Single Depth-image 3D Reflection Symmetry and Shape Prediction

Zhaoxuan Zhang, Bo Dong*, Tong Li, Felix Heide, Pieter Peers, Baocai Yin*, Xin Yang*

Abstract

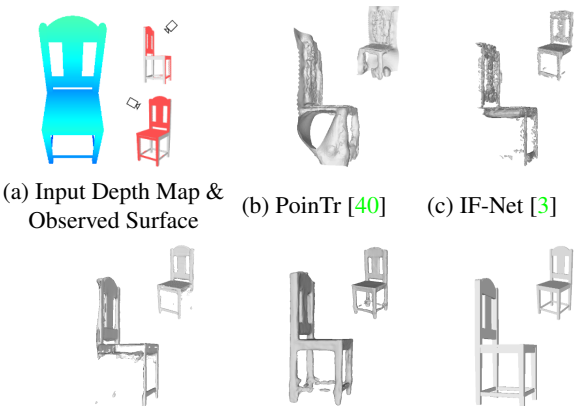
In this paper, we present Iterative Symmetry Completion Network (ISCNet), a single depth-image shape completion method that exploits reflective symmetry cues to obtain more detailed shapes. The efficacy of single depth-image shape completion methods is often sensitive to the accuracy of the symmetry plane. ISCNet therefore jointly estimates the symmetry plane and shape completion iteratively; more complete shapes contribute to more robust symmetry plane estimates and vice versa. Furthermore, our shape completion method operates in the image domain, enabling more efficient high-resolution, detailed geometry reconstruction. We perform the shape completion from pairs of viewpoints, reflected across the symmetry plane, predicted by a reinforcement learning agent to improve robustness and to simultaneously explicitly leverage symmetry. We demonstrate the effectiveness of ISCNet on a variety of object categories on both synthetic and real-scanned datasets.

1. Introduction

Symmetry is an intrinsic geometric property present in the vast majority of man-made and natural objects [12, 17], and which has been exploited to improve the efficacy of applications such as shape matching [12], segmentation [13], object retrieval [8], and robotic grasping [34].

In this paper, we aim to leverage symmetry as an aid for the completion of shapes from partial observations [23, 24, 28]. Recently, Yao *et al*. [39] presented Front2Back, a framework for 3D shape reconstruction from a single RGB image. A key observation in Front2Back is that a (nearly) complete 3D model can be effectively described by a pair of 2.5D visible surface images taken from opposite views (*i.e*., front and back views). Based on this observation, Front2Back completes a shape by synthesizing a back view using a global 3D reflective symmetry plane from a given front view. However, the reconstruction quality is significantly affected by the input front view angle (Figure 1(d)). Moreover, Front2Back assumes the availability of a 'perfect' symmetry plane, which in practice is challenging to obtain from partial observations [21, 22, 45].

*Co-corresponding authors.



(a) Input Depth Map & Observed Surface (b) PoinTr [40] (c) IF-Net [3]

(d) Front2Back [39] (e) Ours (f) Ground Truth

Figure 1. ISCNet more efficiently leverages symmetry cues for robust shape completion from a single depth image compared to prior methods.

We present ISCNet (Iterative Shape Completion Network), a single depth-image shape completion method that explicitly leverages symmetry cues. Our key insight is that symmetry detection and symmetry-based shape completion are complementary tasks that can provide constructive cues to the other task. Symmetry detection can help shape completion by providing object-centric information about missing geometrical features, and shape completion can provide more robust symmetry cues by filling in missing geometry. Departing from existing work [33] that assumes an ideal prediction of the symmetry plane, ISCNet iteratively refines the symmetry plane and the reconstructed shape jointly. Furthermore, unlike Front2Back [39], ISCNet does not rely on the initial input viewpoint to a-priori fix the synthetic back view camera position but instead leverages a trained reinforcement learning (RL) policy to select optimal pairs of reflection viewpoints at each iteration. We empirically found that predetermining the viewpoints is less effective than RL-based viewpoint selection which can more effectively deal with the dynamic shape of a partial point cloud during the refinement process.

Specifically, ISCNet takes as input a single depth image and reprojects it to an incomplete point cloud. Instead of explicitly detecting a symmetry plane, we estimate the pose [30] that aligns the symmetry plane to the X-Y plane. Based on the current estimate of the point cloud and symmetry aligning pose, an RL agent determines a pair of *re-*

flection viewpoints, reflected across the symmetry plane, for extracting a pair of partial depth and normal maps that are subsequently repaired using a novel 2D inpainting module. By directly working on depth image pairs, our method is independent of the size of the point cloud, and it can leverage geometry information from the symmetric counter view. However, because the virtual depth images are reprojections that differ from the input view, not all reprojected points are valid (e.g., the backside of points visible through a 'hole'). Therefore, we first identify reliable pixels before performing depth inpainting in the projected image domain. Finally, we reproject the inpainted maps back to 3D and update the point cloud. We progressively refine the point cloud by alternating between estimating reflection viewpoints and completing the point cloud for 20 iterations, updating the symmetry plane every 4th iteration, to balance efficiency versus efficacy. When updating the symmetry plane, we 'reset' the point cloud to its initial state to mitigate the impact of prior inaccurate symmetry plane estimations on the shape completion.

We validate that ISCNet produces more accurate shape completions than existing methods (Figure 1) and confirm the contribution of each component in our model via a thorough ablation study. In summary, our contributions are: 1) a novel robust joint symmetry-detection / shape completion method; 2) a symmetric viewpoint-based 3D shape completion method for detailed high-resolution shape reconstruction; and 3) a reinforcement learning-based optimal reflection viewpoint selection solution for progressive shape completion.

2. Related Work

Symmetry Detection on 3D Shapes forms a key component in many downstream applications ranging from shape matching [12], to segmentation [13], to 3D object retrieval [8], to shape completion [23, 24, 27, 28] and to deformable object learning [36]. Symmetry detection of incomplete shape observations is significantly more challenging, and recent learning-based solutions have explored the use of multitask learning [21], sampling the distribution of plausible shapes [22], and building 3D cost volumes [45] to predict 3D symmetry planes. An alternative approach to predicting arbitrarily oriented 3D symmetry planes is to estimate a camera pose that aligns the local coordinate frame such that the X-Y plane acts as a canonical symmetry plane [30, 44]. Our symmetry detection module falls in this latter category. However, unlike prior work that a-priori fixes the symmetry information as a prior, we jointly perform shape completion yielding a more robust and mutually reinforcing symmetry detection and shape completion.

Learning-based 3D Shape Completion of Partial Point Cloud Observations have explored coarse-to-fine methods [15, 41], transformer-networks [37, 40], probabilistic modeling [16], and topology-aware approaches [26] to complete the 3D point cloud. However, these methods are memory intensive and their scalability (*e.g.*, beyond ≈ 16384 points) is limited. In contrast, our approach does not impose such limitation on the number of 3D points as our method leverages 2D depth inpainting.

Single-image Shape Reconstruction aims to complete approximately half of the shape either directly from a 2D image [6, 7, 29, 31] or (indirectly) from a 2.5D input [25, 35, 39]. Front2Back [39] estimates, besides depth and surface normals, also symmetry information to aid in the surface completion task. However, Front2Back is limited to a single front-back camera pair. Our method leverages an RL agent to predict multiple virtual reflection viewpoints for shape completion, and it takes better advantage of the symmetry prior by inpainting symmetric image pairs. In addition, we iteratively refine the symmetry plane to obtain more accurate estimates of both shape and symmetry.

3. Iterative-Symmetry Completion Network

Our Iterative Symmetry Completion Network (ISCNet) takes a single-view depth map of a target object as input and iteratively completes the 3D mesh by alternating between refining the symmetry plane, the virtual reflection viewpoints, and the shape completion. To bootstrap the iterative process, we first project the input single-view depth map to an incomplete point cloud that is subsequently passed to a symmetry detection module and point cloud transposition operator. The symmetry detection module (subsection 3.1) provides an estimate of the object's symmetry plane that is subsequently leveraged by the point cloud transposition operator to align the point cloud symmetrically around the estimated symmetry plane. Next, a novel 3D shape completion module (subsection 3.2) progressively completes the 3D point cloud based on a pair of reflection viewpoints selected by an RL agent. For each reflection viewpoint pair, a depth and normal map pair is extracted from the partial point cloud. Due to the partial completion of the point cloud, the extracted depth and normal map pairs exhibit holes. To address this issue, we employ a specifically crafted inpainting module, which not only repairs these holes but also assigns a confidence score to each repaired pixel. Next, the inpainted depth and normal maps are merged into the point cloud guided by their associated confidence scores. Finally, to improve robustness, we iteratively refine the symmetry plane, the reflection viewpoints, and the shape completion (subsection 3.3). We limit the number of iterations to 20 and only update the symmetry plane and reset the point cloud every 4th iteration to balance robustness versus efficiency; Figure 2 visually summarizes ISCNet.

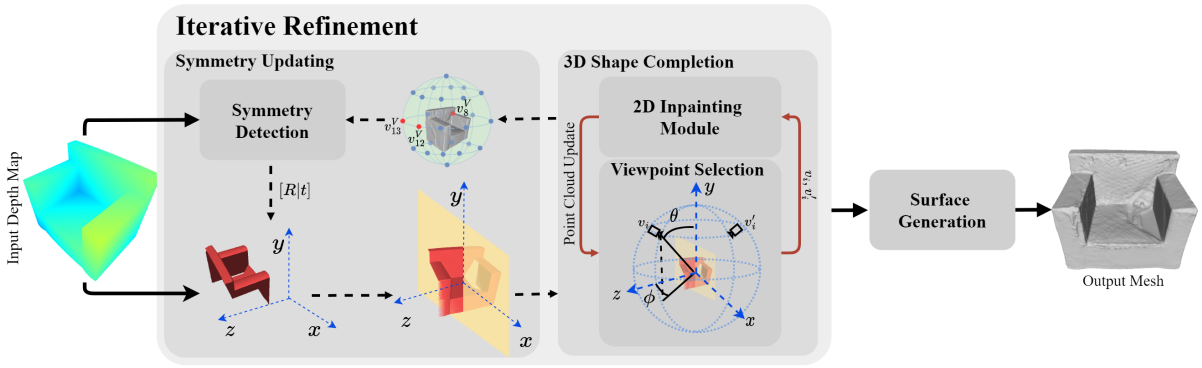


Figure 2. Summary of ISCNet: a partial shape extracted from a single depth-image is aligned such that the X-Y plane corresponds to the (detected) symmetry plane. Next, a pair of depth and normal maps are rendered from reflection viewpoints selected by an RL agent, and each symmetric pair of synthetic depth and normal maps is inpainted in the image-domain (see also Figure 3). These inpainted maps are then back-projected to a point cloud and merged. The symmetry plane is updated every 4th iteration, where the merged shape is then visualized from 3 selected views, v_8^V, v_{12}^V, and v_{13}^V in this case, passing into the symmetry detection module separately, and the best-detected symmetry plane (with the highest confidence score) is used as input for the next iteration.

3.1. 3D Reflection Symmetry Detection

We estimate the symmetry plane of an object via a proxy task, namely, camera pose estimation, such that the symmetry plane aligns in local coordinates with the X-Y plane. Our Symmetry Detection Module follows the architecture of DenseFusion [30]. We pass a depth image to obtain a point cloud, pixel-wise features, and finally a pose estimation. Different from DenseFusion, we only use depth instead of RGB channels, and we do not require an object segmentation stage. Instead of predicting the symmetry parameters directly, we found it more robust to estimate a rotation matrix $R \in SO(3)$ and a translation vector $t \in \mathbb{R}^3$ such that the symmetry plane's center point p_s and normal direction n_s are defined by:

$$\begin{pmatrix} p_s \\ n_s \end{pmatrix} = \begin{pmatrix} R^{-1} \cdot ([0,0,0] - t) \\ R^{-1} \cdot ([0,0,1] - t) \end{pmatrix}. \quad (1)$$

3.2. 3D Shape Completion

Given a predicted symmetry plane, ISCNet progressively completes the 3D shape. Our shape completion module consists of three sequential sub-modules: viewpoint selection, 2D inpainting, and point cloud updating.

Viewpoint Selection Module: ISCNet completes 3D shapes via 2D projected depth and normal maps. The exact choice of virtual viewpoints from which these depth and normals are generated is critical for robust shape completion, and it is sensitive to the irregular structures present in the partially completed point cloud. As the shape of a partial point cloud dynamically changes, it is impractical to pre-determine viewpoints; see our supplementary material for more details.

We model the problem of selecting a viewpoint v_i as a Markov decision process (MDP) because, similar to [38, 42], we are only concerned about the current state of a partial point cloud. To this end, we leverage an RL agent to select an optimal viewpoint trained with an Actor-Critic Proximal Policy Optimisation (PPO) algorithm [20]. The RL agent predicts an optimal viewpoint (*i.e.*, action) based on the current 3D point cloud (*i.e.*, state) to maximize an appropriately designed reward.

Specifically, the continuous action space of the RL agent is defined by θ and ϕ angles (Figure 2), with corresponding viewpoint:

$$x = a \sin\theta \sin\phi, \quad (2)$$
$$y = a \cos\theta, \quad (3)$$
$$z = a \sin\theta \cos\phi, \quad (4)$$

where $\theta \in [0°, 180°]$ and $\phi \in [-180°, 180°]$; a is set at 1.5m. The view direction is always aimed at the object center.

The prediction action reward is determined by four factors: accuracy R^a, shape completeness R^s, number of recovered effective points R^p, and viewpoint diversity R^v. R^a and R^s encourage an agent to choose a new viewpoint that benefits the prediction of missing parts from the original point cloud. R^p and R^v encourage the agent to increase the viewpoint diversity. Formally, the reward function for

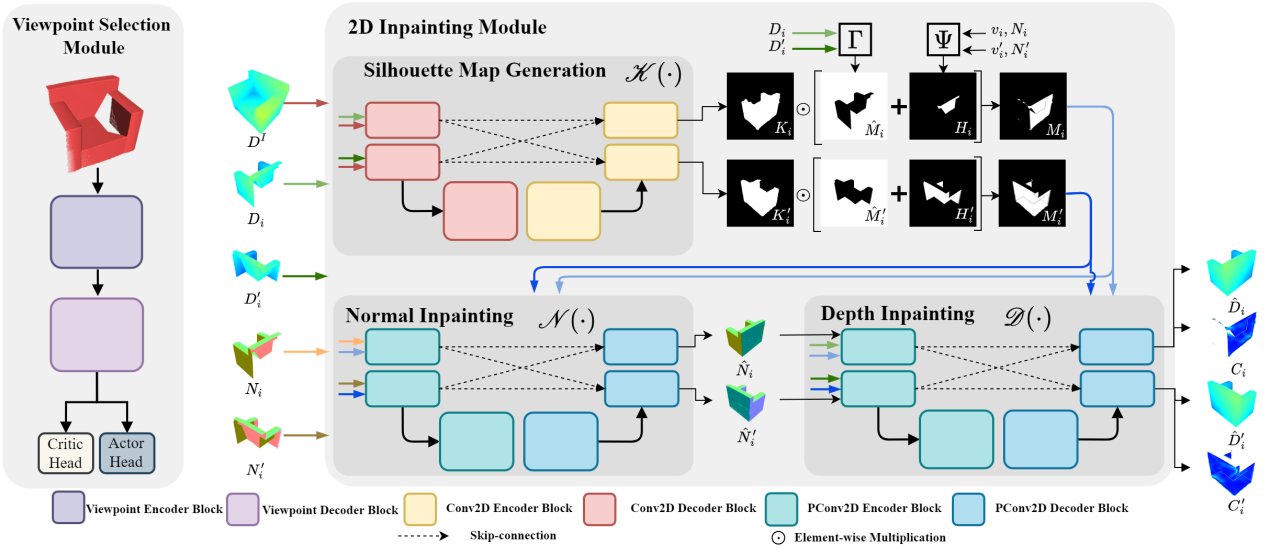


Figure 3. Detailed visual summary of the viewpoint selection and the 2D inpainting modules.

an action is defined as:

$$R = w_1 R^a + w_2 R^s + w_3 R^p + w_4 R^v, \tag{5}$$

$$R^a = \frac{1}{\mathcal{N}(\hat{P})} \sum_{x \in \hat{P}} \min_{y \in P_{GT}} \|x - y\|_2^2, \tag{6}$$

$$R^s = S - S', \tag{7}$$

$$R^p = \mathcal{N}(\{d(x, P') < 0.002 | x \in \hat{P}\}), \tag{8}$$

$$R^v = (|\theta - \theta'| + |\phi - \phi'|)/180 \cdot \pi, \tag{9}$$

$$S = \frac{\mathcal{N}(\{d(y, P) < 0.02 | y \in P_{GT}\})}{\mathcal{N}(P_{GT})}, \tag{10}$$

where P, P', and $\hat{P}$ are the 3D points of the current point cloud, previous point cloud, and recovered point cloud by the action, respectively. Intuitively, $P = P' + \hat{P}$; P_{GT} is the ground truth reference point set; $d(x, P)$ denotes the shortest distance from a point x to a point set P; S and S' are defined based on P and P', respectively; $\mathcal{N}(\cdot)$ counts the number of points of a point set; w_1, w_2, w_3, w_4 are the weights for balancing the contribution of the corresponding reward, which are set to 25, 1, 1.5, and 2 in all experiments.

The architecture of the RL agent is simple yet effective; it contains an encoder, decoder, critic and actor heads (Figure 3). We adopt a commonly used 3D point encoder [15, 18] which extracts features from the input point cloud for downstream processing by the actor and critic. The critic head is only used during training, and the actor head predicts a viewpoint based on the input 3D point cloud.

The 2D Inpainting Module consists of three components: mask generation, normal map inpainting, and depth-map inpainting.

The *mask generator* takes as input symmetric depth map pairs D_i and D_i' corresponding to symmetric views v_i and v_i', and outputs two masks M_i and M_i' that mark the areas for each input map that need to be completed. To compute M_i, we start from an initial completion mask $\hat{M}_i$ obtained by marking all 'black' areas of a depth map D_i. However, this initial mask misses two important corner cases. First, not all 'black' areas should be completed. For example, the space between the legs of a chair. Second, areas with a missing front surface covered by a backside surface also need to be filled in. To cover the first case, we utilize a silhouette map generator to predict silhouette maps K_i of the target object to exclude erroneous 'holes'. The silhouette generation is modified from a standard U-Net architecture [19] by adding an extra input head and output head to address the pairwise inputs and outputs. Furthermore, we add pairwise skip connections between the two input and output heads for exchanging symmetry information. To provide additional depth cues, we concatenate to each input D_i (and D_i') the initial input depth map, D^I, along the channel dimension. To address the second corner case, we compute a front-surface mask H_i by comparing the angle between the camera view direction and the surface normal at each point; we set the front-surface mask to 1 if the angle is less than 90°, or 0 otherwise (*i.e.*, back facing). The final mask M_i is then obtained by combining $\hat{M}_i$, K_i and H_i as:

$$M_i = (\hat{M}_i + H_i) \odot K_i, \tag{11}$$

$$\hat{M}_i = \Gamma(D_i), \tag{12}$$

$$H_i = \Psi(N_i, v_i), \tag{13}$$

$$K_i, K_i' = \mathscr{K}([D_i, D^I], [D_i', D^I]), \tag{14}$$

$$\Gamma(X), \Psi(Z, y) = \begin{cases} 1 & X = 0, \langle Z, y \rangle > 0 \\ 0 & \text{otherwise} \end{cases}, \tag{15}$$

where $\odot$ is the element-wise multiplication; $\mathscr{K}$ denotes the

silhouette map generator; $[\cdot,\cdot]$ indicates a channel-wise concatenation operation; $\Gamma(X)$ and $\Psi(Z,y)$ are two indicator functions, that only differ in the conditions for 1.

The *normal inpainting* submodule $\mathscr{N}$ takes as input a pair of partial normal maps N_i and N_i' corresponding to symmetric viewpoints v_i and v_i' as well as their corresponding masks M_i and M_i', and outputs a pair of completed normal maps $\hat{N}_i$ and $\hat{N}_i'$ respectively:

$$\hat{N}_i, \hat{N}_i' = \mathscr{N}([N_i, \mathbf{1} - M_i], [N_i', \mathbf{1} - M_i']). \tag{16}$$

The *normal inpainting submodule* is based on PConv [14], a partial-convolution-layer-based UNet architecture that is robust to irregularly shaped holes [9, 14]. We modify PConv to accommodate the dual in and output by duplicating the input head and last layer of the decoder and adding pairwise skip connections. Note, PConv also applies a convolution with a unit fixed kernel on the input mask to perform mask-updating, and hence the target inpainting region should be marked 0 in the mask (*i.e.*, the inverse of mask M_i).

The *depth-map inpainting* submodule $\mathscr{D}$ takes as input a pair of partial depth maps D_i and D_i' corresponding to symmetric viewpoints v_i and v_i' as well as their corresponding masks (M_i and M_i') and inpainted normal maps ($\hat{N}_i$ and $\hat{N}_i'$), and outputs a pair of completed depth maps $\hat{D}_i$ and $\hat{D}_i'$ and corresponding confidence maps C_i and C_i':

$$(\hat{D}_i, C_i), (\hat{D}_i', C_i') = \mathscr{D}(\left[D_i, \hat{N}_i, \mathbf{1} - M_i\right], \left[D_i', \hat{N}_i', \mathbf{1} - M_i'\right]). \tag{17}$$

The confidence scores indicate the confidence of the inpainted pixel (*i.e.*, the distance of the estimated depth-map to the ground truth map on missing regions). Our implementation of the depth inpainting submodule is also based on PConv [14] with similar modifications to support the additional input and output channels.

Point Cloud Update: We merge the inpainted depth, normal, and confidence maps into the previous point cloud by reprojecting all points in each depth map and corresponding normals to a point cloud if their confidence score exceeds a threshold (set to 0.5 in our implementation).

3.3. Iterative Refinement

To improve the robustness of the symmetry plane estimation and of the shape completion, we iteratively refine the output of ISCNet. A key observation is that we can select any suitable viewpoint to generate a new depth image to serve as input to update the symmetry plane. Because it is unlikely that a single viewpoint would be ideal for all cases, we project the updated point cloud from three pre-selected trial viewpoints to generate their respective depth maps. Next, the three depth maps are passed (separately) into the symmetry detection module, and the symmetry plane with the highest confidence score is subsequently used for the remainder of our pipeline. The set of pre-determined viewpoints (v_8^V, v_{12}^V and v_{13}^V from VRCNet [16]) are empirically chosen and kept constant for all experiments. The final updated point cloud is converted to a mesh surface using Poisson Surface Reconstruction [11]; see Figure 2.

To balance computational cost versus gain in accuracy, the number of iterations is set to 20 based on an experimental analysis on ShapeNet. We update the symmetry plane every 4th iteration such that the RL agent can leverage the progressively completed point cloud for improving the reflection viewpoint estimates. In addition, to avoid accumulating incorrectly completed shape-parts due to an inaccurate symmetry plane, we reset the point cloud to its initial state when re-estimating the symmetry plane.

3.4. Loss Functions

ISCNet is trained in three stages. In the first stage, the symmetry detection component and 2D inpainting module are trained separately. In the second stage, all components except the RL agent of ISCNet are then jointly fine-tuned. Finally, we train the RL agent by using a PPO algorithm with the parameters of the other components fixed using the loss function from [20].

The first-stage symmetry detection loss is defined by:

$$\mathcal{L}^S = \mathcal{C}(\mathcal{M}(P, S_{\text{E}}), \mathcal{M}(P, S_{\text{G}})) + \mathcal{B}(\Theta_{t_1}(S_{\text{E}}^N, S_{\text{G}}^N), \mathbf{1}), \tag{18}$$

where $\mathcal{C}$ measures the Chamfer Distance (CD) between the reconstructed 3D points P reflected (via a reflection function $\mathcal{M}(\cdot,\cdot)$) over the estimated symmetry plane S_{E} and reference symmetry plane S_{G}. We further regularize this loss via a binary cross entropy (BCE) loss ($\mathcal{B}$) measuring the *acute* angle similarity between S_{E}^N and S_{G}^N, the corresponding normalized symmetry plane normals of the estimated and reference, via an indicator function $\Theta_{t_1}(X,Y)$ defined as:

$$\Theta_{t_1}(X,Y) = \begin{cases} 1 & \arccos(|\langle X, Y\rangle|) <= t_1 \\ 0 & \text{otherwise} \end{cases}, \tag{19}$$

with t_1 a slack parameter set to 5°.

The first-stage inpainting loss is defined as:

$$\mathcal{L}^I = \mathcal{L}_{\text{K}} + \mathcal{L}_{\text{N}} + \mathcal{L}_{\text{D}}, \tag{20}$$

$$\mathcal{L}_{\text{K}} = \mathcal{B}(K_{\text{E}}, K_{\text{G}}), \tag{21}$$

$$\mathcal{L}_N = \|N_E^{M_{\text{G}}} - N_G^{M_{\text{G}}}\|_1, \tag{22}$$

$$\mathcal{L}_{\text{D}} = \|D_{\text{E}}^{M_{\text{G}}} - D_{\text{G}}^{M_{\text{G}}}\|_1 + \mathcal{B}(C_{\text{E}}^{M_{\text{G}}}, \Phi_{t_2}(D_{\text{E}}^{M_{\text{G}}}, D_{\text{G}}^{M_{\text{G}}})), \tag{23}$$

$$X^{M_{\text{G}}} = X \odot M_{\text{G}}, \tag{24}$$

$$\Phi_{t_2}(X,Y) = \begin{cases} 1 & \|X - Y\|_1 <= t_2 \\ 0 & \text{otherwise} \end{cases}, \tag{25}$$

CD/MD ($\times 10^2$)	*Plane*	*Cabinet*	*Car*	*Chair*	*Lamp*	*Sofa*	*Table*	*Vessel*	*Average*
PCN° [41]	2.87/2.44	3.42/3.65	2.12/2.01	4.32/4.08	5.49/5.19	2.88/3.35	4.49/3.72	2.23/2.25	3.48/3.34
MSN° [15]	3.29/2.64	**2.57**/**2.20**	1.95/1.69	3.60/2.91	4.58/4.31	3.61/3.12	3.62/3.04	2.08/1.89	3.16/2.72
PoinTr° [40]	**1.10**/**0.98**	2.86/3.12	**1.72**/**1.65**	3.33/2.92	3.45/2.92	2.58/2.97	3.44/2.82	**1.40**/**1.41**	2.48/2.35
VRCNet° [16]	2.61/1.92	**2.77**/**2.20**	1.92/**1.61**	3.43/2.61	3.59/2.87	3.41/2.73	3.89/3.12	1.92/1.57	2.94/2.33
SnowF° [37]	**1.08**/**0.83**	2.83/3.02	**1.73**/1.66	3.08/2.81	3.53/2.78	**2.46**/2.51	3.51/2.86	**1.52**/**1.30**	2.47/2.22
VE-PCN° [32]	8.80/5.39	6.72/5.95	2.91/2.37	7.87/6.57	10.94/8.53	8.76/7.22	6.33/4.99	3.75/2.73	7.01/5.47
IFNet◊ [3]	1.33/1.36	5.17/4.77	3.16/2.97	3.13/3.24	**1.74**/**1.66**	3.61/3.87	**2.26**/**2.41**	2.26/2.52	2.83/2.85
Front2Back_R5† [39]	1.44/1.33	3.14/2.59	1.94/1.92	**2.49**/**2.31**	2.48/2.30	2.53/**2.21**	3.02/3.02	1.88/1.49	**2.36**/**2.15**
Front2Back_GT† [39]	1.19/1.18	2.78/2.21	1.90/1.77	**2.00**/**1.86**	**1.69**/**1.77**	**2.13**/**2.00**	**2.03**/**2.26**	1.53/1.51	**1.91**/**1.82**
Ours† (Iteration 20)	**0.82**/**0.84**	**2.04**/**1.63**	**1.56**/**1.35**	**1.54**/**1.39**	**1.30**/**1.34**	**1.84**/**1.53**	**1.67**/**1.54**	**1.04**/**1.00**	**1.48**/**1.33**
Unaligned ISCNet	1.43/1.43	2.56/2.09	2.15/1.88	2.10/1.92	1.76/1.87	2.33/1.99	2.33/2.25	1.66/1.60	2.04/1.88
No-Pair ISCNet	0.88/1.02	2.28/1.99	1.90/1.69	1.75/1.47	2.12/1.83	2.07/1.76	2.11/1.54	1.41/1.54	1.82/1.61
No-Normal ISCNet	1.08/1.07	2.41/2.04	2.18/1.86	1.89/1.53	1.91/1.78	2.27/1.81	2.14/1.81	1.61/1.59	1.94/1.69
Iteration 4	0.89/0.83	2.23/1.94	1.74/1.52	1.67/1.54	1.43/1.42	2.03/1.79	1.70/1.58	1.15/1.06	1.61/1.46
Iteration 8	0.85/0.87	2.19/1.73	1.66/1.44	1.60/1.45	1.32/1.39	1.92/1.63	1.73/1.62	1.10/1.04	1.55/1.40
Iteration 12	0.84/0.87	2.16/1.72	1.60/1.38	1.57/1.41	1.28/1.37	1.88/1.55	1.70/1.58	1.06/1.04	1.51/1.36
Iteration 16	0.83/0.84	2.11/1.67	1.57/1.36	1.55/1.40	1.26/1.36	1.92/1.62	1.67/1.54	1.04/1.00	1.49/1.35

Table 1. Quantitative evaluation of ISCNet compared to prior work on the synthetic ShapeNet dataset (○ point cloud methods, ◊ implicit function methods, and † image-based methods) in terms of a mesh-to-mesh symmetric L_1 metric (MD) and the Chamfer Distance (CD) (**lowest**, **2nd lowest**, **3rd lowest** error color coding). In addition, we show the errors on three ablation variants of ISCNet (without symmetry plane alignment, without pair-wise inpainting, and without utilizing normal information), as well as the evolution of errors per iteration.

CD/MD ($\times 10^2$)	*Cabinet*	*Chair*	*Lamp*	*Sofa*	*Table*	*Average*
PCN° [41]	9.76/10.44	6.27/7.60	9.60/10.45	6.02/7.63	11.83/11.94	8.70/9.61
MSN° [15]	**5.71**/**5.17**	**5.86**/**5.40**	**5.74**/**4.77**	**4.95**/**4.01**	6.43/**6.86**	**5.74**/**5.24**
PoinTr° [40]	8.33/7.68	6.40/6.33	9.66/9.19	6.12/5.21	7.84/8.47	7.67/7.38
VRCNet° [16]	**5.05**/**3.98**	5.87/**5.56**	**5.95**/**5.03**	**4.45**/**3.64**	**6.00**/**6.60**	**5.47**/**4.96**
SnowF° [37]	8.66/7.89	6.47/6.43	8.99/8.87	5.63/4.91	7.22/8.75	7.39/7.37
VE-PCN° [32]	9.26/6.92	6.38/5.98	6.12/5.48	5.64/4.70	8.06/7.73	7.09/6.16
IFNet◊ [3]	12.62/7.09	**5.38**/6.60	8.95/8.20	5.05/4.21	**5.92**/7.13	7.59/6.64
Front2Back_GT† [39]	11.22/8.87	6.86/7.55	7.81/6.64	6.92/6.57	6.85/7.73	7.93/7.47
Ours † (Iteration 20)	**4.88**/**3.89**	**3.67**/**3.99**	**4.62**/**4.16**	**3.27**/**2.86**	**4.53**/**5.16**	**4.19**/**4.01**

Table 2. Quantitative evaluation of ISCNet compared to prior work on real-world ScanNet dataset. The notations and colors follow Table 1.

where $\mathcal{L}_{\text{K}}$, $\mathcal{L}_{\text{D}}$ and $\mathcal{L}_{\text{N}}$ measure the loss over: silhouette mask K, depth D, and normal N between the estimated $(\cdot_E)$ and reference $(\cdot_G)$ components. Φ_{t_2} is an indicator function on the L_1 distance between maps with a slack parameter t_2 set to 0.02, and X^{M_G} is the masking operator with a reference mask over a component $X \in \{N, D, C\}$. At this stage, we do not yet explicitly enforce symmetry to allow the inpainting network to converge to a good starting point for the second stage.

The second-stage loss is the sum of both first-stage losses plus an additional loss that explicitly enforces symmetry:

$$\mathcal{L} = \alpha\mathcal{L}^S + \beta\mathcal{L}^I + \gamma\mathcal{L}^J, \tag{26}$$

$$\mathcal{L}^J = \Sigma_{i=1}^{4}(\|D_i^{M_\text{G}} - \mathcal{F}(D_i') \odot M_\text{G}\|_1 + \|N_i^{M_\text{G}} - \mathcal{F}(N_i') \odot M_\text{G}\|_1), \tag{27}$$

where α, β, and γ are balancing weights, $\mathcal{F}$ is a 2D horizontal mirror operator, and X and X' the respective components from symmetric viewpoints.

4. Assessment

We implement ISCNet in PyTorch [10], and train and test ISCNet for eight categories (airplane, cabinet, car, chair, lamp, sofa, table, and vessel) from the ShapeNet dataset [2]. As in prior work [15, 41], we partition the dataset in 29,774 shapes for training and 1,200 for testing. We use the 26 viewpoints provided by VRCNet [16] to generate synthetic depth images and normal maps for training and 8 randomly generated viewpoints for testing. In addition, we also test ISCNet with 456 scanned objects from ScanNet [5], excluding the categories not present in ScanNet (*i.e.* airplanes, cars, and vessels). The corresponding reference shapes are provided by Scan2CAD [1]; our and all competing methods are only trained with ShapeNet.

To train ISCNet, we first pre-train the four submodules separately for 50 epochs: symmetry detection using $\mathcal{L}^{\text{S}}$ (Equation 18), mask generation using $\mathcal{L}_{\text{K}}$ (Equation 21), 2D normal inpainting using $\mathcal{L}_{\text{N}}$ (Equation 22), and 2D depth inpainting using $\mathcal{L}_{\text{D}}$ (Equation 23). Finally, we refine the complete network for an additional 10 epochs using $\mathcal{L}$ as loss and set the weight parameters α, β, and γ to 0.01, 0.05, and 1, respectively (Equation 26). While the

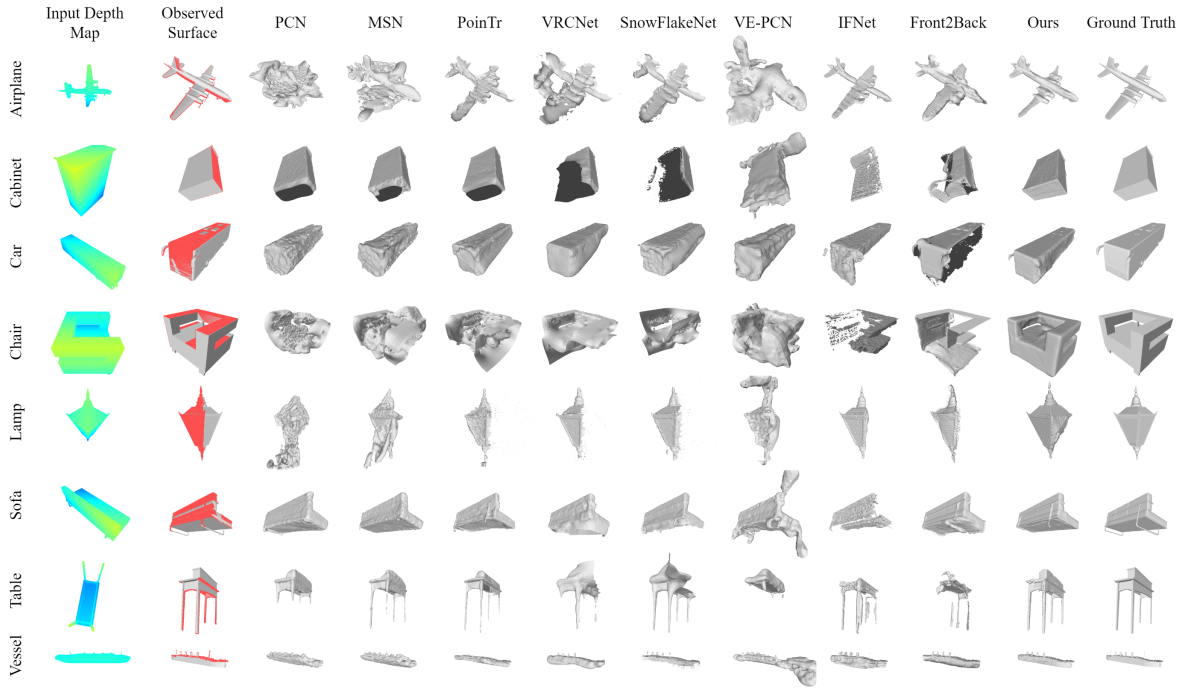


Figure 4. Visual comparison of shape completion quality of ISCNet versus prior work on the synthetic ShapeNet dataset; the observed areas marked in red (the second column).

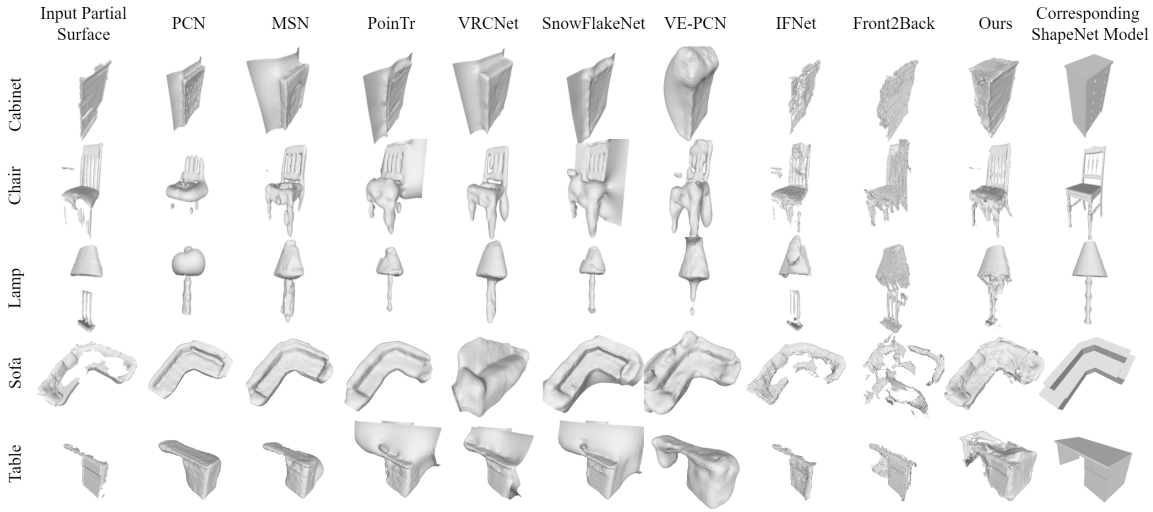


Figure 5. Visual comparison of shape completion quality of ISCNet versus prior work on real-world ScanNet dataset. Note that the sofa and table are *non-symmetric* objects.

training datasets are generated from 26 pre-defined viewpoints for the first training stage, the second stage (fine-tuning) uses randomly selected viewpoints instead of the viewpoints chosen by the RL agent. In the latter case, we generate the corresponding ground truth depth images and normal maps on the fly during the training process. As a consequence of this phased training strategy, the distribution of (RL selected) viewpoints on the testing dataset cover the whole sphere, thereby avoiding overfitting to the 26 pre-selected viewpoints (see the supplemental material for more details).

We train all networks with the Adam optimizer with a learning rate of $2e^{-4}$ for pretraining and $4e^{-5}$ for fine-tuning on a dual NVIDIA Gforce RTX 3090 setup. On this setup, for 20 iterations, ISCNet takes 81 seconds to complete a shape from a single depth map, where the image

rendering and point clouds merging take around 59 seconds.

For validation, we measure the reconstruction quality by a mesh-to-mesh symmetric L_1 distance (MD) [4] between the reference and completed meshes. Further, to quantify the reconstruction accuracy of surface details, we follow Yao *et al*. [39] and use the Chamfer Distance (CD) on 100K Poisson disk sampled [43] surface points from the reconstructed and reference meshes. Finally, following Zhou *et al*. [45] we quantify symmetry plane accuracy by comparing the cosine distance between symmetry plane normals.

4.1. Quantitative Evaluation

3D Shape Completion. We compare the performance of ISCNet to 8 competing methods subdivided in three categories: *Point-cloud completion methods* (PCN [41], MSN [15], PoinTr [40], VRCNet [16], ShowFlakeNet [37], VE-PCN [32]), *implicit-function-based methods* (IFNet [3]), and *image-based completion methods* (2 variants of Front2Back [39][1]). All competing methods are retrained on ShapeNet and tested with identical exemplars under the same conditions. The quantitative comparisons (using the CD and MD metrics) are summarized in Table 1 and Table 2 for ShapeNet and ScanNet, respectively. From these, we can see that ISCNet offers the best overall performance for the majority of categories on both synthetic and scanned data, followed by Front2Back [39]. SnowFlakeNet [37] performs better than ISCNet on the *plane* shapes with a marginal accuracy gain. Compared to Front2Back, we can see that ISCNet outperforms Front2Back with the *ground truth symmetry information*. For completeness, Table 3 compares ISCNet with Front2Back and IFNet for the remaining five classes from ShapeNet, without retraining or finetuning. Our approach still outperforms existing work except for the *rifle* class compared to IFNet.

($\times 10^2$)		Bench	Display	Speaker	Rifle	Phone	Average (*unseen-5*)	Average (*total-13*)
CD	F2B	2.37	3.19	3.40	1.72	3.13	2.76	2.24
	IFNet	1.81	3.73	6.35	**1.37**	3.34	3.32	3.02
	Ours	**1.44**	**2.47**	**3.09**	1.43	**2.10**	**2.11**	**1.72**
MD	F2B	2.19	2.97	3.17	1.59	2.97	2.58	2.11
	IFNet	1.77	3.65	6.24	**1.35**	3.28	3.26	3.01
	Ours	**1.41**	**2.45**	**3.01**	1.42	**2.07**	**2.07**	**1.61**

Table 3. Quantitative evaluation of ISCNet compared to Front2Back and IFNet for the remaining 5 classes of ShapeNet, *i.e*., *unseen-5* labels.

Figure 4 and 5 provide additional qualitative comparisons. Visually, ISCNet provides the 'cleanest' shape completion for all eight tasks. In particular, in the *plane* completion task, ISCNet is better able to retain high-frequency surface details, and in the *cabinet* and *lamp* completion tasks ISCNet is the only method to fully complete the surface without holes. In the case of the *chair* and *table* completion tasks, ISCNet more robustly identifies areas to be completed without erroneously merging thin features. Although our method is designed for symmetric objects, we also obtain good reconstruction quality on non-symmetric objects (*sofa* and *table* in Figure 5).

Symmetry Detection. To assess the accuracy of the symmetry plane estimation, we compare it against two state-of-the-art single-view based pose estimation and symmetry detection methods, RotationContinuity [44] and NeRD [45]. Figure 6a shows that ISCNet's second update of the symmetry plane offers a similar performance as NeRD. Furthermore, with increasing iterations, the accuracy of the estimated symmetry plane improves (*i.e*., compared to the first symmetry plane estimate[2], the occurrence of symmetry plane estimates with an error less than $5°$, improves from 32.86% to 56.96%), demonstrating the effectiveness of the iterative refinement for symmetry plane estimation.

4.2. Ablation Study

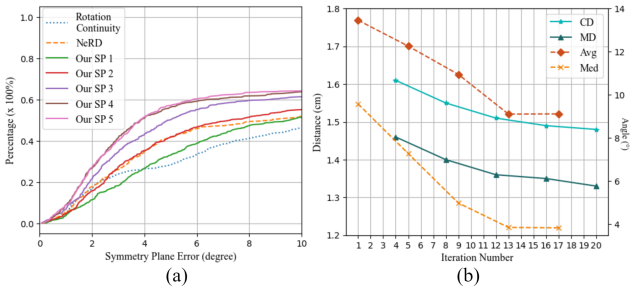
Importance of Symmetry. We validate the effectiveness of symmetry cues for shape completion in two ablation experiments. As a first experiment (Table 1, Unaligned ISCNet), we replace the estimated transformation that aligns the symmetry plane with the X-Y plane by a random rotation matrix and a perturbed center (adding normal distributed noise with $\sigma = 0.1$). Consequently, ISCNet cannot leverage any symmetry cues. In a second experiment (Table 1, No-Pair ISCNet), we process each of the reflection viewpoints separately. From Table 1, we can see that the *unaligned ISCNet* suffers a significant performance degradation; CD increases by 0.56 on average, and MD increases 0.55 on average. By aligning the shape to the symmetry plane, *no-pair ISCNet* is able to leverage symmetry implicitly (by learning to copy from the negative Z coordinate). However, compared to the full ISCNet, the no-pair ISCNet still sees a performance drop (CD and MD are increased by 0.34 and 0.28, respectively). We refer to the supplementary material for a qualitative comparison of excluding explicit leveraging of the symmetry cues.

Role of Normal Inpainting. To demonstrate the importance of performing normal inpainting, we eliminate the inpainted normal as input from the depth inpainting submodule (Table 1, No-Normal ISCNet). Without normal cues, CD increases by 0.46 and MD by 0.36. We refer to the supplementary material for a qualitative visualization of the impact of omitting the normal inpainting step.

[1]The information in [39] was insufficient to replicate the symmetry plane estimation module. Therefore, we include 2 variants, trained and tested with the ground truth symmetry plane (Front2Back_GT), and with the reference symmetry plane normal perturbed within $[-5°, 5°]$ (Front2Back_R5).

[2]The accuracy of first symmetry estimate is similar to that of DenseFusion [30], as our symmetry estimation is based this method.

Impact of Iterative Refinement. To demonstrate that both symmetry plane estimation as well as shape completion improve with additional iterations, we compare the MD and CD errors per iteration (Table 1) as well as the symmetry plane normal error progression per iteration (Figure 6b). In both cases, we can see that the error improves per iteration and the improvement diminishes per subsequent iteration (justifying a 20 iteration stopping criterion).



(a) (b)

Figure 6. Impact of the number of iterations on (a) the accuracy of the symmetry plane estimation, and (b) the shape reconstruction CD/MD accuracy and average and median symmetry plane estimation errors. 'Our SP n' refers to ISCNet's n-th predicted symmetry plane.

Limitations The main bottleneck in ISCNet is the image rendering and the 3D point cloud processing step in each iteration. Furthermore, ISCNet is only able to leverage a single global symmetry plane, and thus its performance is suboptimal when multiple symmetries are present.

5. Conclusion

We present ISCNet, a single-depth image joint symmetry plane estimation and shape completion method that explicitly leverages symmetry cues. ISCNet alternates between estimating the symmetry plane and completing the shape to enhance the accuracy of both components. Our method is informed by two key insights: (1) shape completion and symmetry plane estimation are mutually beneficial tasks, and (2) by selecting symmetric virtual viewpoints, we can explicitly inform the inpainting module on symmetry relations. Our experiments and ablation studies demonstrate that ISCNet can yield more accurate shape completions from a single-depth image than competing methods.

Acknowledgments

We thank the anonymous reviewers for the insightful and constructive comments. This work was supported in part by National Key Research and Development Program of China (2022ZD0210500, 2021ZD0111902), National Natural Science Foundation of China (61972067, U19B2039, U21B2038), the Distinguished Young Scholars Funding of Dalian (No. 2022RJ01). Pieter Peers was supported in part by NSF grant IIS-1909028. Felix Heide was supported by an NSF CAREER Award (2047359), a Packard Foundation Fellowship, a Sloan Research Fellowship, a Sony Young Faculty Award, a Project X Innovation Award, and an Amazon Science Research Award.

References

[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 2614–2623, 2019. 6

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 6

[3] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6970–6981, 2020. 1, 6, 8

[4] Paolo Cignoni, Claudio Rocchini, and Roberto Scopigno. Metro: measuring error on simplified surfaces. In *Computer Graphics Forum*, volume 17, pages 167–174, 1998. 8

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 6

[6] Shivam Duggal and Deepak Pathak. Topologically-aware deformation fields for single-view 3d reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1536–1546, 2022. 2

[7] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 605–613, 2017. 2

[8] Boqing Gong, Jianzhuang Liu, Xiaogang Wang, and Xiaoou Tang. Learning semantic signatures for 3d object retrieval. *IEEE Trans. Multimedia*, 15(2):369–377, 2012. 1, 2

[9] Xiaoguang Han, Zhaoxuan Zhang, Dong Du, Mingdai Yang, Jingming Yu, Pan Pan, Xin Yang, Ligang Liu, Zixiang Xiong, and Shuguang Cui. Deep reinforcement learning of volume-guided progressive view inpainting for 3d point scene completion from a single depth image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 234–243, 2019. 5

[10] Sagar Imambi, Kolla Bhanu Prakash, and GR Kanagachidambaresan. Pytorch. In *Programming with TensorFlow*, pages 87–104. 2021. 6

[11] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 5

[12] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Symmetry descriptors and 3d shape matching. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH*

symposium on Geometry processing, pages 115–123, 2004. 1, 2

[13] Wai Ho Li and Lindsay Kleeman. Segmentation and modeling of visually symmetric objects by robot actions. *The International Journal of Robotics Research*, 30(9):1124–1142, 2011. 1, 2

[14] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Eur. Conf. Comput. Vis.*, pages 85–100, 2018. 5

[15] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *AAAI*, volume 34, pages 11596–11603, 2020. 2, 4, 6, 8

[16] Liang Pan, Xinyi Chen, Zhongang Cai, Junzhe Zhang, Haiyu Zhao, Shuai Yi, and Ziwei Liu. Variational relational point completion network. In *Int. Conf. Comput. Vis.*, pages 8524–8533, 2021. 2, 5, 6, 8

[17] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas Funkhouser. A planar-reflective symmetry transform for 3d shapes. In *ACM SIGGRAPH 2006 Papers*, pages 549–559. 2006. 1

[18] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 652–660, 2017. 4

[19] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015. 4

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 3, 5

[21] Yifei Shi, Junwen Huang, Hongjia Zhang, Xin Xu, Szymon Rusinkiewicz, and Kai Xu. Symmetrynet: learning to predict reflectional and rotational symmetries of 3d shapes from single-view rgb-d images. *ACM Trans. Graph.*, 39(6):1–14, 2020. 1, 2

[22] Yifei Shi, Xin Xu, Junhua Xi, Xiaochang Hu, Dewen Hu, and Kai Xu. Learning to detect 3d symmetry from single-view rgb-d images with weak supervision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022. 1, 2

[23] Ivan Sipiran, Robert Gregor, and Tobias Schreck. Approximate symmetry detection in partial 3d meshes. In *Computer Graphics Forum*, volume 33, pages 131–140. Wiley Online Library, 2014. 1, 2

[24] Pablo Speciale, Martin R Oswald, Andrea Cohen, and Marc Pollefeys. A symmetry prior for convex variational 3d reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 313–328. Springer, 2016. 1, 2

[25] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2974–2983, 2018. 2

[26] Junshu Tang, Zhijun Gong, Ran Yi, Yuan Xie, and Lizhuang Ma. Lake-net: Topology-aware point cloud completion by localizing aligned keypoints. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1726–1735, 2022. 2

[27] Demetri Terzopoulos, Andrew Witkin, and Michael Kass. Symmetry-seeking models and 3d object reconstruction. *Int. J. Comput. Vis.*, 1(3):211–221, 1988. 2

[28] Sebastian Thrun and Ben Wegbreit. Shape from symmetry. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1824–1831. IEEE, 2005. 1, 2

[29] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2626–2634, 2017. 2

[30] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3343–3352, 2019. 1, 2, 3, 8

[31] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Eur. Conf. Comput. Vis.*, pages 52–67, 2018. 2

[32] Xiaogang Wang, Marcelo H Ang, and Gim Hee Lee. Voxel-based network for shape completion by leveraging edge generation. In *Int. Conf. Comput. Vis.*, pages 13189–13198, 2021. 6, 8

[33] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 790–799, 2020. 1

[34] Boyan Wei, Xianfeng Ye, Chengjiang Long, Zhenjun Du, Bangyu Li, Baocai Yin, and Xin Yang. Discriminative active learning for robotic grasping in cluttered scene. *IEEE Robotics and Automation Letters*, 8(3):1858–1865, 2023. 1

[35] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. *Adv. Neural Inform. Process. Syst.*, 2017. 2

[36] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2020. 2

[37] Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In *Int. Conf. Comput. Vis.*, pages 5499–5509, 2021. 2, 6, 8

[38] Xin Yang, Yuanbo Wang, Yaru Wang, Baocai Yin, Qiang Zhang, Xiaopeng Wei, and Hongbo Fu. Active object reconstruction using a guided view planner. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 4965–4971, 2018. 3

[39] Yuan Yao, Nico Schertler, Enrique Rosales, Helge Rhodin, Leonid Sigal, and Alla Sheffer. Front2back: Single view 3d

shape reconstruction via front to back prediction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 531–540, 2020. 1, 2, 6, 8

[40] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Int. Conf. Comput. Vis.*, pages 12498–12507, 2021. 1, 2, 6, 8

[41] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *Int. Conf. 3D Vis.*, pages 728–737, 2018. 2, 6, 8

[42] Zhaoxuan Zhang, Xiaoguang Han, Bo Dong, Tong Li, Baocai Yin, and Xin Yang. Point cloud scene completion with joint color and semantic estimation from single rgb-d image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3

[43] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 8

[44] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5745–5753, 2019. 2, 8

[45] Yichao Zhou, Shichen Liu, and Yi Ma. Nerd: neural 3d reflection symmetry detector. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 15940–15949, 2021. 1, 2, 8