmetry, a smaller threshold δ (e.g. by reducing it by half: 0.0075) will give the result that it is asymmetric. We also find that our approach can simultaneously detect multiple symmetry planes for certain types of meshes, such as the Eight, Skyscraper, Bottle, Cup, Desk Lamp, and Sword in [43] and [42], as shown in Fig. 5. But we need to mention due to the limitation of CPCA and the sensitivity property to minor changes of the viewpoint entropy feature, there are a few fail cases or certain cases where the proposed method can only partially determine a set of reflection planes. Examples of such models are non-uniform cubes, butterflies, tori, and pears, as demonstrated in Fig. 6: (a) because of non-uniform triangulation, the cube model cannot be perfectly aligned with CPCA, resulting in the unsuccessful symmetry plane detection. However, we have found that for most symmetric models (e.g. Mug, NonWheelChair, and WheelChair classes) that cannot be perfectly aligned with CPCA [8], our approach can still successfully detect their symmetry planes (e.g. the detection rates of Algorithm 1 for those types of models mentioned above are as follow: Mug: 7/8, NonWheelChair: 18/19, and WheelChair: 6/7). Three examples can be found in Fig. 7; (b) the symmetry plane of the butterfly cannot be detected if based on the default threshold δ=0.015, and only after increasing it till 0.0166 we can detect the plane; (c) only the red symmetry plane of the torus is detected based on the default threshold value, while both the red and green planes will be detected if we increase the threshold δ to 0.02 and all the three symmetry planes can be detected if we further increase it till 0.0215; (d) a false positive (blue) symmetry plane of the pear model will appear under the condition of the default threshold, however the error will be corrected with a little smaller threshold of 0.0133. An adaptive strategy of threshold selection is among our next work plan.

Finally, we evaluate the overall performance of our viewpoint entropy distribution-based symmetry detection algorithm based on the NIST benchmark [42]. In total, we have detected 647 symmetry planes for all the 800 models (some of them are asymmetric). To know the general performance of our algorithm, we manually observe the symmetry property of each of the first 200/300/400 models and label its symmetry plane(s)/degree(s) to form the ground truth. Then, we exam-

Table 2: Mean/max errors [44] comparison of our methods and other two state-of-the-art approaches. For the Cube model, since there are three detected symmetry planes, we use their normal directions (x/y/z) to differentiate them.

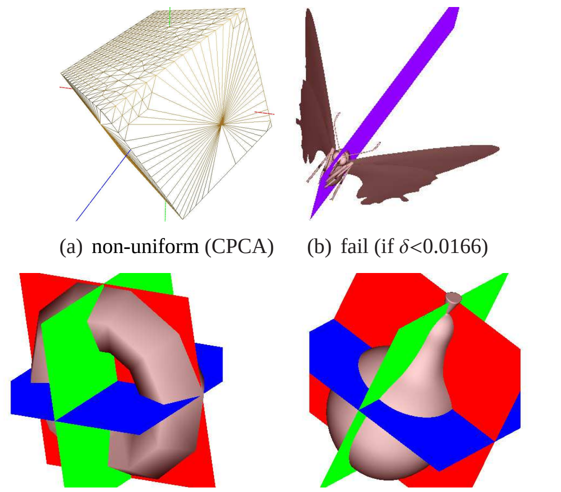
Methods	**Cube**		**Beetle**		**Homer**		**Mannequin**	
	mean	max	mean	max	mean	max	mean	max
Mean shift [12]	N.A.	N.A.	N.A.	N.A.	0.059	0.018	0.111	0.037
3D Hough [13]	N.A.	N.A.	N.A.	N.A.	0.007	0.001	0.046	0.009
Our method	0.0005 (x) 0.0057 (y, z)	0.0008 (x) 0.0082 (y, z)	0.0062	0.0062	0.0013	0.0036	0.0096	0.0210



(a) eight (b) skyscraper (c) bottle

(d) cup (e) desk lamp (f) sword

Figure 5: Multiple detected symmetry planes examples.



(a) non-uniform (CPCA) (b) fail (if δ<0.0166)

(c) partially (if δ<0.0215) (d) one false positive (if δ>0.0133)

Figure 6: Failed or partially failed examples.

ine each detected symmetry plane to see whether it is a True Positive (TP) or False Positive (FP). Similarly, we set the True Negative (TN) value of a model to be 1 if it is asymmetric and our algorithm also does not detect any symmetry plane. While, if a symmetry plane of a symmetric model is not detected, we increase its False Negative (FN) by 1. Table 3 gives the evaluation results (177/277/386 detected symmetry planes) on the first 200/300/400 models (having 191/278/388 symmetry planes in total), which are uniformly divided into 10/15/20 classes. Here, for later analysis we successively list the names of the 20 classes: Bird, Fish, NonFlyingInsect, FlyingInsect, Biped, Quadruped, ApartmentHouse, Skyscraper, SingleHouse, Bottle, Cup, Glasses, HandGun, SubmachineGun, MusicalInstrument, Mug, FloorLamp, DeskLamp, Sword, and Cellphone.

Table 3: Overall symmetry detection performance of our algorithm based on the first 200/300/400 models of the NIST benchmark.

# models	**TP**	**FP**	**TN**	**FN**
200	141	36	37	32
300	216	61	60	45
400	292	94	77	77

Based on the TP, FP, TN and FN values, we compute the following nine detection evaluation metrics [45], as listed in Table 4: Tracker Detection Rate (TRDR, $\frac{TP}{TG}$), False Alarm Rate (FAR, $\frac{FP}{TP+FP}$), Detection Rate (DR, $\frac{TP}{TP+FN}$), Specificity (SP, $\frac{TN}{FP+TN}$), Accuracy (AC, $\frac{TP+TN}{TF}$), Positive Prediction (PP, $\frac{TP}{TP+FP}$), Negative Prediction (NP, $\frac{TN}{FN+TN}$), False Negative Rate (FNR or Miss Rate, $\frac{FN}{FN+TP}$), and False Positive Rate (FPR, $\frac{FP}{FP+TN}$), where the total number of symmetry planes in the 200/300/400 Ground Truth models TG=191/278/388 and the total number of our detections (including both trues and falses) TF=TP+FP+TN+FN=246/382/540. As can be seen, besides the better accuracy in the detected symmetry planes as mentioned before, our detection performance (e.g., for the first 200/300/400 models, Detection Rate DR=81.50%/82.76%/79.13%, and Tracker Detection Rate TRDR=73.82%/77.70%/75.26%) is also good enough. What's more, the minor difference among the detection performance of our algorithm on the 200, 300 and 400 models shows that the overall performance of our algorithm is stable and robust in terms of model type diversity and number of models evaluated.

In a word, as demonstrated by all the above evaluation results, better accuracy and efficiency than state-of-the-art approaches have been achieved by our simple but effective symmetry detection method. It also has good stability in dealing with various model types.

4.2. *Evaluation w.r.t to Robustness*

Robustness to View Sampling. First, we also test our algorithm with different levels of subdivided icosahedron for the view

Table 4: Overall symmetry detection accuracy of our algorithm based on the first 200/300/400 models of the NIST benchmark.

# models	**TRDR**	**FAR**	**DR**	**SP**	**AC**	**PP**	**NP**	**FNR**	**FPR**
200	73.82%	20.34%	81.50%	50.68%	72.36%	79.66%	53.62%	18.50%	49.32%
300	77.70%	22.02%	82.76%	49.59%	72.25%	77.98%	57.14%	17.24%	50.41%
400	75.26%	24.35%	79.13%	45.03%	68.33%	75.65%	50.00%	20.87%	54.97%

sampling, e.g., L_2, L_3, and L_4. Table 5 compares the mean/max errors and running time for the four models listed in Table 1. As can be seen, increasing the view sampling often cannot increase the accuracy while the running time will be significantly increasing. Thus, we choose to sample the views based on L_1 which gives better overall performance in both the accuracy and efficiency.

Robustness to Number of Vertices. We also test the robustness of our algorithm w.r.t the change of the (especially large) number of vertices (resolution) that a 3D model contains. We first subdivide a triangular mesh into its finer version based on several iterations of midpoint subdivision by utilizing the tool of MeshLab [46] and then use the resulting meshes for the test and comparison. We have tested the Elephant, Mannequin and Cube models, and found that our algorithm can stably and accurately detect their symmetry planes, independent of the number of vertices. Table 6 compares their mean/max errors and timings. We can see that the increase in computational time is often significantly slower (especially for models with an extremely large number of vertices; e.g. for Mannequin (467,587 vertices) and Cube (196,610 vertices) they are about 8 and 28 times slower, respectively) than the increase in the number of vertices since rendering the sampling views to compute their viewpoint entropy dominates the running time.

Table 6: Mean/max errors and timing comparison of our algorithm w.r.t the robustness to different number of vertices. For the Cube model, since there are three detected symmetry planes, we use their normal directions (*x*/*y*/*z*) to differentiate them.

Models	**#Vertices**	**mean**	**max**	**time**
Elephant	29,285	0.0003	0.0027	3.0
	116,920	0.0003	0.0027	12.3
	467,252	0.0003	0.0027	48.4
Mannequin	17,450	0.0091	0.0210	2.6
	29,194	0.0091	0.0210	3.8
	467,587	0.0091	0.0210	48.2
Cube	6,146	0.0050 (x)	0.0077 (x)	1.5
		0.0082 (y)	0.0137 (y)	
		0.0061 (z)	0.0093 (z)	
	24,578	0.0002 (x)	0.0003 (x)	3.0
		0.0002 (y)	0.0004 (y)	
		0.0001 (z)	0.0001 (z)	
	196,610	0.0003 (x)	0.0005 (x)	5.8
		0.0003 (y)	0.0004 (y)	
		0.0001 (z)	0.0002 (z)	

Robustness to Noise. Finally, we want to test the versatility as well as sensitivity of our algorithm when processing a modified version of a symmetric model by adding a certain amount of noise. Due to certain factors such as creation, storage, transmission, and modification, 3D models can be noisy. A symmetry detection algorithm should be robust, thus still applicable in the case of small amounts of noise. We test the robustness of our symmetry detection algorithm against noise by randomly adding a small amount of displacement to the vertices of a 3D model.

Figure 8 demonstrates the detected symmetry planes of three example models. Table 7 shows a comparative evaluation on the detection results w.r.t the mean/max errors and the minimum entropy difference threshold value, denoted by $\min\delta$, for a successful detection of the symmetry plane(s) of a model. The results show that our algorithm has a good robustness property against a small amount of noise: by choosing different levels of entropy difference threshold values δ, we will have different tolerant levels of noise to detect symmetry planes. That is, a symmetry detection will be possible if we choose a bigger threshold if there exists a bigger amount of noise. This is contributed to our utilization of the accurate viewpoint entropy feature with a threshold for the feature paring process, since in general viewpoint entropy is stable under small changes in the vertices' coordinates of a 3D model.

4.3. Evaluation w.r.t Different 3D Alignment Algorithms

Considering the apparent advantages of the Minimum Projection Area (MPA)-based 3D alignment algorithm in finding the ideal canonical coordinate frame of a model, besides CPCA, we also evaluate the performance of a variation of our algorithm by only replacing the CPCA algorithm module with MPA. However, we found that the results are not as stable as those of the original CPCA-based version in terms of the percentage of either true or false positives based on the same threshold (δ). Choosing the threshold is also more difficult and sensitive when employing MPA since bigger threshold usually results in more false positives.

An initial analysis based on the experimental results is as follows. Due to the viewpoint sampling precision in MPA, especially for the search of the second principle axis of a 3D model which is based on a step of 1 degree, the axes found by MPA is not precise enough for this viewpoint entropy-based symmetry detection purpose, though for the 3D model retrieval application, as mentioned in the paper, the accuracy is enough. However, since our algorithm directly uses the cameras' locations to compute the symmetry plane(s) by just utilizing their correspondence relationships, it requires that the 3D model is as accurately as possible aligned w.r.t the three standard axes in order to reduce the search space and the number of viewpoints to achieve better efficiency.

Table 5: Mean/max errors and timing comparison of our algorithm with different view sampling. For the Cube model, since there are three detected symmetry planes, we use their normal directions ($x/y/z$) to differentiate them.

View sampling	Cube			Beetle			Homer			Mannequin		
	mean	max	time	mean	max	time	mean	max	time	mean	max	time
L_1	0.0005 (x) 0.0057 (y) 0.0057 (z)	0.0008 (x) 0.0082 (y) 0.0082 (z)	0.7	0.0062	0.0062	0.8	0.0013	0.0036	1.0	0.0096	0.0210	1.1
L_2	0.0005 (x)	0.0008 (x)	3.4	0.0062	0.0062	3.6	0.0013	0.0036	3.8	0.0096	0.0210	3.7
L_3	0.0057 (y)	0.0082 (y)	22.6	0.0062	0.0062	16.9	0.0013	0.0036	19.5	0.0096	0.0210	27.3
L_4	0.0057 (z)	0.0082 (z)	2481.7	0.0062	0.0062	1048.0	0.0013	0.0036	1600.5	0.0096	0.0210	3465.1

Table 7: Comparison of the mean/max errors and the minimum entropy difference threshold values min δ of our algorithm for successful symmetry detections of the variations of three example models after we add different levels of noise.

Noise level (%)	Beetle			Homer			Mannequin		
	mean	max	min δ	mean	max	min δ	mean	max	min δ
0.0	0.006	0.006	0.003	0.001	0.004	0.002	0.010	0.021	0.012
0.1	0.010	0.010	0.003	0.004	0.006	0.002	0.010	0.022	0.011
0.5	0.019	0.022	0.008	0.005	0.011	0.003	0.012	0.022	0.009
1.0	0.010	0.022	0.013	0.008	0.019	0.007	0.012	0.026	0.012

What's more, to align a 3D model, MPA usually takes around 30 seconds if based on 40 Particle Swarm Optimization (PSO) iterations while CPCA needs less than 1 second, which demonstrates another advantage of CPCA over MPA. In addition, we also have found that if based on CPCA, using bounding sphere for the 3D normalization can achieve more accurate results than the case of using bounding box. This should be due to the fact that we also sample the viewpoints on the same bounding sphere. However, if based on MPA, either using bounding sphere or bounding box has only trivial influence on the symmetry detection performance. The reason is that the accuracy of the found axes has much more direct and decisive influence on the symmetry detection performance. In conclusion, using CPCA is more stable, accurate and efficient than MPA, but we believe an improved MPA algorithm should be more promising in thoroughly solving existing errors in CPCA and achieving even better results, which is among our future work.

4.4. Limitations

Firstly, though in Section 4.1 we have performed an overall symmetry detection evaluation of our algorithm on the first 200/300/400 models of the NIST benchmark, we could not perform a comparative evaluation, similar to the one we did based on the four models in Section 4.1, in terms of the accuracy of the detected symmetry planes. The main difficulty is that: to the best of our knowledge, few prior symmetry detection papers evaluated their symmetry detection performance on a benchmark dataset, which is also not available till now. In addition, their code is not publicly available to facilitate such comparative evaluation.

Secondly, we mainly concentrated on global symmetry detection performance when we compared our algorithm with Mean shift [12] and 3D Hough transform [13], though as mentioned in Section 3.3 our approach can perform approximate symmetry detection as well: "using a bigger threshold, we allow some minor differences and detect rough symmetry properties".

In fact, global approximate symmetry detection is one of the two research topics (another one is, partial and approximate symmetry detection) in Mean shift [12]. While, global symmetry detection is the only topic for 3D Hough transform [13], which also compares with Mean shift [12] in its experiment section, in terms of the performance of global symmetry detection accuracy and efficiency, and based on the same model set as ours. All the available (for us) models selected from the model set have been tested and compared in Fig. 4 and Tables 1~2. We also referred to some of the evaluation results of 3D Hough transform [13] as well for a quantitative comparison.

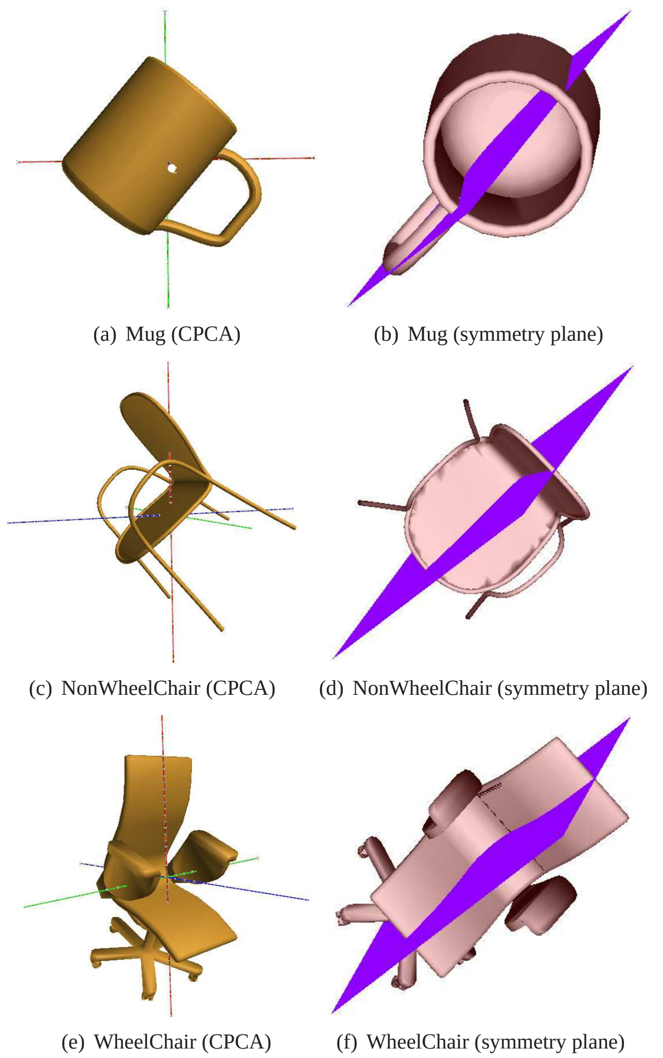
Although we have noticed that there are other related global symmetry detection papers, such as [47] and [48], mainly due to the fact that their code/executable is not available, we have not performed a comparison with them. But, according to the above facts, we believe it is enough and even better to compare with the two more recent works: Mean shift [12] and 3D Hough transform [13].

5. Applications

Finally, we also explore two interesting applications of our symmetry detection algorithm: 3D model alignment and best view selection.

5.1. 3D Model Alignment

As we know, the main shortcoming of PCA-based approach is that the directions of the largest extent found based on the purely numerical PCA analysis are not necessarily parallel to the axes of the ideal canonical coordinate frame of a 3D model. This is because during the alignment process it lacks semantic



(a) Mug (CPCA) (b) Mug (symmetry plane)

(c) NonWheelChair (CPCA) (d) NonWheelChair (symmetry plane)

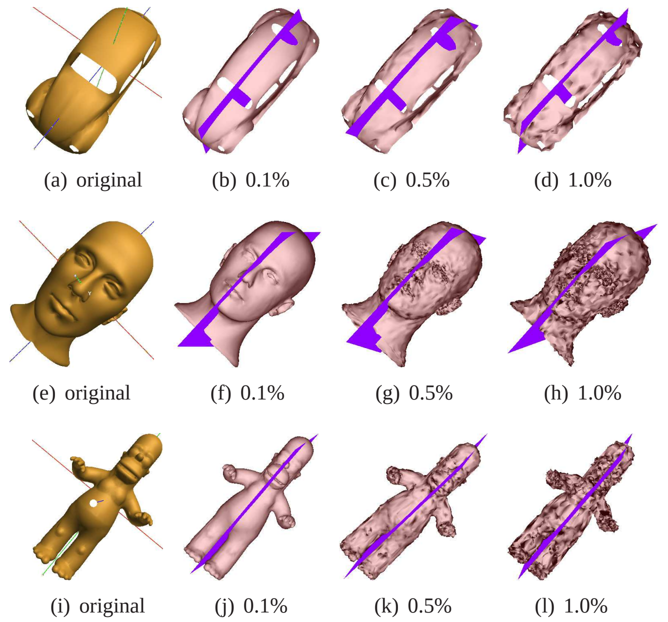
(e) WheelChair (CPCA) (f) WheelChair (symmetry plane)

Figure 7: Examples to demonstrate that our algorithm can successfully detect the symmetry planes for most symmetric models that are not perfectly aligned with CPCA: first column shows the CPCA alignment results; second column demonstrates the detected symmetry planes.

analysis in a 3D model's symmetry [2] [15], or its stability [49] after the alignment.

Based on the detected symmetry planes and the basic idea of PCA, it is straightforward to apply our algorithm to 3D alignment: the first principal axis gives the maximum symmetry degree (that is, it has the smallest total matching cost in terms of viewpoint entropy for the symmetric viewpoint pairs w.r.t the axis) and the second principal axis is both perpendicular to the first axis and also has the maximum symmetry degree among all the possible locations within the perpendicular plane. Finally, we assign the orientations of each axis. This alignment algorithm is promising to achieve similar results as those in [15] which is based on a planar-reflective symmetry transform, while outperforms either PCA or CPCA for certain models with symmetry plane(s). However, our algorithm has better efficiency than [15], thus will be more promising for related real-time applications including 3D model retrieval.

Now we present some experimental results of the above alignment algorithm. As mentioned in Section 2, there are four reflection symmetry types: cyclic, dihedral, rotational, and unique. In fact, some of our previous experiments already demonstrate the main alignment results of several models which fall into three of the above four types. For instance, Fig. 5 shows the two/three principal planes (thus axes) of six models that have a cyclic reflection symmetry (see (c) bottle, (d) cup, and (e) desk lamp), or dihedral reflection symmetry (see (a) eight, (b) skyscraper, and (f) sword). Fig. 4 and Fig. 7 demonstrate the first principle planes/axes of several example models with a unique symmetry based on our idea. It is a trivial task to continue to find other principle axes. For completeness, for example, in Fig. 9, we demonstrate the complete alignment results of three models that have a rotational symmetry, or do not have any reflection symmetry (zero symmetry), or have an approximate symmetry. In a word, the alignment algorithm is promising to be used in dealing with diverse types of models with different degrees of symmetries.



(a) original (b) 0.1% (c) 0.5% (d) 1.0%

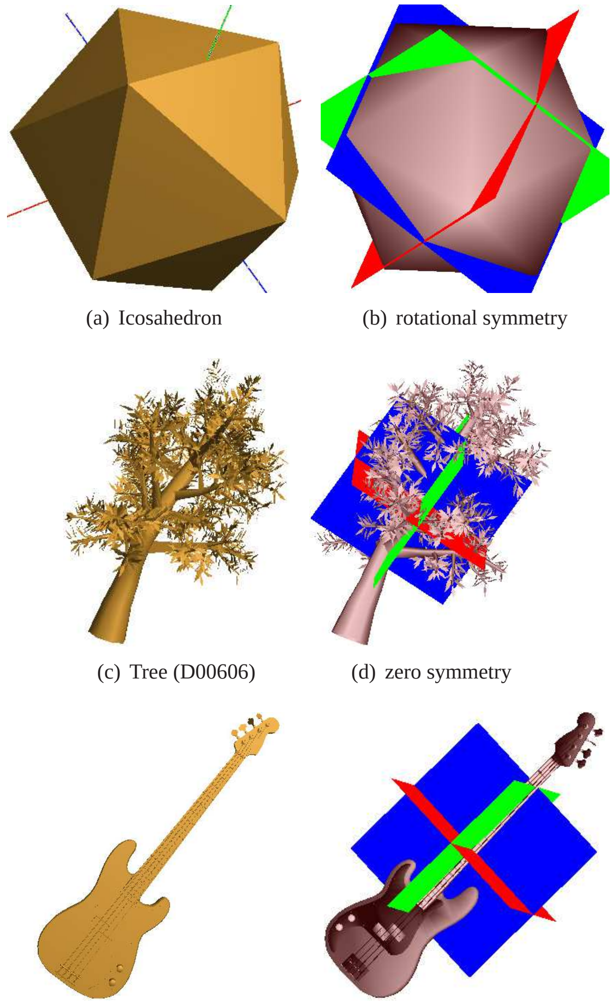
(e) original (f) 0.1% (g) 0.5% (h) 1.0%

(i) original (j) 0.1% (k) 0.5% (l) 1.0%

Figure 8: Examples indicating our algorithm's robustness to noise: symmetry detection results of our algorithm in dealing with model variations with different levels of noise. The first column: original 3D models without adding any noise; The second to the fourth columns: detection results of the models when we add noise by randomly moving each vertex with a small displacement vector whose norm is bounded by 0.1%, 0.5%, and 1% of the diameter of each model's bounding box, respectively.

5.2. Best View Selection

Here, we provide another option to define and search for the best view of a 3D model based on our algorithm. Our definition of symmetry is related to viewpoint entropy which indicates the amount of information that a view contains. In an analogy to 3D model alignment, we use the total viewpoint entropy matching cost, that is an indicator of asymmetry, to indicate the goodness of a candidate best view corresponding to a viewpoint: the bigger the summed matching cost is, the better (more asymmetry) the viewpoint is, since it indicates that there is less redundant in-



(a) Icosahedron (b) rotational symmetry

(c) Tree (D00606) (d) zero symmetry

(e) MusicalInstrument (D00292) (f) approximate symmetry

Figure 9: Alignment results of Icosahderon and two other example models from the NIST dataset. Icosahedron has 15 symmetry planes, Tree type model D00606.off has no symmetry plane, while MusicalInstrument type model D00292.off has a roughly symmetry plane.

formation in the view. When we compute the viewpoint matching cost of a candidate view, we only consider visible viewpoints as seen from the candidate view, for instance, within 180 degrees. Algorithm 1 targets finding the minimum viewpoint matching cost in terms of entropy, while we now want to find the viewpoint that gives a maximum viewpoint entropy matching cost. Thus, we develop our algorithm for this application by modifying Algorithm 1, including changing the "$>$"s or "$\geq$"s to their inverses and setting a bigger threshold δ (e.g., 0.2 in our experiments). The complete best view selection algorithm is given in Algorithm 2. Fig. 10 demonstrates several promising informative example results based on the Algorithm 2 (using L_1 for view sampling).

6. Conclusions and Future Work

In this paper, we have proposed an efficient and novel view-based symmetry detection algorithm based on viewpoint entropy distribution. We have compared with the two latest symmetry detection approaches based on a common set of selected models and demonstrated the better performance of our method in terms of accuracy and efficiency. A detailed evaluation of our approach on a dataset of 400 models and the promising results of two related applications also validate its good robustness, detection rate, and flexibility.

To further improve and explore the algorithm, we list several promising directions here as our next work. Firstly, traditional PCA-based approaches cannot guarantee that the directions of the largest extent are parallel to the axes of the ideal canonical coordinate frame of 3D models. One promising approach to achieve further improvement in terms of alignment accuracy is an improved version of the Minimum Projection Area (MPA) [39] alignment method. We can improve its accuracy to meet our precision requirement by applying the PSO-based method used in the first principle axis search in the search for the second principle axis as well. We are also interested in combining it with CPCA for the 3D alignment process: first performing CPCA for an initial alignment and then correcting possible tilt views (poses) by utilizing a similar idea as MPA. It is promising to help to achieve even better symmetry detection performance, especially for decreasing the percentage of False Negative (FN) since more symmetry planes can be successfully detected, thus avoiding the fail case like Fig. 6 (a) because of the limitation of CPCA.

Secondly, to further improve the efficiency of our algorithm, we could consider Hough transform for symmetry evidence voting. For example, each pair of matched viewpoints casts a vote for their bisecting plane, while the peaks of the voting distribution correspond to prominent symmetry planes. We need to mention that directly applying Hough voting may not work because rather like geometric values, symmetric viewpoints do not perfectly match each other based on their viewpoint entropy values, which has been explained in Section 3.3.

Finally, an automatic and adaptive strategy to select an appropriate threshold δ for respective models or classes is another interesting research direction and deserves our further exploration.

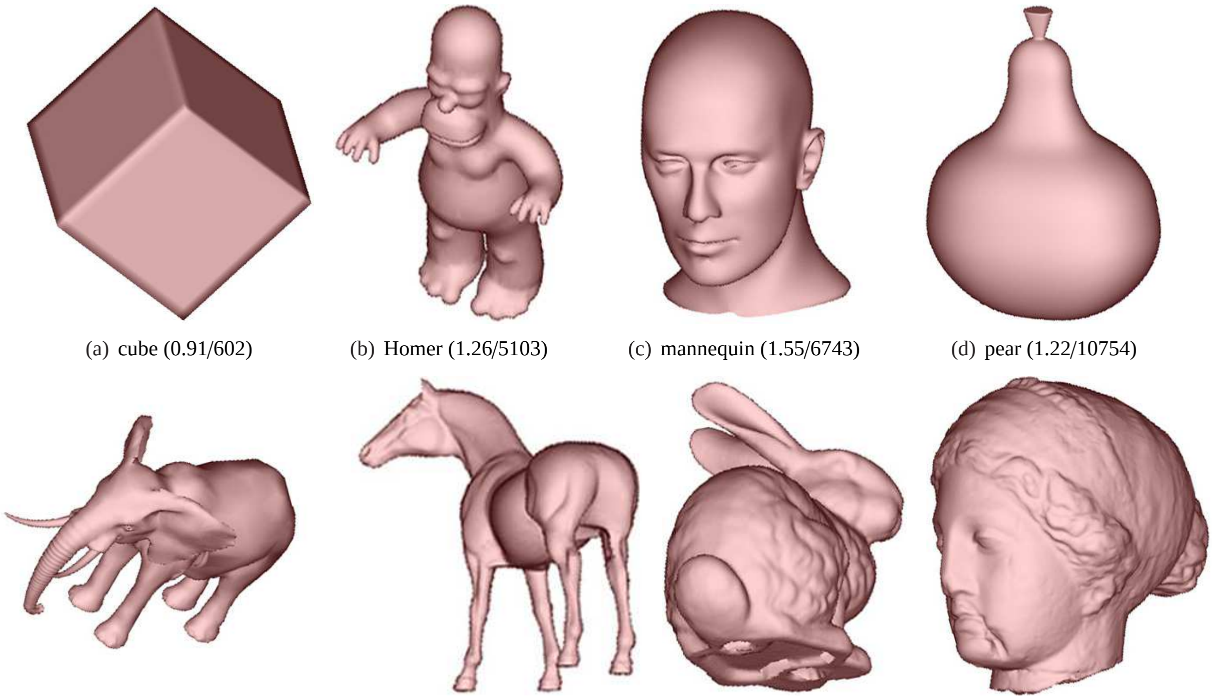
Acknowledgments

The work of Bo Li, Yuxiang Ye and Yijuan Lu is supported by the Texas State University Research Enhancement Program (REP), Army Research Office grant W911NF-12-1-0057, and NSF CRI 1305302 to Dr. Yijuan Lu.

Henry Johan is supported by Fraunhofer IDM@NTU, which is funded by the National Research Foundation (NRF) and managed through the multi-agency Interactive & Digital Media Programme Office (IDMPO) hosted by the Media Development Authority of Singapore (MDA).

References

[1] Y. Liu, H. Hel-Or, C. S. Kaplan, L. J. V. Gool, Computational symmetry in computer vision and computer graphics, Foundations and Trends in Computer Graphics and Vision 5 (2010) 1–195.



(a) cube (0.91/602) (b) Homer (1.26/5103) (c) mannequin (1.55/6743) (d) pear (1.22/10754)

(e) elephant (1.85/19753) (f) horse (2.50/19851) (g) Bunny (3.01/34834) (h) Igea (10.62/134345)

Figure 10: Best views (in terms of asymmetry property) of eight example models. The two numbers in each parenthesis are the running time (in seconds) for the model based on a computer with an Intel(R) Xeon(R) X5675 @ 3.07GHz CPU and the number of vertices the model has.

[2] M. Chaouch, A. Verroust-Blondet, Alignment of 3D models, Graphical Models 71 (2009) 63–76.

[3] M. M. Kazhdan, T. A. Funkhouser, S. Rusinkiewicz, Symmetry descriptors and 3D shape matching, in: J.-D. Boissonnat, P. Alliez (Eds.), Symp. on Geom. Process., volume 71 of *ACM International Conference Proceeding Series*, Eurographics Association, 2004, pp. 115–23.

[4] J. Podolak, A. Golovinskiy, S. Rusinkiewicz, Symmetry-enhanced remeshing of surfaces, in: A. G. Belyaev, M. Garland (Eds.), Symp. on Geom. Process., volume 257 of *ACM International Conference Proceeding Series*, Eurographics Association, 2007, pp. 235–42.

[5] P. D. Simari, D. Nowrouzezahrai, E. Kalogerakis, K. Singh, Multi-objective shape segmentation and labeling, Comput. Graph. Forum 28 (2009) 1415–25.

[6] K. Sfikas, T. Theoharis, I. Pratikakis, Rosy+: 3D object pose normalization based on PCA and reflective object symmetry with application in 3D object retrieval, Int. J. Comput. Vis. 91 (2011) 262–79.

[7] P.-P. Vázquez, M. Feixas, M. Sbert, W. Heidrich, Viewpoint selection using viewpoint entropy, in: T. Ertl, B. Girod, H. Niemann, H.-P. Seidel (Eds.), VMV, Aka GmbH, 2001, pp. 273–80.

[8] D. Vranic, 3D Model Retrieval, Ph.D. thesis, University of Leipzig, 2004.

[9] B. Li, H. Johan, Y. Ye, Y. Lu, Efficient view-based 3D reflection symmetry detection, in: SIGGRAPH Asia 2014 Creative Shape Modeling and Design, Shenzhen, China, December 03 - 06, 2014, 2014, p. 2. URL: `http://doi.acm.org/10.1145/2669043.2669045`. doi:`10.1145/2669043.2669045`.

[10] T. Sawada, Z. Pizlo, Detecting mirror-symmetry of a volumetric shape from its single 2D image, in: CVPR Workshops, CVPRW '08, 2008, pp. 1 –8. doi:`10.1109/CVPRW.2008.4562976`.

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

[12] N. J. Mitra, L. J. Guibas, M. Pauly, Partial and approximate symmetry detection for 3D geometry, ACM Trans. Graph. 25 (2006) 560–8.

[13] D. Cailliere, F. Denis, D. Pele, A. Baskurt, 3D mirror symmetry detection using hough transform, in: ICIP, IEEE, 2008, pp. 1772–5.

[14] H. Zabrodsky, S. Peleg, D. Avnir, Symmetry as a continuous feature, IEEE Trans. Pattern Anal. Mach. Intell. 17 (1995) 1154–66.

[15] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, T. A. Funkhouser, A planar-reflective symmetry transform for 3D shapes, ACM Trans. Graph. 25 (2006) 549–59.

[16] M. Bokeloh, A. Berner, M. Wand, H.-P. Seidel, A. Schilling, Symmetry detection using feature lines, Comput. Graph. Forum 28 (2009) 697–706.

[17] T. Sawada, Visual detection of symmetry of 3D shapes, Journal of Vision 10 (2010) 4:1–4:22.

[18] H. L. Zou, Y. T. Lee, Skewed rotational symmetry detection from a 2D line drawing of a 3D polyhedral object, Computer-Aided Design 38 (2006) 1224–32.

[19] H. L. Zou, Y. T. Lee, Skewed mirror symmetry detection from a 2D sketch of a 3D model, in: S. N. Spencer (Ed.), GRAPHITE, ACM, 2005, pp. 69–76.

[20] A. Martinet, C. Soler, N. Holzschuch, F. X. Sillion, Accurate detection of symmetries in 3D shapes, ACM Trans. Graph. 25 (2006) 439–64.

[21] K. Xu, H. Zhang, A. Tagliasacchi, L. Liu, G. Li, M. Meng, Y. Xiong, Partial intrinsic reflectional symmetry of 3D shapes, ACM Trans. Graph. 28 (2009).

[22] K. Xu, H. Zhang, W. Jiang, R. Dyer, Z. Cheng, L. Liu, B. Chen, Multi-scale partial intrinsic symmetry detection, ACM Transactions on Graphics (Special Issue of SIGGRAPH Asia) 31 (2012) 181:1–181:10.

[23] J. Liu, Y. Liu, Curved reflection symmetry detection with self-validation, in: R. Kimmel, R. Klette, A. Sugimoto (Eds.), ACCV (4), volume 6495 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 102–14.

[24] D. Raviv, A. M. Bronstein, M. M. Bronstein, R. Kimmel, Full and partial symmetries of non-rigid shapes, International Journal of Computer Vision 89 (2010) 18–39.

[25] C. Li, A. B. Hamza, Symmetry discovery and retrieval of nonrigid 3D shapes using geodesic skeleton paths, Multimedia Tools Appl. 72 (2014) 1027–47.

[26] Y. Wang, K. Xu, J. Li, H. Zhang, A. Shamir, L. Liu, Z.-Q. Cheng, Y. Xiong, Symmetry hierarchy of man-made objects, Comput. Graph. Forum 30 (2011) 287–96.

[27] V. G. Kim, Y. Lipman, X. Chen, T. A. Funkhouser, Möbius transformations for global intrinsic symmetry analysis, Comput. Graph. Forum 29 (2010) 1689–700.

[28] Y. Lipman, T. A. Funkhouser, Möbius voting for surface correspondence,

ACM Trans. Graph. 28 (2009).

[29] H. Wang, P. Simari, Z. Su, H. Zhang, Spectral global intrinsic symmetry invariant functions, Proc. of Graphics Interface (2014).

[30] J. Sun, M. Ovsjanikov, L. J. Guibas, A concise and provably informative multi-scale signature based on heat diffusion, Comput. Graph. Forum 28 (2009) 1383–92.

[31] M. Aubry, U. Schlickewei, D. Cremers, The wave kernel signature: A quantum mechanical approach to shape analysis, in: ICCV Workshops, IEEE, 2011, pp. 1626–33.

[32] A. V. Tuzikov, O. Colliot, I. Bloch, Evaluation of the symmetry plane in 3D MR brain images, Pattern Recogn. Lett. 24 (2003) 2219–33.

[33] J. Tedjokusumo, W. K. Leow, Normalization and alignment of 3D objects based on bilateral symmetry planes, in: T.-J. Cham, J. Cai, C. Dorai, D. Rajan, T.-S. Chua, L.-T. Chia (Eds.), MMM (1), volume 4351 of *Lecture Notes in Computer Science*, Springer, 2007, pp. 74–85.

[34] A. Golovinskiy, J. Podolak, T. A. Funkhouser, Symmetry-aware mesh processing, in: E. R. Hancock, R. R. Martin, M. A. Sabin (Eds.), IMA Conference on the Mathematics of Surfaces, volume 5654 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 170–88.

[35] A. Berner, M. Wand, N. J. Mitra, D. Mewes, H.-P. Seidel, Shape analysis with subspace symmetries, Comput. Graph. Forum 30 (2011) 277–86.

[36] A. Tagliasacchi, H. Zhang, D. Cohen-Or, Curve skeleton extraction from incomplete point cloud, ACM Transactions on Graphics (Special Issue of SIGGRAPH) 28 (2009) Article 71, 9 pages.

[37] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, Z. Su, Point cloud skeletons via laplacian-based contraction, in: Proc. of IEEE Conf. on Shape Modeling and Applications, 2010, pp. 187–97.

[38] I. Jolliffe, Principal Component Analysis (2nd edition), Springer, Heidelberg, 2002.

[39] H. Johan, B. Li, Y. Wei, Iskandarsyah, 3D model alignment based on minimum projection area, The Visual Computer 27 (2011) 565–74.

[40] S. Takahashi, I. Fujishiro, Y. Takeshima, T. Nishita, A feature-driven approach to locating optimal viewpoints for volume visualization, in: IEEE Visualization, IEEE Computer Society, 2005, pp. 495–502.

[41] C. Loop, Smooth Subdivision Surfaces Based on Triangles, Master's thesis, University of Utah, 1987.

[42] R. Fang, A. Godil, X. Li, A. Wagan, A new shape benchmark for 3D object retrieval, in: G. Bebis, et al. (Eds.), ISVC (1), volume 5358 of *Lecture Notes in Computer Science*, Springer, 2008, pp. 381–92.

[43] AIM@SHAPE, http://shapes.aimatshape.net/, 2014.

[44] P. Cignoni, C. Rocchini, R. Scopigno, Metro: Measuring error on simplified surfaces, Comput. Graph. Forum 17 (1998) 167–74.

[45] V. Manohar, P. Soundararajan, H. Raju, D. B. Goldgof, R. Kasturi, J. S. Garofolo, Performance evaluation of object detection and tracking in video, in: P. J. Narayanan, S. K. Nayar, H.-Y. Shum (Eds.), ACCV (2), volume 3852 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 151–61.

[46] MeshLab, http://meshlab.sourceforge.net/, 2014.

[47] C. Sun, J. Sherrah, 3d symmetry detection using the extended gaussian image, IEEE Trans. Pattern Anal. Mach. Intell. 19 (1997) 164–8.

[48] M. M. Kazhdan, B. Chazelle, D. P. Dobkin, T. A. Funkhouser, S. Rusinkiewicz, A reflective symmetry descriptor for 3d models, Algorithmica 38 (2003) 201–25.

[49] H. Fu, D. Cohen-Or, G. Dror, A. Sheffer, Upright orientation of man-made objects, ACM Trans. Graph. 27 (2008).

Algorithm 2: Best view selection based on maximum viewpoint entropy matching cost.

Input : N: number of viewpoints;
$Pos[N]$: positions of N viewpoints;
$E[N]$: entropy values of N viewpoints;
n: icosahedron subdivision level;
δ=0.2: entropy difference threshold;
ϵ=1e-5: small difference in double values

Output: Symmetry planes' equations, if applicable

begin
- `// initialize maximum viewpoint entropy matching cost`
- $max\_cost \longleftarrow 0.0$;
- `// loop viewpoint pairs (u, v)`
- **for** $u \leftarrow 0$ **to** $N-1$ **do**
  - $P_u \longleftarrow Pos[u]$;
  - **for** $v \leftarrow 0$ **to** $N-1$ **do**
    - **if** $u == v$ **then**
      - continue;
    - $P_v \longleftarrow Pos[v]$, $T_1 \longleftarrow normalize(P_u - P_v)$;
    - `// initialize the viewpoint entropy matching cost for the current view`
    - $cur\_cost \longleftarrow 0$;
    - `// matching other viewpoint pairs`
    - **for** $i \leftarrow 0$ **to** $N-2$ **do**
      - **if** $i == u\ OR\ i == v$ **then**
        - continue;
      - $P_i \longleftarrow Pos[i]$;
      - **for** $j \leftarrow i+1$ **to** $N-1$ **do**
        - **if** $j == u\ OR\ j == v\ OR\ j == i$ **then**
          - continue;
        - $P_j \longleftarrow Pos[j]$, $P_m \longleftarrow \frac{P_i+P_j}{2}$;
        - $T_2 = normalize(P_i - P_j)$;
        - $CT = T_1 \times T_2$, $DT = T_1 \cdot T_2$;
        - **if** $|DT| < 0$ **then**
          - continue;
        - **if** $\|CT\| > \epsilon\ AND\ |DT| \neq 0$ **then**
          - continue;
        - **if** $|T1 \cdot P_m| > \epsilon$ **then**
          - continue;
        - $cur\_cost = cur\_cost + |E[i] - E[j]|$;
        - break;
    - **if** $cur\_cost > max\_cost$ **then**
      - $max\_cost = cur\_cost$;
      - $T \longleftarrow T_1$;
- `// output the best view`
- $T[0] * x + T[1] * y + T[2] * z = 0$