PIE-NET: Parametric Inference of Point Cloud Edges (Supplementary Material)

Anonymous Author(s)
Affiliation
Address
email

6 Closed curve proposal generation

Our closed curve proposal module is inspired by [45]. In particular, we first identify the subset of points belonging to closed curves via feature *clustering*, and then *fit* a closed curve to each proposed cluster, while simultaneously estimating the *confidence* of the fit; see Figure 10. Here we use the edge/corner classification described in Section 3.1 of the main paper. Note our method currently only handles curves with a *circular* profile, but it can be easily extended to other types of closed curves.

Clustering. We train an equivariant PointNet++ network to produce a point-wise feature $F(\cdot)$ for each of the M input edge points. Based on such features, we then create a similarity matrix $S{\in}\mathbb{R}^{M\times M}$, where $S_{ij}{=}\|F(p_i) - F(p_j)\|_2$. We can then interpret each of the M rows of S as a *proposal*, and consider the set $C_m{=}\{j \text{ s.t. } S_{m,j}{<}\bar{S}\}$ as the edge points of the m-th proposal. $\bar{S}$ is a threshold to filter out the points attaining very different feature scores (i.e., they probably do not belong to the same curve). Potentially redundant proposals are dealt with in the *selection* phase of our pipeline, as described in Section 7.

Fitting. We take each proposal C_m, and regress the parameters β of the corresponding curve, as well as its confidence γ. We parameterize each circle proposal via three points $\beta{=}\{p_a{+}\Delta_a, p_b{+}\Delta_b, p_c{+}\Delta_c\}$. We obtain $\{p_a, p_b, p_c\}$ by furthest point sampling in C_m initialized with $p_a{=}p_m$. We then train a PointNet architecture with two fully-connected heads. The first head regresses the offsets $\{\Delta_a, \Delta_b, \Delta_c\}$, while the second head predicts γ.

Losses. We train our network to predict similarity matrices S given ground truth $\hat{S}$, confidences $\Gamma{=}\{\gamma_n\}$ given ground truth $\hat{\Gamma}$, and a collection of points sampling the ground truth curve:

$$\mathcal{L}_{\text{closed}} = \mathcal{L}_{\text{sim}}(S, \hat{S}) + \mathcal{L}_{\text{score}}(\Gamma, \hat{\Gamma}) + \mathcal{L}_{\text{para}}(\beta). \tag{5}$$

Given that $\hat{S}_{ij}{=}0$ if points p_i and p_j belong to the same ground truth curve and $\hat{S}_{ij}{=}1$ otherwise, we supervise for similarity via:

$$\mathcal{L}_{\text{sim}} = \sum_{ij} S_{ij}, \tag{6}$$

where

$$S_{ij} = \begin{cases} \|F(p_i) - F(p_j)\|_2, & \hat{S}_{ij} = 1 \\ \max\{0, K - \|F(p_i) - F(p_j)\|_2\}, & \hat{S}_{ij} = 0 \end{cases}$$

where F is point-wise feature computed with PointNet++. K controls the dissimilarity between elements in different parts, which is set to $K{=}100$ in our experiments. For $\mathcal{L}_{\text{score}}(\cdot)$ we employ L_2 loss, where $\hat{\Gamma}$ is the segmentation confidence. Positive training examples come from seed points belonging to ground truth closed curves, and their IoU with ground truth segmentations is larger than 0.5. Negative training examples are those with IoU smaller than 0.5. For parameter regression, we minimize

$$\mathcal{L}_{\text{para}} = \hat{T}_{\text{circle}} \cdot \mathcal{L}_{\text{circle}}(\beta), \tag{7}$$

where $\hat{T}_{\text{circle}}$ is the ground truth one-hot labels, and $\mathcal{L}_{\text{circle}}(\cdot)$ is the Chamfer distance between the curve represented by β and the ground truth. In particular, we first compute the circle according to

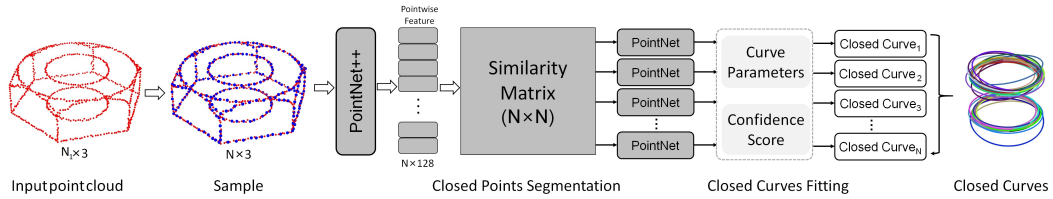

Figure 10: **Closed curve proposal** – We first identify the subset of points belonging to closed curves via feature-based clustering and then fit a closed curve to each cluster. The network outputs both curve parameters and confidence scores.

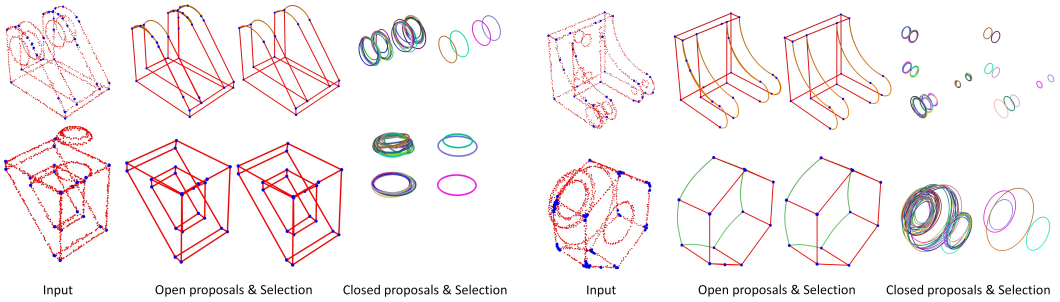

Figure 11: **Proposal selection** – The curves generated by open/closed proposals, and the ones that were accepted by our selection process.

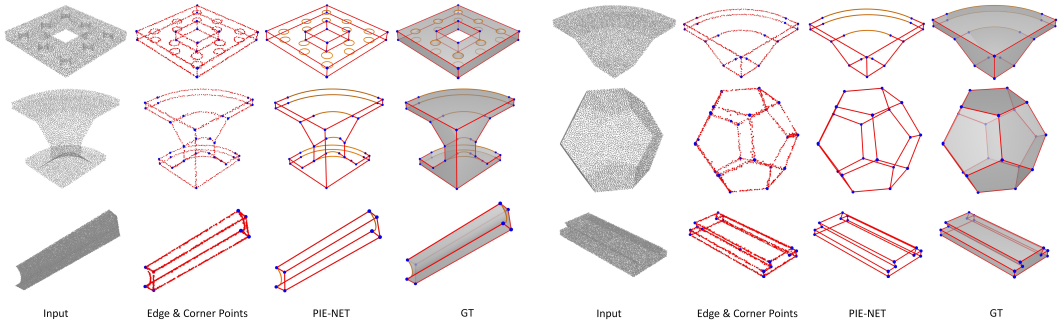

Figure 12: Results on edge and corner detection and parametric curve inference by PIE-NET.

the estimated β, and then sample it by points. We then compute the Chamfer distance between the estimated circle and its corresponding point-sampled ground-truth.

7 Curve proposal selection

Similar to proposal-based object detection for images [44], the final stage of our algorithm is a non-differentiable process for redundant/invalid proposal filtering; see Figure 11. We adopt slightly different solutions for *open* and *closed* curves.

Open curve selection. Given the segmentations, i.e., a set of points associated with a curve (see Figure 5 in the main paper) corresponding with two proposals, we first measure overlap via $O(A,B) = max\{I(A,B)/A, I(A,B)/B\}$, where $I(A,B)$ is the cardinality of the intersection between the sets. We then merge the two candidates, if $O(A,B){>}\tau_o$ and retain the curve with larger cardinality, where we use $\tau_o{=}0.8$ as determined by hyper-parameter tuning.

Closed curve selection. The similarity matrix produces a closed-curve proposal for *each* of its N rows. Even after discarding proposals with confidence score $\gamma_n{<}\tau_\gamma$, many are *non-closed* curves, or represent the *same* closed curve; see Figure 11. We perform agglomerative clustering for proposals when $\text{IoU}(A,B){>}\tau_{\text{iou}}$, and retain the proposal in the cluster with the highest confidence. We use

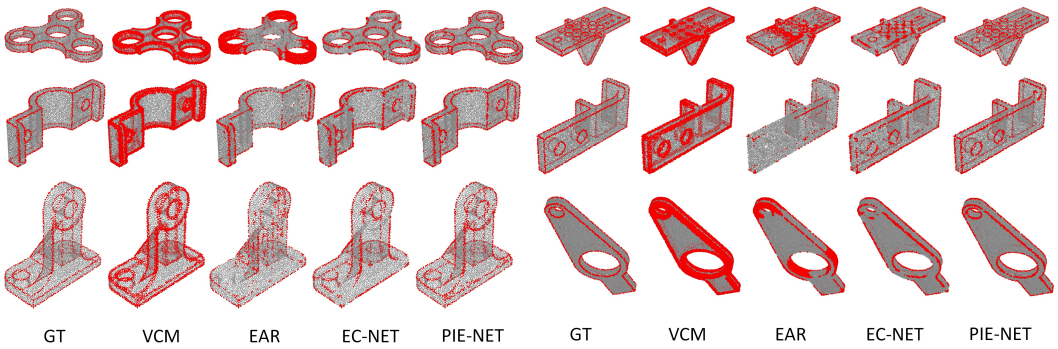

Figure 13: **Comparisons to state-of-the-art methods** – Qualitative comparisons against edge detection techniques – VCM [50], EAR [51], and EC-Net [10].

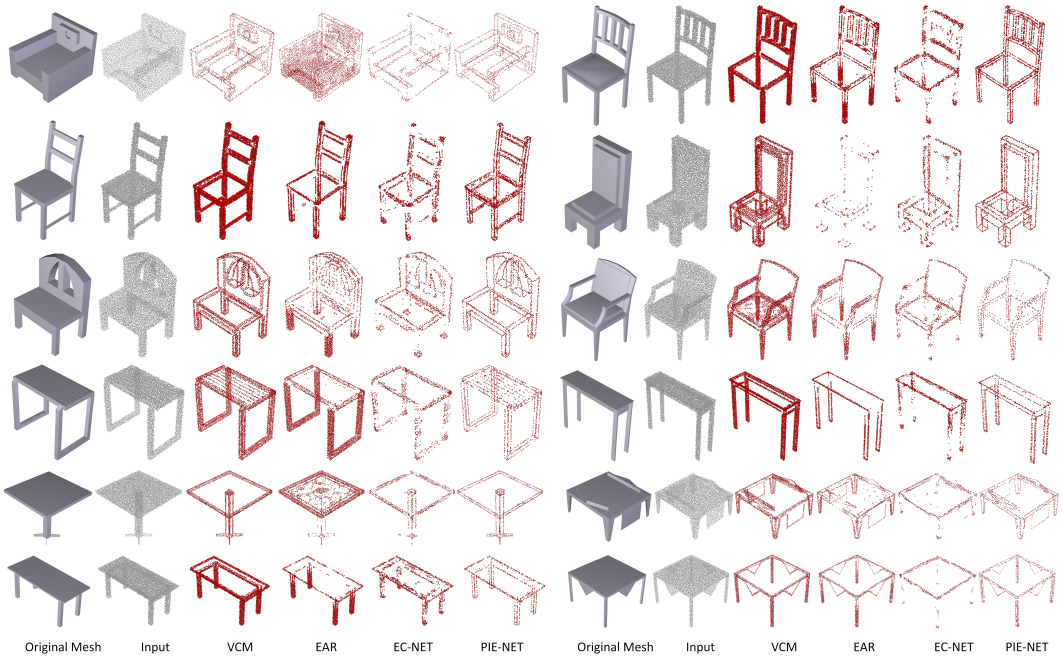

Figure 14: **Generalization to novel object categories** – While PIE-NET is trained on CAD models of mechanical assemblies from the ABC dataset [46], our edge detector is immediately applicable to 3D point clouds of general 3D objects and consistently outperforms VCM, EAR, and EC-Net.

τ_γ=0.6 and τ_{iou}=0.6 for all our experiments. Finally, for each closed segment, we select the best matching closed curve. Specifically, we use the Chamfer Distance to measure the matching score.

8 Backbone architectures

8.1 Classification

We use PointNet++ [47] as the feature backbone network. Following the same notations in PointNet++, SA(K,r,n,[l_1,l_2,...,l_d]) is a set abstract layer with K local regions of balls with radius r. n points are sampled for each local region. The SA layer uses d 1*1 conv layers with output channels l_1,l_2,...,l_d respectively. FP(l_1,l_2,...,l_d) is a feature propagation (FP) layer with d 1*1 conv layers, whose output channels are l_1,l_2,...,l_d. In our task, we use four set abstract (SA) layers and four feature propagation (FP) layers. The parameters of set abstract (SA) and feature propagation (FP) layers are as follows: SA(4096, 0.05, 32, [32,32,64]), SA(2048, 0.1, 32, [64,64,128]), SA(1024, 0.2, 32, [18,128,256]), SA(512, 0.4, 32, [256,256,512]), FP(256,256), FP(256,256), FP(256,128),

FP(128,128,128). Following PointNet++, we also added four separate fully-connected layers in order to predict the following tasks: edge point classification $\hat{T_e}$, point-to-curve distance vector regression $\hat{D_e}$, corner point classification $\hat{T_c}$ and corner point residual vector regression $\hat{D_c}$.

8.2 Open curve proposal

For each corner pair, we use a PointNet++ as the backbone with two multi-layer perceptrons (MLP) layers. MLP($[l_1, ..., l_n]$) indicates several MLPs with output channels $l_1, ..., l_n$. The parameters of these two MLPs are set to [128,256, 512], [256, 256]. Following PointNet, we added three separate fully-connected layers in order to predict the following tasks: curve segmentation, curve type classification, and curve parameters estimation.

8.3 Clustering

We use a PointNet++ as the clustering backbone. In our task, we use four set abstract (SA) layers and four feature propagation (FP) layers. The parameters of set abstract (SA) and feature propagation (FP) layers are as follows: SA(512, 0.1, 32, [32,32,64]), SA(256, 0.2, 32, [64,64,128]), SA(128, 0.4, 32, [18,128,256]), SA(64, 0.8, 32, [256,256,512]), FP(256,256), FP(256,256), FP(256,128), FP(128,128,128). Following PointNet, we added one MLP layer in order to predict similarity matrix.

8.4 Closed curve fitting

In this task, we use a PointNet as the backbone with two MLP layers. The parameters of the two MLPs are set to [128,256, 512], [256, 256]. Following PointNet, we added two separate fully-connected layers to predict the curve parameters and the confidence scores.