PQDT: Pseudo-Query Dual Transformer for Robust Point Cloud Restoration

Supplementary Material

In this supplementary material, we provide additional content to complement the main paper. Sec. A presents the full implementation details of PQDT, including training setups, network components. Sec. B gives the definition of the evaluation metrics. Sec. C describes the datasets used in our experiments and the pre-processing strategies applied. Sec. E reports extended ablation studies that analyze the contribution of each module in our design. Sec. F provides additional quantitative evaluation results across multiple benchmarks. Finally, Sec. G includes more qualitative visualizations to further illustrate the effectiveness of our approach.

A. Implementation Details

Training Setups. We implement PQDT using PyTorch [35] and train the network with the Adam optimizer [23]. The initial learning rate is set to 0.0001. We adopt a cosine annealing learning rate scheduler [30] with a minimum learning rate of 0.00001, along with a linear warm-up from 0.00001 to 0.0001 over the first 10 epochs. For the ShapeNet-55/34 and ShapeNet-Deform datasets, we train the model for 300 epochs with a batch size of 32. For smaller-scale datasets, ShapeNetCar-Occ and PFS, the batch sizes are set to 16 and 8, respectively, and the models are trained for 200 epochs. For all ShapeNet-family datasets, the network takes 2,048 points as input and predicts 8,192 points as output. On the PFS dataset, we instead sample 8,192 points from the geometry as inputs and predict 8,192 points as output. We train the model with a single Nvidia A100 80GB GPU, and all the models are tested with the batch size of 1 and same random seed.

Feature Extraction. We employ a modified DGCNN [52] to extract features from the input point cloud. The network expands feature dimensionality while progressively downsampling points, following the architecture:

$\text{Linear}(C_{in}=3, C_{out}=8) \rightarrow \text{EC}(C_{in}=8, C_{out}=64, K=16, M=512) \rightarrow \mathcal{E}(C=64, H=1, L=2) \rightarrow \text{EC}(C_{in}=64, C_{out}=256, K=16, M=128) \rightarrow \mathcal{E}(C=256, H=4, L=2)$,

where $C_{\text{in}}, C_{\text{out}}$ denote input and output feature channels; EC is an EdgeConv block with K nearest neighbors and farthest point sampling size M; and $\mathcal{E}$ is a lightweight transformer encoder with feature dimension C, number of heads H, and number of layers L. The resulting coarse level features are: $\mathcal{F}^c_{src} \in \mathbb{R}^{128\times256}, \mathcal{P}^c_{src} \in \mathbb{R}^{128\times3}$.

Transformer. As illustrated in Fig. 2, the dual transformer module, comprising geometric-embedding transformer encoders and decoders (GEE/GED), denoted $\text{M}_{E_{1,2}}$ and $\text{M}_{D_{1,2}}$, processes the coarse features and coordinates as follows:

First-Stage Encoding: $\text{M}_{E_1}(C=384, H=6, L=4) \rightarrow \mathcal{F}_1 \rightarrow \text{MaxPool} \rightarrow f_{g_1}$.

First-Stage Decoding: $\text{Agg}(\text{FPS}(\mathcal{P}_{sph}(R=0.8), M=384), f_{g_1}) \rightarrow \text{MLP}(C_{in}=3+1024, C_{out}=384) \rightarrow \text{M}_{D_1}(C=384, H=6, L=4, \mathcal{V}=\mathcal{F}_1) \rightarrow \mathcal{F}_{pq'} \rightarrow \text{MLP}(C_{in}=384, C_{out}=3) \rightarrow \mathcal{P}_{pq'}$. $\text{S}_1(N_{in}=384, N_{pad}=128, N_{out}=384, \mathcal{X}_{in}=\mathcal{F}_{pq'}, \mathcal{P}_{in}=\mathcal{P}_{pq'}) \rightarrow \mathcal{F}_{pq}, \mathcal{P}_{pq}$.

Second-Stage Encoding: $\text{M}_{E_2}(C=384, H=6, L=4) \rightarrow \mathcal{F}_2 \rightarrow \text{MaxPool} \rightarrow f_{g_2}$. $\text{S}_2(N_{in}=384, N_{pad}=256, N_{out}=512, \mathcal{X}_{in}=\mathcal{F}_2, \mathcal{P}_{in}=\mathcal{P}_{pq}) \rightarrow \mathcal{V}, \mathcal{P}_Q$.

Second-Stage Decoding: $\text{Agg}(\mathcal{P}_Q, f_{g_1}, f_{g_2}) \rightarrow \text{MLP}(C_{\in}=3+1024+1024, C_{out}=384) \rightarrow \mathcal{Q} \rightarrow \text{M}_{D_2}(C=384, H=6, L=8, \mathcal{V}=\mathcal{V}) \rightarrow \text{MaxPool} \rightarrow f_{g_q}$.

Output Aggregation: $\text{Agg}(\mathcal{P}_Q, \mathcal{Q}, f_{g_q}) \rightarrow \text{MLP}(C_{in}=3+384+1024, C_{out}=384) \rightarrow \mathcal{H}$.

Here, Agg denotes feature alignment and concatenation. $\text{S}_{1,2}$ denote dynamic query selection modules with input size N_{in}, padding size N_{pad}, and output size N_{out}. $\mathcal{P}_{sph}$ is a point set sampled from a sphere of radius R. Features preceding max pooling are projected to 1024 channels. The final coarse prediction outputs are: $\mathcal{F}^c_{pred} = \mathcal{H} \in \mathbb{R}^{512\times384}, \mathcal{P}^c_{pred} = \mathcal{P}_Q \in \mathbb{R}^{512\times3}$.

Geometric-Embedding. We follow the calculation of distance and angular embedding from [40] to form our sparse geometric-embedding (SGE):

Pair-wise Distance Embedding: By applying the sinusoidal encoding [49] to the Euclidean distance $\rho_{i,j} = \|P_i - P_j\|_2$ between points P_i and P_j, the distance embedding is defined as:

$$\begin{cases} r^D_{i,j,2k} = \sin\left(\dfrac{\rho_{i,j}/\sigma_d}{10000^{2k/C_e}}\right), \\ r^D_{i,j,2k+1} = \cos\left(\dfrac{\rho_{i,j}/\sigma_d}{10000^{2k/C_e}}\right), \end{cases} \tag{14}$$

where $\sigma_d = 0.2$ is the temperature which is used to tune the sensitivity on distance variation and we use same value as in the original work. C_e is the channel size of embedding.

Triplet-wise Angular Embedding: The angular embedding for a point P_i is computed using its k nearest neighbors $\mathcal{K}_i$. For each neighbor $P_x \in \mathcal{K}_i$, we evaluate the angle $\alpha^x_{i,j} = \angle(\Delta_{x,i}, \Delta_{j,i})$, where $\Delta_{i,j} = P_i - P_j$. Applying a sinusoidal positional encoding to $\alpha^x_{i,j}$, we obtain the angular

embedding:

$$\begin{cases} r^A_{i,j,x,2l} = \sin\left(\dfrac{\alpha^x_{i,j}/\sigma_a}{10000^{2l/C_e}}\right), \\ r^A_{i,j,x,2l+1} = \cos\left(\dfrac{\alpha^x_{i,j}/\sigma_a}{10000^{2l/C_e}}\right), \end{cases} \tag{15}$$

where $\sigma_\alpha = 15$ is the temperature which control the sensitivity on angular variations. Then we assemble these two types of embeddings can compute SGE using Eqs. (5) and (6).

Spherical Initialization. We adopt a DETR-like decoder [3] initialized with a set of learnable spherical queries (see Fig. 8). Specifically, we empirically sample 384 queries on the surface of a fixed sphere with radius 0.8, while all input point clouds are normalized to lie within the unit sphere. This design bounds the displacement between the initial queries and the target geometry, enabling the network to deform the spherical points toward missing regions via cross-attention. Extremely large missing regions may complicate global scale estimation and introduce potential bias, but such cases are uncommon in standard benchmarks. Some queries may not correspond to meaningful regions of the final geometry; however, the attention mechanism naturally suppresses such queries during decoding. The resulting redundancy increases flexibility in modeling diverse geometric structures without introducing noticeable computational overhead. Overall, the spherical initialization provides a stable geometric prior that facilitates robust and consistent shape completion.

Gradient approximation for Dynamic Query Selection. In Sec. 3, the query selection step involves Gumbel-perturbed Top-k sampling followed by hard indexing, which makes the operation non-differentiable. Ideally, training would minimize the expected reconstruction loss over the stochastic selection process:

$$\mathcal{L}(\theta) = \mathbb{E}_{S\sim p_\theta(S)}\left[\ell\big(f_\theta(X,S)\big)\right], \tag{16}$$

where X denotes the point features, S denotes the selected candidate set, $p_\theta(S)$ is the selection distribution induced by the scoring network, f_θ denotes the downstream decoding function, and ℓ is the reconstruction loss. In principle, unbiased gradients of this objective require score-function estimators (e.g., REINFORCE [55]), which propagate gradients through the sampling probabilities but typically exhibit high variance and unstable optimization. Instead, we adopt a straight-through estimator that ignores gradients through the sampling distribution and backpropagates only through the selected feature paths [1]. While this yields a biased gradient estimator, it significantly reduces variance and leads to stable training in practice. The injected Gumbel noise promotes stochastic exploration so that different candidates can be selected across iterations, partially compensating for the biased gradient approximation. As the noise scale is annealed during training, the selection gradually becomes deterministic while retaining the benefits of early-stage exploration.

Upsampling. The coarse predictions $\mathcal{F}^c_{pred}$ and $\mathcal{P}^c_{pred}$ are fed into a hierarchical upsampling transformer [75] to progressively refine the point cloud and recover fine-grained geometric details. We employ three upsampling stages, each implemented as UpTrans($C_{in,out} = 384, C_d = 64, K = 16, U$), where the upsampling rates are $U \in \{1, 4, 4\}$. Here, C_{in} denotes both the input and output feature dimensions, C_d is the latent feature dimension of the upsampling transformer, and K is the number of nearest neighbors used in the subtraction-based attention mechanism. Starting from 512 coarse points, the hierarchical upsampling produces the final prediction $\mathcal{P}^f_{pred} \in \mathbb{R}^{8192\times 3}$.

B. Metrics Details

Chamfer Distance. We use the average Chamfer Distance introduces by [9] to measures the discrepancy between the predicted point cloud $\mathcal{P}$ and the ground truth $\mathcal{G}$ at the point set level. For each prediction, the Chamfer Distance (CD) is calculated as:

$$CD(\mathcal{P},\mathcal{G}) = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}\min_{g\in\mathcal{G}}\|p-g\| \;+\; \frac{1}{|\mathcal{G}|}\sum_{g\in\mathcal{G}}\min_{p\in\mathcal{P}}\|g-p\|. \tag{17}$$

We use the L1-norm of the CD ($\text{CD}_{\ell 1}$) in the loss function and L2-norm ($\text{CD}_{\ell 2}$) as evaluation metric.

F-Score. Following the experiment setups of previous works, we also use the F-Score [45] as an extra metric for the evaluation, which is define as:

$$\text{F-Score}(d) = \frac{2\,P(d)\,R(d)}{P(d)+R(d)}, \tag{18}$$

where $P(d)$ and $R(d)$ denote the precision and recall with the threshold of distance d, respectively.

$$P(d) = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}\left[\min_{g\in\mathcal{G}}\|p-g\| < d\right], \tag{19}$$

$$R(d) = \frac{1}{|\mathcal{G}|}\sum_{g\in\mathcal{G}}\left[\min_{p\in\mathcal{P}}\|g-p\| < d\right]. \tag{20}$$

We set $d = 0.01$ for evaluation across all datasets, and additionally use $d = 0.005$ for the PFS dataset to better capture small discrepancies between the prediction and the ground truth.

C. Dataset Details

ShapeNet-55/34. The ShapeNet-55 dataset contains 55 object categories with 41,952 training samples and 10,518 testing samples. ShapeNet-34 is designed to evaluate model

generalization on unseen categories, consisting of 46,765 training samples from 34 categories. The test set includes 3,400 shapes from 34 seen categories and 2,305 shapes from 21 unseen categories. Following [4, 67, 75], we generate three levels of incompleteness (simple, moderate, and hard) by randomly preserving 75%, 50%, and 25% of the original points, respectively. We also evaluate the five most frequent categories (Table, Chair, Airplane, Car, and Sofa) and the five least frequent categories (Birdhouse, Bag, Remote, Keyboard, and Rocket), in addition to the average over all 55 categories at each incompleteness level.

ShapeNet-Deform. To enrich geometric diversity and improve robustness to non-rigid variations, we apply a 3D Gabor noise [26] deformation to all objects of ShapeNet-55 and keep the same partial points generation method from the ground truth. Gabor noise produces smooth, directionally-coherent perturbations and is widely used to synthesize natural stochastic patterns. For an input point cloud $\mathbf{X} \in \mathbb{R}^{N\times 3}$, the deformation is generated by aggregating multiple randomly sampled 3D Gabor kernels.

Gabor Kernel: A single 3D Gabor kernel is defined over a point $\mathbf{x} \in \mathbb{R}^3$ as:

$$g(\mathbf{x}) = \exp\left(-\tfrac{1}{2}\left(\tfrac{\mathbf{d}^\top \mathbf{x}}{\sigma}\right)^2\right) \times \cos\left(2\pi f\, \mathbf{d}^\top \mathbf{x} + \phi\right), \quad (21)$$

where $\mathbf{d} \in \mathbb{R}^3$ is a unit direction vector, f is the kernel frequency, σ is the Gaussian bandwidth, and ϕ is the phase shift. Given a batched point cloud $\mathbf{X} \in \mathbb{R}^{B\times N\times 3}$, we compute a batch-wise projection

$$s = \mathbf{d}^\top \mathbf{X} \in \mathbb{R}^{B\times N}, \quad (22)$$

and evaluate the Gaussian and sinusoidal components as implemented in our batched kernel:

$$g(\mathbf{X}) = \exp\left(-\tfrac{1}{2}(s/\sigma)^2\right) \odot \cos(2\pi f s + \phi). \quad (23)$$

3D Noise Aggregation: For each of the three axes, we sample K random kernels and average them:

$$\mathbf{N}(\mathbf{X})_{:,:,a} = \frac{1}{K}\sum_{k=1}^{K} g(\mathbf{X} - \mathbf{o}_k; \mathbf{d}_k, f, \sigma, \phi_k), \quad (24)$$

where $\mathbf{o}_k$ is a random spatial offset. This yields a full 3D noise field $\mathbf{N} \in \mathbb{R}^{B\times N\times 3}$.

Final Deformation: The noisy point cloud is obtained via:

$$\tilde{\mathbf{X}} = \mathbf{X} + \alpha\, \mathbf{N}(\mathbf{X}), \quad (25)$$

where α controls the deformation magnitude.

To enhance geometric variability, all Gabor noise parameters are randomly sampled from specified ranges during training. For evaluation, we instead fix the parameters to the midpoint of their ranges and use a fixed random seed to generate deterministic noise. Our setups are given in Tab. 4. The dataset is available via https://github.com/ins-uni-bonn/PQDT.

Table 4. Gabor noise parameters of ShapeNet-Deform

Param	Training	Evaluation
α	$\mathcal{U}(0.2,\ 0.6)$	0.4
K	$\{8, 9, \ldots, 24\}$	16
f	$\mathcal{U}(1.0,\ 3.0)$	2.0
σ	$\mathcal{U}(0.4,\ 0.6)$	0.5
ϕ	$\mathcal{U}(0, 2\pi)$	Seed$(0, 2\pi)$
$\mathbf{d}_k$	$\mathcal{U}$(unit sphere)	Seed(unit sphere)
$\mathbf{o}_k$	$\mathcal{U}(-5, 5)^3$	Seed$(-5, 5)^3$

Table 5. Parameters of ShapeNet-Occ under 3 difficulty levels

Param	Simple	Moderate	Hard
β	1.1	1.2	1.3
σ_{noise}	0.003	0.004	0.005
n_{occ}	10	20	30

ShapeNetCar-Occ. To simulate realistic real-world occlusions around vehicles, we construct the ShapeNetCar-Occ dataset by placing random occluding objects (e.g., traffic signs, pedestrians) around each ShapeNet car category and rendering partial observations via raycasting. Given a car mesh $\mathcal{M}$ and an occluder mesh $\mathcal{O}$, we first sample a virtual LiDAR viewpoint. For each model instance, we compute the ray from the LiDAR origin toward the car center, determine the first intersection with $\mathcal{M}$, and place the occluder near this intersection point. The occluder is randomly rotated around the vertical axis and slightly shifted along the viewing direction to ensure it lies between the sensor and the car while remaining grounded. A second raycasting pass uniformly samples n_{rays} viewing directions on the sphere (Fibonacci lattice). Only rays whose directions face the car are used. Intersections with the combined scene $\mathcal{S} = \mathcal{M} \cup \mathcal{O}$ produce the observed point set, while a separate raycasting pass on the clean mesh $\mathcal{M}$ identifies which points originate from occluders. Samples with insufficient occlusion points are resampled. Points outside a rescaled car bounding box are removed, and the final point cloud is subsampled to N points using FPS and perturbed with Gaussian noise.

Since online raycasting–based data generation is computationally expensive, we generate the ShapeNetCar-Occ dataset offline using the three difficulty levels defined in Tab. 5.

Here, β denotes the bounding-box rescaling factor: larger values relax spatial filtering and allow more distant points to be preserved. The Gaussian noise level σ_{noise} controls the magnitude of sensor-like perturbations. The threshold n_{occ} specifies the minimum number of occlusion points required for a valid sample, thereby regulating occlusion

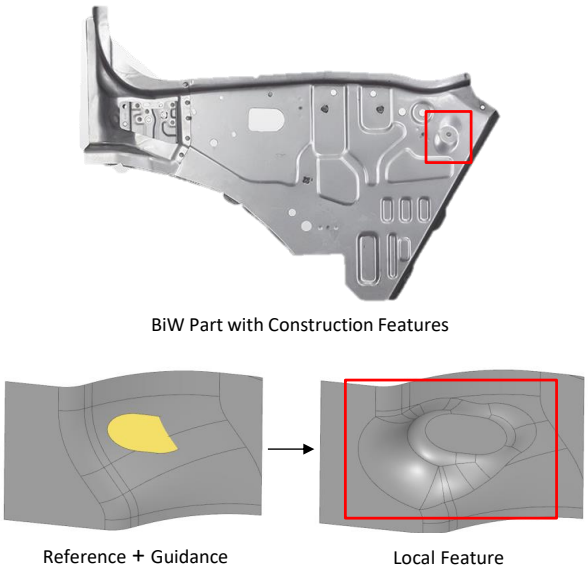

Figure 9. Illustration of PFS usage in an industrial application. The BiW part image (top) is sourced from [13].

severity. For each difficulty level, 32 partial point clouds are generated per model from random viewpoints. These settings jointly produce progressively more challenging scenarios and enable controlled evaluation of robustness under partial observations. The dataset is available via https://github.com/ins-uni-bonn/PQDT.

PFS. We use the patchified freeform surface (PFS) data from Body-in-White (BiW) components in the automotive industry as an additional target domain for our restoration backbone. As illustrated in Fig. 9, the dataset is constructed by taking the reference geometry and the guidance geometry (bottom left) as inputs, while the geometry containing local construction features (e.g., embossments for reinforcement, mounting, welding, etc.) serves as the ground truth. The guidance geometry is manually annotated as a simplified cue indicating where local features should be generated. Using CAD software, we capture part snapshots both with and without these local features. The dataset consists of 800 training samples and 80 evaluation samples.

Compared with the previously used datasets, this task exhibits different characteristics: the completion requirement is smaller, while deformation plays a more significant role. Consistency and fine-detail reconstruction become key challenges for this dataset. In practical applications, e.g., reverse engineering, one might start with an incomplete 3D scan and use the guidance geometry to refine the coarse surface. Similarly, this setup enables adding, editing, or inpainting local features on an early-stage CAD model through sketch-based guidance.

Due to the confidential nature of the data and its inclusion of sensitive product-related information, this dataset cannot be made publicly available.

Table 6. Complexity Analysis

Method	Params	FLOPs	$CD_{\ell 2}$-55	$CD_{\ell 2}$-34
FoldingNet [65]	2.3M	27.6G	3.12	3.62
PCN [69]	5.0M	15.3G	2.66	3.85
GRNet [61]	73.2M	40.4G	1.97	2.99
PoinTr [67]	33.0M	8.9G	1.09	2.05
SeedFormer [75]	3.2M	10.3G	0.92	1.34
AdaPoinTr [68]	32.5M	18.1G	0.81	1.23
ProxyFormer [28]	12.2M	9.9G	0.93	1.42
AnchorFormer [4]	30.5M	8.1G	0.76	1.19
PQDT	64.2M	60.3G	**0.68**	**0.99**

D. Complexity Analysis

Benefiting from the dual-stage transformer architecture, our model naturally carries a larger parameter count and higher theoretical computation cost (FLOPs) than most existing approaches. However, this increased capacity directly translates into significantly stronger geometric reasoning and superior completion quality. As shown in Tab. 6, PQDT achieves the best reconstruction accuracy on ShapeNet-55 by a clear margin, outperforming all prior baselines across both benchmarks and our newly introduced datasets. Despite its enhanced capability, the model size and computational cost remain well within a practical and efficient range, offering an excellent balance between complexity and the substantial performance gains delivered.

E. Additional Ablation Study

To evaluate the effectiveness of our model design, we conduct a comprehensive ablation study, as summarized in Tab. 7, focusing on the submodules introduced in Sec. 3. We use a vanilla Transformer for point clouds as the baseline (Model A) and analyze performance differences across three key aspects: query generation, seed prediction, and positional encoding. In each experiment, we modify only the component of interest while keeping all other settings consistent with the best-performing configuration.

Query Generation. Starting from the single-stage variant (Model B), we first incorporate the pseudo-query mechanism (Model C) to examine the impact of introducing the second stage in the transformer. Importantly, this transition does not involve adding extra transformer layers; instead, it restructures the query formulation within the existing architecture. As shown in Tab. 7, the parameter count increases only marginally from 55.0M to 57.8M, indicating that the performance gains are not simply due to increased model capacity. Despite this minimal overhead, the addition of pseudo-queries leads to consistent improvements in both CD and F1-Score, clearly demonstrating the effectiveness of our dual-stage design. This suggests that the performance boost

Table 7. Ablation study on ShapeNetCar-Occ. We report the evaluation results and model sizes with different model designs including single query generation (Query), auxiliary pseudo-query generation (Pseudo-Query), Dynamic Query Selection (DQS), linear seed projection (Linear Proj.), query-based decoder with partial points and spherical points as initialization (P/S-Q Dec.), coordinates encoding (CE), kNN feature grouping (kNN), and geometric-embedding (GE).

Category	Model	Setups				$CD_{\ell 2}$ (↓)	F1 (↑)
Baseline	A	Query + Linear Proj. + SA				0.857	0.243
		Query	Pseudo-Query	DQS		$CD_{\ell 2}$ (↓)	F1 (↑)
Query Generation	B	✓				0.838	0.256
	C	✓	✓			0.827	0.260
	PQDT	✓	✓	✓		0.818	0.261
		Linear Proj.	P-Q Dec.	S-Q Dec.		$CD_{\ell 2}$ (↓)	F1 (↑)
Seed Prediction	D	✓				0.851	0.235
	E		✓			0.841	0.260
	PQDT			✓		0.818	0.261
		CE	kNN	GE		$CD_{\ell 2}$ (↓)	F1 (↑)
Positional Encoding	F	✓				0.850	0.248
	G	✓	✓			0.865	0.251
	PQDT	✓		✓		0.818	0.261

Model	A	B	C	D	E	F	G	PQDT
Params	44.9M	55.0M	57.8M	58.6M	64.2M	63.1M	63.1M	64.2M

stems from better feature refinement and query representation rather than scaling the model size. Further introducing the dynamic query selection (DQS) module (PQDT) yields additional gains, achieving a CD of 0.82 and an F1-Score of 0.261.

Seed Prediction. We compare the commonly used linear projection from global features (Model D) with our query-based DETR-style decoding scheme, which achieves a significant improvement, particularly in F1-Score. Using partial input initialization (Model E) performs slightly worse than the spherical initialization adopted in our final model, suggesting that the latter provides a more stable prior for query generation.

Positional Encoding. Model F encodes spatial information using only the point coordinates, while Model G incorporates local context through kNN-based neighborhood aggregation. The latter slightly decreases CD performance (by 0.02), likely due to overfitting. In contrast, our geometric-embedding achieves the best overall results, highlighting the advantage of our geometry-aware positional design.

We also investigate the component level ablation study for the DQS and query-based seed generation on PFS in Tabs. 8 and 9.

Gumbel Noise Perturbation. We vary the scaling factor of the Gumbel noise added to the query scores in the DQS module to study its effect on performance. A larger noise scale introduces more randomness in query selection, which can strengthen the regularization effect; however, if the noise becomes too large, performance degrades. Conversely, using a very small scale (i.e., effectively no noise) also harms performance. In our setup, we use a noise scale of 1.0.

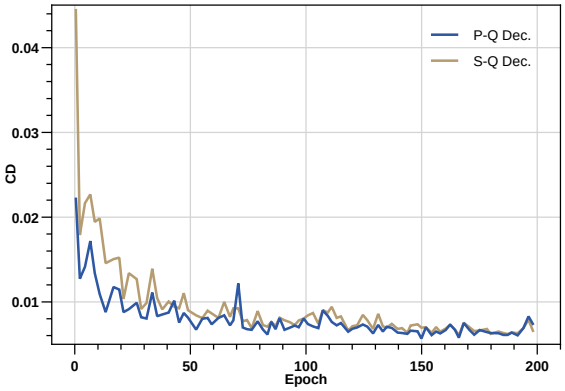

Figure 10. Training loss under different initializations of query-based seed generation on PFS.

Table 8. Ablation study on Gumbel-Top-k of dynamic query selection module on ShapeNetCar-Occ.

Method	$CD_{\ell 2}$ (↓)	F1 (↑)
w/o noise	0.833	0.253
noise scale = 0.5	0.825	0.256
noise scale = 1.0	0.818	0.261
noise scale = 2.0	0.834	0.247

Initialization of Seed Generation. Beyond the comparison with Model E in Tab. 7 on ShapeNetCar-Occ, we also investigate how different initializations of the query-based decoder influence seed generation in the first stage of the transformer. We evaluate partial-initialized and spherical-initialized queries (P/S-Q Dec.) on the PFS dataset. Because the input and ground-truth shapes in PFS share a substantial amount of common geometry, using the partial input as queries should theoretically yield better performance. As shown in Tab. 9, both variants achieve similar CD scores, but P-Q Dec. attains a higher F-Score, and Fig. 10 further shows that it converges faster.

In all experiments across the datasets mentioned in the paper, we adopt spherical initialization to demonstrate the generality of our model. However, it is also possible to tune the query initialization, such as using partial input, to better suit specific tasks or datasets.

F. Additional Evaluation Results

In Tab. 10, we report results for the five most frequent categories (Table, Chair, Airplane, Car, and Sofa) and the five least frequent categories (Birdhouse, Bag, Remote, Keyboard, and Rocket), along with the average $CD_{\ell 2}$ and F1-Score. These metrics are compared against the baselines

Table 9. Ablation study on query-based seed generation with partial points and spherical points as initialization (P/S-Q Dec.) on PFS.

Method	$CD_{\ell 2}$ (↓)	F1 (↑)	F0.5 (↑)
P-Q Dec.	0.17	0.847	0.327
S-Q Dec.	0.16	0.834	0.311

under three levels of incompleteness. In Tab. 11, we further provide categorical results for all 55 categories on ShapeNet-55 and ShapeNet-Deform. In Tab. 12, we present results on the 22 unseen categories from ShapeNet-34. Finally, in Tab. 13, we report detailed comparisons with baselines across the three difficulty levels of ShapeNetCar-Occ.

G. Qualitative Samples

We provide additional qualitative results for PQDT on ShapeNet-Deform in Fig. 11, ShapeNetCar-Occ in Fig. 12, and PFS in Fig. 13.

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 5, 2

[2] Matthew Berger, Andrea Tagliasacchi, Lee Seversky, Pierre Alliez, Joshua Levine, Andrei Sharf, and Cláudio Silva. State of the art in surface reconstruction from point clouds. *Eurographics 2014-State of the Art Reports*, 2014. 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. 3, 2

[4] Zhikai Chen, Fuchen Long, Zhaofan Qiu, Ting Yao, Wengang Zhou, Jiebo Luo, and Tao Mei. Anchorformer: Point cloud completion from discriminative nodes. In *CVPR*, pages 13581–13590, 2023. 2, 4, 6, 7, 3, 9

[5] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *CVPR*, pages 5868–5877, 2017. 2

[6] Dasith de Silva Edirimuni, Xuequan Lu, Zhiwen Shao, Gang Li, Antonio Robles-Kelly, and Ying He. Iterativepfn: True iterative point cloud filtering. In *CVPR*, pages 13530–13539, 2023. 2

[7] Yi Du, Zhipeng Zhao, Shaoshu Su, Sharath Golluri, Haoze Zheng, Runmao Yao, and Chen Wang. Superpc: a single diffusion model for point cloud completion, upsampling, denoising, and colorization. In *CVPR*, pages 16953–16964, 2025. 2, 3, 7

[8] Fan Duan, Jiahao Yu, and Li Chen. T-corresnet: template guided 3d point cloud completion with correspondence pooling query generation strategy. In *ECCV*, pages 90–106, 2024. 2, 6, 7

[9] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, pages 605–613, 2017. 5, 2

[10] Shachar Fleishman, Daniel Cohen-Or, and Cláudio T Silva. Robust moving least-squares fitting with sharp features. *ACM TOG*, 24(3):544–552, 2005. 2

[11] Bingchen Gong, Yinyu Nie, Yiqun Lin, Xiaoguang Han, and Yizhou Yu. Me-pcn: Point completion conditioned on mask emptiness. In *ICCV*, pages 12488–12497, 2021. 2

[12] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *ECCV*, pages 230–246, 2018. 3

[13] SKH Group. Body in white (biw). https://www.skhgroup.co.in/our-products/body-in-white-biw. Accessed: Nov. 2025. 4

[14] Xiaoguang Han, Zhen Li, Haibin Huang, Evangelos Kalogerakis, and Yizhou Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *ICCV*, pages 85–93, 2017. 2

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, pages 6840–6851, 2020. 3

[16] Tao Hu, Zhizhong Han, Abhinav Shrivastava, and Matthias Zwicker. Render4completion: Synthesizing multi-view depth maps for 3d shape completion. In *ICCVW*, 2019. 2

[17] Wei Hu, Zeqing Fu, and Zongming Guo. Local frequency interpretation and non-local self-similarity on graph for point cloud inpainting. *IEEE TIP*, 28(8):4087–4100, 2019. 2

[18] Wei Hu, Xiang Gao, Gene Cheung, and Zongming Guo. Feature graph learning for 3d point cloud denoising. *IEEE TSP*, 68:2841–2856, 2020. 2

[19] Tianxin Huang, Hao Zou, Jinhao Cui, Xuemeng Yang, Mengmeng Wang, Xiangrui Zhao, Jiangning Zhang, Yi Yuan, Yifan Xu, and Yong Liu. Rfnet: Recurrent forward network for dense point cloud completion. In *ICCV*, pages 12508–12517, 2021. 2

[20] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *CVPR*, pages 7662–7670, 2020. 1, 2

[21] Yoni Kasten, Ohad Rahamim, and Gal Chechik. Point cloud completion with pretrained text-to-image diffusion models. In *NeurIPS*, pages 12171–12191, 2023. 2

[22] Vladimir G Kim, Wilmot Li, Niloy J Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas Funkhouser. Learning part-based templates from large collections of 3d shapes. *ACM TOG*, 32(4):1–12, 2013. 2

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[24] Lingdong Kong, Youquan Liu, Xin Li, Runnan Chen, Wenwei Zhang, Jiawei Ren, Liang Pan, Kai Chen, and Ziwei Liu. Robo3d: Towards robust and reliable 3d perception against corruptions. In *ICCV*, pages 19994–20006, 2023. 2

[25] Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *ICML*, pages 3499–3508, 2019. 5

[26] Ares Lagae, Sylvain Lefebvre, George Drettakis, and Philip Dutré. Procedural noise using sparse gabor convolution. *ACM SIGGRAPH*, 28(3):54–64, 2009. 6, 3

Table 10. Results of our method and state-of-the-art methods on ShapeNet-55. We report the detailed results for each method on 10 categories and the overall results on 55 categories for three difficulty degrees. We use CD-S, CD-M and CD-H to represent the CD results under the Simple, Moderate and Hard settings. We also provide results under the F-Score@1% metric.

Method	Tab.	Cha.	Pla.	Car	Sof.	Bir.	Bag	Rem.	Key.	Roc.	CD-S	CD-M	CD-H	$CD_{\ell 2}$ (↓)	F1 (↑)
FoldingNet [65]	2.53	2.81	1.43	1.98	2.48	4.71	2.79	1.44	1.24	1.48	2.67	2.66	4.05	3.12	0.082
PCN [69]	2.13	2.29	1.02	1.85	2.06	4.50	2.86	1.32	0.89	1.32	1.94	1.96	4.08	2.66	0.133
GRNet [61]	1.63	1.88	1.02	1.62	1.72	2.97	2.06	1.09	0.89	1.03	1.35	1.71	2.85	1.97	0.238
PoinTr [67]	0.81	0.95	0.44	0.91	0.79	1.86	0.93	0.53	0.38	0.57	0.58	0.88	1.79	1.09	0.464
SnowflakeNet [60]	0.98	1.12	0.54	0.98	1.02	1.93	1.08	0.57	0.48	0.61	0.70	1.06	1.96	1.24	0.398
SeedFormer [75]	0.72	0.81	0.40	0.89	0.71	1.51	0.79	0.46	0.36	0.50	0.50	0.77	1.49	0.92	0.472
AdaPoinTr [68]	0.62	0.69	0.33	0.81	0.63	1.33	0.68	0.38	0.33	**0.34**	0.49	0.69	1.24	0.81	0.503
ProxyFormer [28]	0.70	0.83	0.34	0.78	0.69	1.57	0.79	0.36	0.34	0.46	0.49	0.75	1.55	0.93	0.483
T-CorresNet [8]	0.60	0.68	**0.32**	-	-	1.17	**0.60**	-	0.27	-	0.50	0.68	1.23	0.80	0.485
AnchorFormer [4]	0.58	0.67	0.33	0.69	0.58	1.35	0.64	0.36	0.27	0.42	0.41	0.61	1.26	0.76	0.558
PQDT	**0.52**	**0.60**	**0.32**	**0.65**	**0.51**	**1.13**	**0.60**	**0.31**	**0.24**	0.41	**0.34**	**0.55**	**1.13**	**0.68**	**0.570**

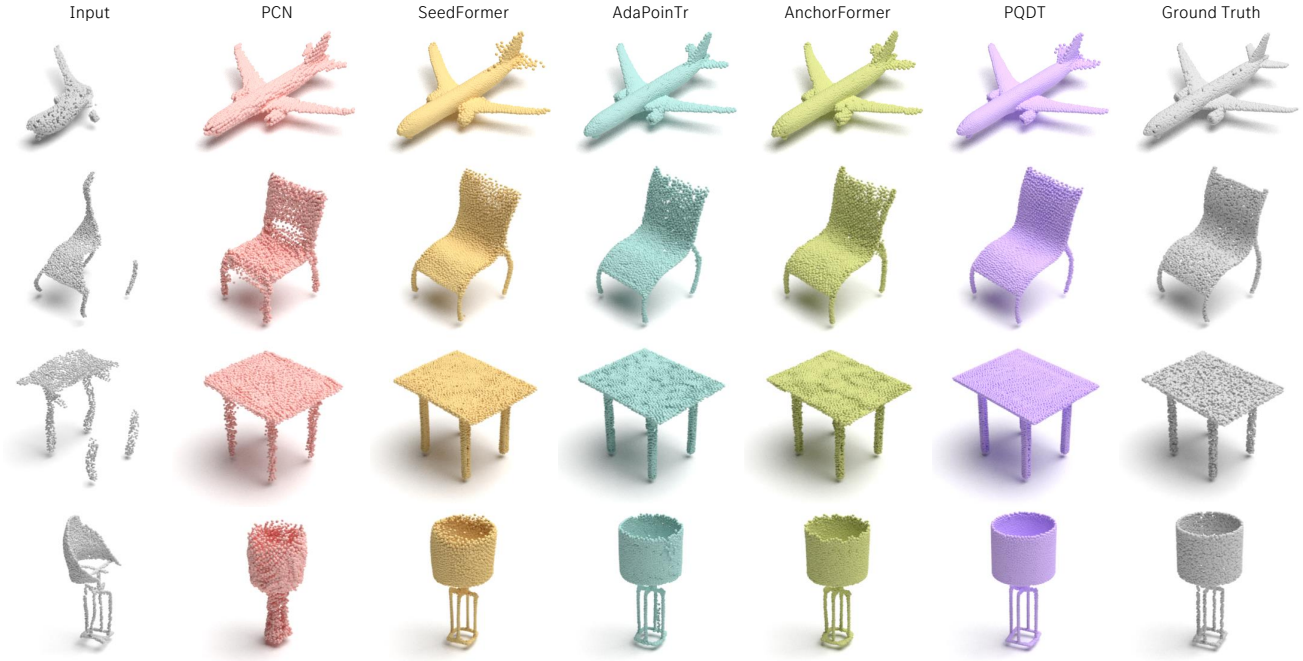

Figure 11. Qualitative evaluation on ShapeNet-Deform

[27] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1988. 6

[28] Shanshan Li, Pan Gao, Xiaoyang Tan, and Mingqiang Wei. Proxyformer: Proxy alignment assisted point cloud completion with missing part sensitive transformer. In *CVPR*, pages 9466–9475, 2023. 2, 4, 6, 7

[29] Yangyan Li, Angela Dai, Leonidas Guibas, and Matthias Nießner. Database-assisted object retrieval for real-time 3d reconstruction. In *CGF*, pages 435–446, 2015. 2

[30] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 1

[31] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, pages 2837–2845, 2021. 3

[32] Aihua Mao, Zihui Du, Yu-Hui Wen, Jun Xuan, and Yong-Jin Liu. Pd-flow: A point cloud denoising framework with normalizing flows. In *ECCV*, pages 398–415, 2022. 2

[33] Aihua Mao, Biao Yan, Zijing Ma, and Ying He. Denoising point clouds in latent space via graph convolution and invertible neural network. In *CVPR*, pages 5768–5777, 2024. 2

[34] Niloy Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. Symmetry in 3d geometry: Extraction and applications. *CGF*, 32(6), 2013. 2

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 1

[36] Mark Pauly, Niloy J Mitra, Joachim Giesen, Markus H Gross,

Table 11. Categorical results of our method on ShapeNet-55 and ShapeNet-Deform. We report the detailed results under $CD_{\ell 2}$ and F-Score@1% metric for Simple, Moderate and Hard settings.

Category	**ShapeNet-55**						**ShapeNet-Deform**					
	Simple		Moderate		Hard		Simple		Moderate		Hard	
	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)
airplane	0.18	0.765	0.26	0.743	0.51	0.627	0.36	0.519	0.45	0.498	0.70	0.433
trash bin	0.49	0.472	0.77	0.461	1.48	0.413	0.83	0.159	1.09	0.151	2.13	0.126
bag	0.32	0.590	0.52	0.576	0.95	0.492	0.63	0.270	0.82	0.252	1.30	0.210
basket	0.46	0.503	0.56	0.491	1.10	0.434	0.76	0.193	0.87	0.180	1.68	0.146
bathtub	0.37	0.549	0.58	0.533	1.07	0.467	0.69	0.237	0.89	0.219	1.48	0.177
bed	0.42	0.544	0.65	0.532	1.35	0.466	0.78	0.226	1.01	0.207	1.79	0.174
bench	0.22	0.651	0.32	0.653	0.65	0.587	0.42	0.399	0.50	0.372	0.86	0.324
birdhouse	0.55	0.517	0.94	0.500	1.90	0.433	0.98	0.177	1.37	0.164	2.60	0.141
bookshelf	0.40	0.519	0.62	0.509	1.22	0.447	0.71	0.222	0.92	0.210	1.59	0.181
bottle	0.22	0.650	0.47	0.615	0.98	0.507	0.47	0.365	0.75	0.331	1.41	0.246
bowl	0.39	0.502	0.46	0.493	0.85	0.433	0.63	0.195	0.73	0.179	1.18	0.151
bus	0.28	0.588	0.40	0.584	0.58	0.533	0.54	0.321	0.64	0.299	0.85	0.245
cabinet	0.38	0.499	0.47	0.495	0.80	0.454	0.64	0.209	0.73	0.199	1.05	0.175
camera	0.56	0.523	1.26	0.504	2.63	0.430	1.09	0.171	1.76	0.164	3.39	0.140
can	0.38	0.515	0.69	0.501	1.48	0.439	0.65	0.248	1.00	0.235	2.30	0.195
cap	0.33	0.572	0.32	0.554	0.67	0.460	0.47	0.281	0.59	0.252	1.00	0.210
car	0.42	0.489	0.62	0.475	0.93	0.424	0.89	0.142	1.01	0.137	1.28	0.123
cellphone	0.21	0.626	0.26	0.620	0.37	0.556	0.36	0.431	0.41	0.416	0.56	0.339
chair	0.28	0.610	0.47	0.591	1.06	0.492	0.57	0.291	0.78	0.263	1.41	0.226
clock	0.37	0.559	0.56	0.547	1.05	0.484	0.64	0.280	0.79	0.267	1.34	0.227
keyboard	0.18	0.650	0.24	0.657	0.31	0.596	0.36	0.445	0.42	0.408	0.60	0.374
dishwasher	0.38	0.503	0.47	0.500	1.03	0.450	0.63	0.223	0.75	0.205	1.39	0.180
display	0.27	0.583	0.43	0.579	0.84	0.518	0.51	0.317	0.65	0.295	1.11	0.260
earphone	0.45	0.600	0.77	0.560	2.00	0.445	0.95	0.225	1.17	0.216	3.06	0.186
faucet	0.38	0.734	0.82	0.658	1.94	0.483	0.71	0.391	1.14	0.353	2.52	0.276
file cabinet	0.42	0.502	0.54	0.498	1.14	0.448	0.71	0.209	0.86	0.201	1.54	0.176
guitar	0.09	0.929	0.14	0.880	0.25	0.794	0.19	0.793	0.25	0.775	0.37	0.715
helmet	0.59	0.501	1.17	0.480	2.92	0.410	1.13	0.141	1.70	0.128	3.38	0.107
jar	0.44	0.537	0.79	0.512	1.79	0.433	0.81	0.211	1.20	0.193	2.51	0.156
knife	0.10	0.926	0.21	0.832	0.36	0.700	0.24	0.766	0.33	0.724	0.51	0.644
lamp	0.31	0.767	0.81	0.700	2.06	0.554	0.64	0.443	1.17	0.406	2.88	0.339
laptop	0.21	0.601	0.24	0.599	0.39	0.534	0.37	0.371	0.41	0.349	0.57	0.304
loudspeaker	0.46	0.519	0.74	0.509	1.42	0.451	0.82	0.217	1.09	0.201	1.82	0.171
mailbox	0.19	0.758	0.47	0.703	1.72	0.548	0.45	0.463	0.82	0.404	2.28	0.334
microphone	0.39	0.790	0.93	0.707	2.32	0.546	0.65	0.482	1.24	0.423	3.58	0.331
microwaves	0.42	0.503	0.54	0.500	1.21	0.455	0.71	0.208	0.90	0.183	1.54	0.165
motorbike	0.45	0.535	0.72	0.513	1.16	0.418	0.97	0.178	1.14	0.176	1.46	0.158
mug	0.54	0.473	0.74	0.462	1.53	0.413	0.85	0.165	1.08	0.155	2.08	0.127
piano	0.44	0.543	0.59	0.540	1.19	0.484	0.73	0.249	0.91	0.230	1.45	0.190
pillow	0.35	0.573	0.50	0.543	0.97	0.448	0.72	0.206	0.87	0.183	1.38	0.143
pistol	0.26	0.701	0.42	0.657	0.70	0.528	0.53	0.394	0.66	0.380	0.93	0.339
flowerpot	0.59	0.522	0.91	0.501	1.74	0.427	1.06	0.179	1.33	0.167	2.28	0.141
printer	0.42	0.533	0.75	0.522	1.61	0.455	0.84	0.200	1.19	0.187	2.21	0.161
remote	0.19	0.666	0.31	0.651	0.44	0.564	0.39	0.452	0.50	0.437	0.69	0.336
rifle	0.19	0.855	0.31	0.787	0.54	0.669	0.37	0.639	0.51	0.610	0.75	0.548
rocket	0.14	0.862	0.37	0.789	0.72	0.687	0.35	0.611	0.61	0.556	1.00	0.474
skateboard	0.16	0.737	0.28	0.714	0.46	0.625	0.33	0.504	0.41	0.479	0.56	0.409
sofa	0.34	0.544	0.45	0.539	0.75	0.489	0.66	0.236	0.74	0.218	1.09	0.182
stove	0.41	0.544	0.63	0.531	1.18	0.465	0.73	0.234	0.97	0.217	1.55	0.191
table	0.29	0.604	0.41	0.601	0.86	0.542	0.49	0.356	0.61	0.336	1.11	0.286
telephone	0.21	0.625	0.26	0.623	0.39	0.562	0.36	0.436	0.42	0.425	0.57	0.352
tower	0.31	0.663	0.58	0.617	1.22	0.491	0.62	0.364	0.92	0.331	1.87	0.258
train	0.31	0.617	0.46	0.607	0.76	0.533	0.61	0.317	0.76	0.303	1.10	0.253
watercraft	0.24	0.717	0.42	0.678	0.72	0.566	0.55	0.376	0.72	0.345	1.08	0.283
washer	0.44	0.494	0.62	0.487	1.70	0.440	0.79	0.197	1.00	0.179	1.88	0.158
mean	0.34	0.608	0.55	0.586	1.13	0.505	0.63	0.319	0.85	0.298	1.54	0.253

Figure 12. Qualitative evaluation on ShapeNetCar-Occ. Red boxes indicate occluding objects.

Table 12. Categorical results of our method on 22 unseen objects of ShapeNet-34. We report the detailed results under $CD_{\ell 2}$ and F-Score@1% metric for Simple, Moderate and Hard settings.

Category	ShapeNet-34					
	Simple		Moderate		Hard	
	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)	$CD_{\ell 2}$ (↓)	F1 (↑)
bag	0.37	0.579	0.61	0.568	1.12	0.485
basket	0.41	0.516	0.60	0.505	1.50	0.438
birdhouse	0.54	0.545	0.91	0.527	1.90	0.452
bowl	0.42	0.499	0.53	0.488	1.11	0.426
camera	0.62	0.528	1.30	0.507	2.98	0.432
can	0.38	0.508	0.61	0.496	1.23	0.436
cap	0.33	0.569	0.94	0.532	3.66	0.426
keyboard	0.22	0.631	0.27	0.634	0.38	0.576
dishwasher	0.42	0.495	0.55	0.491	1.13	0.443
earphone	0.56	0.604	1.48	0.555	4.83	0.437
helmet	0.69	0.494	1.76	0.471	4.21	0.402
mailbox	0.32	0.721	0.67	0.670	1.64	0.531
microphone	0.53	0.803	1.16	0.723	3.04	0.551
microwaves	0.44	0.495	0.58	0.492	1.23	0.447
pillow	0.32	0.581	0.49	0.552	1.08	0.455
printer	0.47	0.522	0.83	0.512	1.91	0.449
remote	0.19	0.670	0.30	0.660	0.42	0.583
rocket	0.20	0.854	0.41	0.779	0.83	0.652
skateboard	0.21	0.724	0.49	0.683	0.84	0.535
tower	0.34	0.655	0.66	0.612	1.47	0.488
washer	0.43	0.497	0.61	0.491	1.54	0.443
mean	0.40	0.595	0.75	0.569	1.81	0.480

Table 13. Results of our method and state-of-the-art methods on ShapeNetCar-Occ. We report the detailed results for each method on Car category for three difficulty degrees. We use CD-S, CD-M and CD-H to represent the CD results under the Simple, Moderate and Hard settings. We also provide results under the F-Score@1% metric.

Method	CD-S	CD-M	CD-H	$CD_{\ell 2}$ (↓)	F1 (↑)
PCN [69]	1.41	1.41	1.43	1.42	0.125
PoinTr [67]	1.06	1.07	1.07	1.07	0.191
SnowflakeNet [60]	0.93	0.94	0.96	0.94	0.22
SeedFormer [75]	0.95	0.97	0.99	0.97	0.231
AdaPoinTr [68]	0.84	0.84	0.86	0.85	**0.251**
AnchorFormer [4]	0.92	0.93	0.95	0.93	0.225
PQDT	**0.81**	**0.82**	**0.82**	**0.82**	**0.261**

and Leonidas J Guibas. Example-based 3d scan completion. In *SGP*, page 32, 2005. 2

[37] Francesca Pistilli, Giulia Fracastoro, Diego Valsesia, and Enrico Magli. Learning graph-convolutional representations for point cloud denoising. In *ECCV*, pages 103–118, 2020. 2

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 1, 2

[39] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 1, 2, 4

[40] Zheng Qin, Hao Yu, Changjian Wang, Yulan Guo, Yuxing Peng, Slobodan Ilic, Dewen Hu, and Kai Xu. Geotransformer: Fast and robust point cloud registration with geometric transformer. *IEEE TPAMI*, 45(8):9806–9821, 2023. 4, 1

[41] Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J Mitra, and Maks Ovsjanikov. Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *CGF*, pages 185–203, 2020. 2

[42] Kripasindhu Sarkar, Kiran Varanasi, and Didier Stricker.

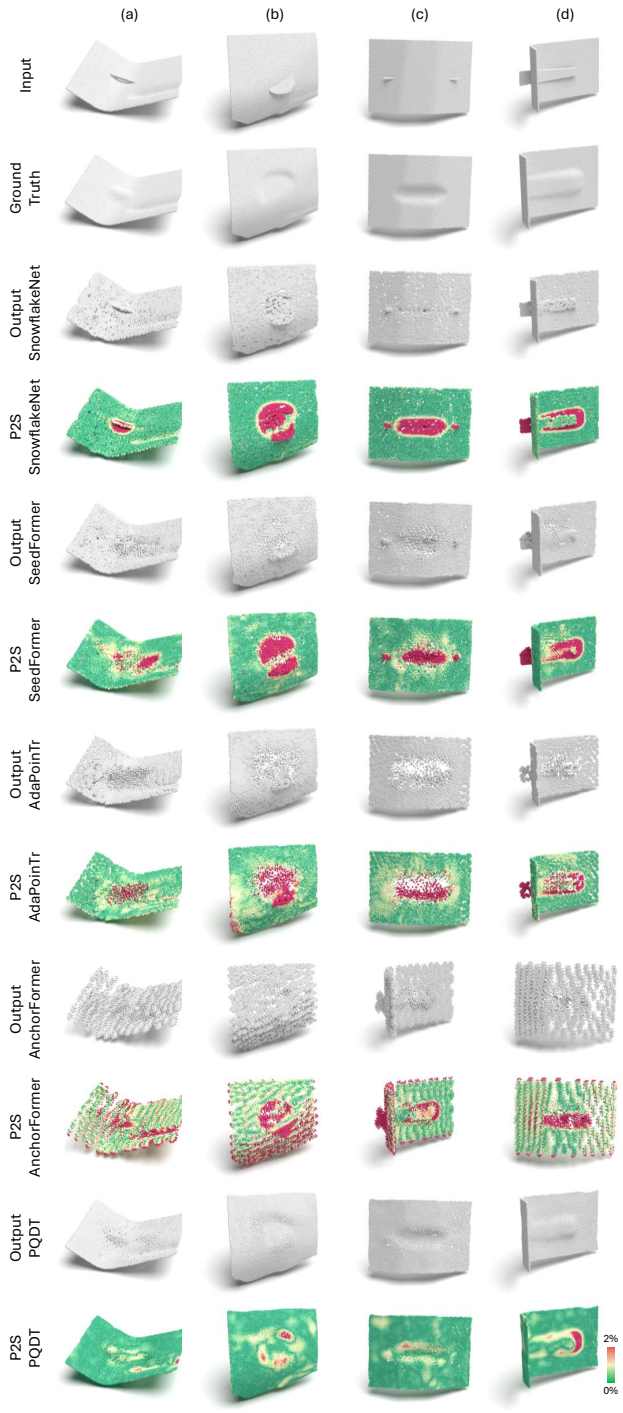

Figure 13. Qualitative evaluation on PFS. Colors indicate the point-to-surface (P2S) distance between the generated point clouds and the ground-truth mesh, with the maximum value (red) clamped to 2% of the radius of the object's bounding sphere. Examples (a)–(d) show BiW patches containing local construction features such as reinforcement or mounting regions, comparing our method against baseline approaches.

Learning quadrangulated patches for 3d shape parameterization and completion. In *3DV*, pages 383–392, 2017. 2

[43] David Stutz and Andreas Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *CVPR*, pages 1955–1964, 2018. 2

[44] Minhyuk Sung, Vladimir G Kim, Roland Angst, and Leonidas Guibas. Data-driven structural priors for shape completion. *ACM TOG*, 34(6):1–11, 2015. 2

[45] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *CVPR*, pages 3405–3414, 2019. 6, 2

[46] Lyne P. Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *CVPR*, pages 383–392, 2019. 2

[47] Mikaela Angelina Uy, Jingwei Huang, Minhyuk Sung, Tolga Birdal, and Leonidas Guibas. Deformation-aware 3d model embedding and retrieval. In *ECCV*, pages 397–413, 2020. 3

[48] Mikaela Angelina Uy, Vladimir G Kim, Minhyuk Sung, Noam Aigerman, Siddhartha Chaudhuri, and Leonidas J Guibas. Joint learning of 3d shape retrieval and deformation. In *CVPR*, pages 11713–11722, 2021. 3

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, page 6000–6010, 2017. 1

[50] Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 3dn: 3d deformation network. In *CVPR*, pages 1038–1046, 2019. 3

[51] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *CVPR*, pages 790–799, 2020. 2

[52] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 38(5): 1–12, 2019. 4, 1

[53] Guangshun Wei, Yuan Feng, Long Ma, Chen Wang, Yuanfeng Zhou, and Changjian Li. Pcdreamer: Point cloud completion through multi-view diffusion priors. In *CVPR*, pages 27243–27253, 2025. 5, 7

[54] Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In *CVPR*, pages 1939–1948, 2020. 1, 2

[55] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992. 2

[56] Lintai Wu, Qijian Zhang, Junhui Hou, and Yong Xu. Leveraging single-view images for unsupervised 3d point cloud completion. *IEEE TMM*, 27:940–953, 2023. 2

[57] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *NeurIPS*, 2022. 1

[58] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *CVPR*, 2024. 1

[59] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 5

[60] 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 *ICCV*, pages 5499–5509, 2021. 1, 2, 5, 6, 7, 9

[61] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *ECCV*, pages 365–381, 2020. 2, 6, 4, 7

[62] Sang Michael Xie and Stefano Ermon. Reparameterizable subset sampling via continuous relaxations. In *IJCAI*, page 3919–3925, 2019. 5

[63] Hang Xu, Chen Long, Wenxiao Zhang, Yuan Liu, Zhen Cao, Zhen Dong, and Bisheng Yang. Explicitly guided information interaction network for cross-modal point cloud completion. In *ECCV*, pages 414–432, 2024. 2

[64] Bo Yang, Hongkai Wen, Sen Wang, Ronald Clark, Andrew Markham, and Niki Trigoni. 3d object reconstruction from a single depth view with adversarial learning. In *ICCVW*, pages 679–688, 2017. 2

[65] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*, pages 206–215, 2018. 5, 6, 4, 7

[66] Hao Yu, Zheng Qin, Ji Hou, Mahdi Saleh, Dongsheng Li, Benjamin Busam, and Slobodan Ilic. Rotation-invariant transformer for point cloud matching. In *CVPR*, pages 5384–5393, 2023. 4, 5

[67] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *CVPR*, pages 12498–12507, 2021. 2, 3, 4, 5, 6, 7, 9

[68] Xumin Yu, Yongming Rao, Ziyi Wang, Jiwen Lu, and Jie Zhou. Adapointr: Diverse point cloud completion with adaptive geometry-aware transformers. *IEEE TPAMI*, 45(12): 14114–14130, 2023. 2, 3, 5, 6, 7, 4, 9

[69] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *3DV*, pages 728–737, 2018. 1, 2, 5, 6, 7, 4, 9

[70] Feng Zhang, Chao Zhang, Huamin Yang, and Lin Zhao. Point cloud denoising with principal component analysis and a novel bilateral filter. *Traitement du signal*, 36:393–398, 2019. 2

[71] Xuancheng Zhang, Yutong Feng, Siqi Li, Changqing Zou, Hai Wan, Xibin Zhao, Yandong Guo, and Yue Gao. View-guided point cloud completion. In *CVPR*, pages 15890–15899, 2021. 2

[72] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, pages 16259–16268, 2021. 1, 2, 4

[73] Yinglong Zheng, Guiqing Li, Xuemiao Xu, Shihao Wu, and Yongwei Nie. Rolling normal filtering for point clouds. *CAGD*, 62(C):16–28, 2018. 2

[74] Feng Zhou, Qi Zhang, Ju Dai, Lei Li, Qing Fan, and Junliang Xing. Position-aware guided point cloud completion with clip model. In *AAAI*, pages 10734–10742, 2025. 2

[75] Haoran Zhou, Yun Cao, Wenqing Chu, Junwei Zhu, Tong Lu, Ying Tai, and Chengjie Wang. Seedformer: Patch seeds based point cloud completion with upsample transformer. In *ECCV*, pages 416–432, 2022. 2, 4, 5, 6, 7, 3, 9

[76] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 6

[77] Zhe Zhu, Honghua Chen, Xing He, Weiming Wang, Jing Qin, and Mingqiang Wei. Svdformer: Complementing point cloud via self-view augmentation and self-structure dual-generator. In *ICCV*, pages 14508–14518, 2023. 2