VISNET: PROJECTIVE GEOMETRY-VIEW FUSION FOR POINT CLOUD VISIBILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Per-point visibility prediction from raw point clouds requires reasoning about both 3D geometry and view-dependent occlusion. Existing learning-based approaches primarily infer visibility from geometric representations conditioned on viewing direction, without explicitly modeling occlusion interactions between points that become locally coupled after projection into the view plane. We introduce *projective geometry-view fusion*, which combines 3D geometric context with explicit occlusion reasoning over projected point neighborhoods. VISNET implements this idea by constructing a view-dependent graph over projected neighborhoods and reasoning about possible occlusion through local message passing. The resulting view-specific representations are then fused with 3D geometric features through cross-cue interactions and neighborhood-level information exchange on the same projected graph. VISNET achieves state-of-the-art performance on sparse, noisy, and out-of-distribution point clouds, improving accuracy by 1.4-3.4% and mIoU by 3.0-6.9% over prior methods across 1k-32k inputs.

1 INTRODUCTION

Point cloud visibility refers to determining the subset of points observable from a given viewpoint. It is a critical prerequisite for downstream tasks, including surface reconstruction, view-dependent rendering, and geometric processing. Unlike meshes or solid models, however, raw point clouds consist of discrete collections of points without explicit connectivity between them. This makes point cloud visibility more challenging, as a viewing ray can pass through gaps between sample points, even when the points collectively represent a continuous opaque surface.

Existing approaches address this ambiguity in different ways. A straightforward strategy is to first reconstruct a surface from the point cloud and then compute visibility based on the reconstructed geometry. However, surface reconstruction is itself an ill-posed problem Berger et al. (2017); Sulzer et al. (2024), and errors in the reconstructed surface can directly affect the resulting visibility estimates. It also introduces substantial computational overhead (see appendix Table 6 for a detailed breakdown of the runtime and visibility accuracy of this reconstruction-based pipeline). Geometric methods such as Hidden Point Removal (HPR) Katz et al. (2007) avoid explicit reconstruction, but their performance is sensitive to sampling density, non-uniformity, and complex geometry. More recently, learning-based approaches such as NeuVis Wang et al. (2025) predict visibility directly from raw points, offering greater robustness to these challenges. However, they primarily rely on 3D geometric representations conditioned on viewing direction, leaving projection-induced point interactions implicit. Figure 1 illustrates these challenges on an unoriented, non-uniform, out-of-distribution point cloud, where thin structures, stacked surfaces, and sharp transitions lead to frequent visibility errors.

The key difficulty is that the relational structure required for visibility computation is not necessarily captured by local neighborhoods that are encoded by standard 3D representations. Instead, occlusion is determined along the camera viewing direction rather than by Euclidean proximity: points lying on different parts of a shape can project to the same local region, where nearer samples may collectively indicate a foreground surface that may occlude farther samples. As a result, representations built primarily around local 3D neighborhoods may miss interactions that are most relevant for visibility. Projection provides a different, view-dependent neighborhood structure that brings these potentially occluding and occluded samples into direct interaction, exposing their relative depth and local spatial arrangement available for visibility reasoning.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

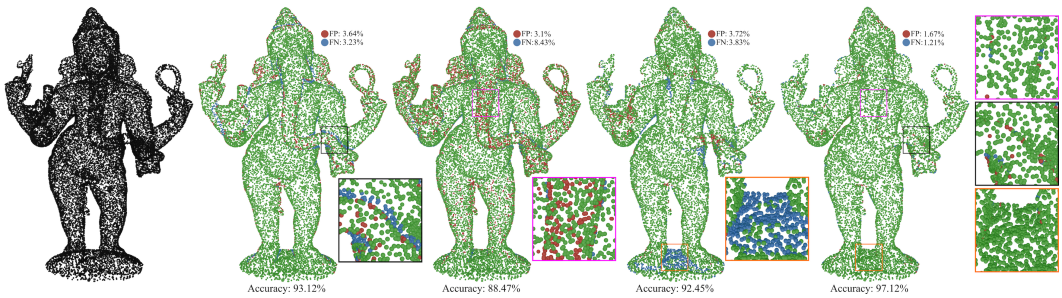

Figure 1: Comparison of point cloud visibility methods on an unoriented, non-uniform, out-of-distribution input (left). Results of HPR Katz et al. (2007), HPRO Katz & Tal (2025), Neural Visibility Wang et al. (2025), and our VisNet. Our approach demonstrates superior accuracy, with significantly lower false-negative (FN) and false-positive (FP) rates, with better handling of complex cases, including thin structures, stacked surfaces, and sharp edge transitions.

To address this representation mismatch, we introduce *projective geometry-view fusion*, which complements 3D geometric representations with a view-dependent relational representation induced by projection. We implement this idea in VISNET, where points are projected into the camera's view plane and the graph is constructed from neighborhoods in the resulting projected space. This produces a view-dependent graph that groups points with nearby projections, including points that may be distant in 3D but compete for visibility along similar viewing directions. Message passing over this graph learns view-specific representations from their projected positions, relative depth, and viewing cues. These representations are then fused with 3D geometric features through point-wise cross-cue interaction, followed by neighborhood-level information exchange over the same projected graph. In this way, the projection-induced neighborhoods support both occlusion reasoning and the propagation of fused geometry-view information.

We validate VISNET on ShapeNet Chang et al. (2015) across varying point densities and noise levels, as well as on out-of-distribution data from the ABC dataset Koch et al. (2019), and real-world scans. Across these settings, VISNET consistently outperforms geometric visibility operators (HPR and HPRO Katz et al. (2007); Katz & Tal (2025)) and the recent learning-based baseline NeuVis Wang et al. (2025), with particularly strong gains under challenging sparse inputs.

In summary, our main contributions are:

- *Projection-induced visibility reasoning.* We identify a mismatch between local 3D point relationships and the view-dependent relational structure required for visibility reasoning. VISNET addresses this mismatch by constructing a view-dependent graph from projected neighborhoods, allowing points that are distant in 3D but potentially coupled through occlusion to directly exchange information.
- *Projective geometry-view fusion.* We introduce a learning formulation that combines 3D geometric context with view-specific relational representations induced by projection. The same projected neighborhood structure supports both occlusion reasoning and subsequent geometry-view feature interaction.

2 RELATED WORK

Point-cloud representation learning. Learning directly from raw point clouds requires representations that are permutation invariant while preserving geometric structure. Early approaches converted point clouds into regular volumetric representations Maturana & Scherer (2015), while PointNet Qi et al. (2017a) introduced direct learning on unordered point sets and PointNet++ Qi et al. (2017b) further incorporated hierarchical local neighborhoods. Subsequent methods explored graph-based operators such as DGCNN Wang et al. (2019), continuous convolutions such as KPConv Thomas et al. (2019), and sparse spatial representations including sparse convolutions Choy et al. (2019) and O-CNN Wang et al. (2017). With the emergence of Transformer architecture Vaswani et al. (2017), point-based attention models further improved local and long-range interaction Zhao et al. (2021b);

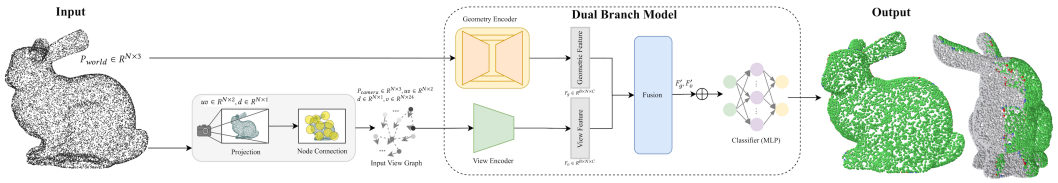

Figure 2: Detailed pipeline of the VISNET framework.

Wu et al. (2022); Lai et al. (2022), while alternative architectures explored MLP-based processing Ma et al. (2022), umbrella structures Ran et al. (2022), curve-based representations Xiang et al. (2021), and self-supervised learning Zha et al. (2024). Recent scalable architectures, such as PointNeXt Qian et al. (2022) and Point Transformer V3 Wu et al. (2024), further improve point-cloud representation learning.

Despite their architectural differences, these methods primarily organize feature extraction according to 3D geometric structure, which does not directly capture the camera-dependent relational structure required for visibility reasoning.

View-dependent and projection-based reasoning. Recent geometric message-passing studies show that neighborhood structure plays an important role in determining how information is exchanged in geometric networks Sverdlov & Dym (2025); Lippmann et al. (2025). For visibility reasoning, however, the relevant relationships are not alone determined by 3D geometry, but also by camera viewpoint. This motivates the use of projection to capture view-dependent interaction between points. In literature, Pittaluga et al. Pittaluga et al. (2019) rasterize SfM points into sparse multi-channel 2D representations for view synthesis, and Vis2Mesh Song et al. (2021) predicts visibility from projected virtual views using a 2D depth-completion network. In both cases, projection primarily serves as a way to transform 3D observations into image-space representations, where interactions are modeled implicitly by 2D convolutional networks.

However, VISNET differs in how it uses projection. Rather than treating the projected point cloud as a rasterized 2D signal, VISNET directly constructs a graph over the original points, with edges determined by proximity in the projected view plane. Projection, therefore, determines which points directly exchange information during message passing.

Point-cloud visibility estimation. Visibility estimation is a fundamental task in computer graphics, particularly for rendering and hidden-surface removal Sutherland et al. (1974); Cohen-Or et al. (2003). For raw point clouds, one strategy is to approximate points using finite-support primitives and apply depth-buffering techniques Guennebaud et al. (2004); Rusinkiewicz & Levoy (2000); Zwicker et al. (2001), while another is to reconstruct an intermediate surface Kazhdan & Hoppe (2013); Parakkat et al. (2024); Hou et al. (2022), and evaluate visibility on the reconstructed geometry. However, surface reconstruction is an ill-posed problem Berger et al. (2017); Sulzer et al. (2024), and reconstruction errors can propagate into the resulting visibility estimates.

At the same time, dedicated point-set methods avoid explicit reconstruction. The most prominent such method, Hidden Point Removal (HPR) Katz et al. (2007), identifies visible points through a viewpoint-dependent transformation and convex-hull construction, with subsequent extensions addressing noisy and incomplete data Mehra et al. (2010), theoretical analysis Katz & Tal (2015), efficiency e Silva et al. (2012), and differentiability Katz & Tal (2025). However, such geometric methods remain sensitive to sampling characteristics and complex local geometry.

Recent learning-based approach NeuVis Wang et al. (2025) provides an alternative by performing direct per-point visibility prediction using a 3D geometric representation conditioned on viewing direction, but its point interactions remain primarily organized by the underlying 3D representation, with viewpoint information serving as a conditioning signal. Our method bridges this gap by explicitly modeling projection-induced point interactions through a view-dependent graph over projected neighborhoods and integrating the resulting view-specific representation with 3D geometric context through projective geometry-view fusion.

3 METHOD

Given a point cloud $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^N$ and a calibrated camera $\Pi = (K, R, \mathbf{t})$, our goal is to predict the subset of points visible from the camera. A point is considered visible if no part of the surface blocks the line segment connecting it to the camera center. Since the underlying surface is unknown, however, we must infer visibility from the discrete point samples alone. This requires jointly reasoning about both the geometric structure of the point cloud and the spatial relationships induced by the camera.

Our overall architecture is shown in Figure 2. It constructs a *projected neighborhood graph* that connects points with nearby view-plane projections, making projection-induced point interactions explicit. The view branch uses this graph to learn view-specific representations through local message passing (Section 3.1). The cross-cue fusion module then combines these representations with the 3D geometric context provided by the geometry branch (Section 3.2) via point-wise cross-cue interactions, followed by neighborhood-level information exchange over the same projected neighborhoods (Section 3.3). The classifier produces per-point visibility predictions (Section 3.4).

3.1 PROJECTION-INDUCED VISIBILITY REASONING

We model projection-induced point interactions using a directed projected neighborhood graph, where the proximity of points in the projected view plane determines candidate interactions, and relative depth determines the direction of information exchange.

Constructing the projected neighborhood graph. Each input point corresponds to a graph node. To define its projected neighborhood, we transform the point into camera coordinates and project it onto the view plane:

$$\mathbf{q}_i = R\mathbf{p}_i + \mathbf{t}, \qquad s_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = K\mathbf{q}_i, \qquad \mathbf{u}_i = (u_i, v_i). \tag{1}$$

Here, $\mathbf{u}_i$ denotes the view-plane projection, s_i is the homogeneous scale factor, and $d_i = |(\mathbf{q}_i)_z|$ denotes the depth. To construct the projected neighborhood graph, we represent each input point as a node and connect it to its k ($k = 20$) nearest neighbors using Euclidean distance between their view-plane projections $\mathbf{u}_i$. This point-based construction preserves explicit point-level interactions without requiring a rasterized intermediate image.

We then orient the edges based on the relative depth from the camera and remove duplicate directed edges. This orientation allows potential background points to receive information from potential foreground points. The resulting direction graph is $\mathcal{G}_\Pi = (\mathcal{V}, \mathcal{E}_\Pi)$, with incoming neighborhoods defined as

$$\mathcal{N}_\Pi^{\text{in}}(i) = \{j \mid (j \to i) \in \mathcal{E}_\Pi\}. \tag{2}$$

For each point i, $\mathcal{N}_\Pi^{\text{in}}(i)$ therefore specifies the set of source points whose information can be aggregated when updating its representation. The directed edges encode potential occlusio n interactions, while the network learns how the connected points contribute to the visibility prediction. We use the directed graph as the default for its accuracy–efficiency trade-off, evaluated in the directed and undirected ablation (see Section 5.2).

Representing view-dependent evidence. We define $\mathbf{r}_i = \mathbf{q}_i / \|\mathbf{q}_i\|_2$ as the unit viewing direction from the camera center toward point i. To help the network model high-frequency variations in visibility with respect to viewing direction, we encode $\mathbf{r}_i$ using a multi-frequency sinusoidal mapping (Mildenhall et al., 2020):

$$\gamma(\mathbf{r}_i) = \big[\sin(2^0\pi\mathbf{r}_i) \| \cos(2^0\pi\mathbf{r}_i) \| \cdots \| \sin(2^{L_\gamma-1}\pi\mathbf{r}_i) \| \cos(2^{L_\gamma-1}\pi\mathbf{r}_i)\big]. \tag{3}$$

Here, $[\cdot\|\cdot]$ denotes feature concatenation. We use $L_\gamma = 4$, yielding a 24-dimensional encoding and a 30-dimensional node input:

$$\mathbf{x}_i^{\text{view}} = [\mathbf{q}_i \| d_i \| \mathbf{u}_i \| \gamma(\mathbf{r}_i)] \in \mathbb{R}^{30}. \tag{4}$$

This node inputs provide information about the spatial arrangement of the projected points and their viewing directions for the subsequent message-passing layers.

Learning view-specific representations. We process these inputs using L_g GENConv-based message-passing blocks (Li et al., 2023) to learn view-representations on the projected neighborhood graph $\mathcal{G}_\Pi$:

$$\begin{aligned} \mathbf{h}_i^{(0)} &= W_{\text{in}}\mathbf{x}_i^{\text{view}} + \mathbf{b}_{\text{in}} \in \mathbb{R}^{d_h}, \\ \mathbf{h}_i^{(\ell)} &= \mathcal{B}_\ell\Big(\mathbf{h}_i^{(\ell-1)}, \{\mathbf{h}_j^{(\ell-1)} : j \in \mathcal{N}_\Pi^{\text{in}}(i)\}\Big), \qquad \ell = 1, \dots, L_g. \end{aligned} \tag{5}$$

Here, $\mathbf{h}i^{(\ell)}$ denotes the representation of point i after the ℓ-th message-passing block, and $\mathcal{B}\ell$ denotes the corresponding GENConv-based block with hidden dimension d_h.Each block aggregates incoming features and updates the d_h-dimensional node representation through residual connections, normalization, and a feed-forward transformation. This propagates evidence about projected positions, depths, and viewing directions along the graph. By stacking multiple blocks, view-dependent information propagates along multi-hop paths in the directed graph, extending each point's context beyond its immediate projected neighbors.

Different message-passing layers provide access to different neighborhood ranges: early layers aggregate features from directly connected neighbors, while later layers receive information propagated along longer paths. To make these intermediate representations available to the view output, we concatenate the outputs of all blocks and apply a learned projection, using concatenation-based layer aggregation (Xu et al., 2018):

$$\mathbf{f}_i^{\text{view}} = \phi_{\text{agg}}\Big([\mathbf{h}_i^{(1)} \| \cdots \| \mathbf{h}_i^{(L_g)}]\Big) \in \mathbb{R}^{d_v}. \tag{6}$$

The resulting view-specific representation summarizes potential occlusion information over projected neighborhoods.

3.2 3D Geometric Context Encoding

The geometry branch provides 3D geometric context to complement the view-specific representations learned over projected neighborhoods. A projected neighborhood can contain points from different surfaces, and view-plane proximity alone does not establish whether the foreground samples form a surface that blocks the target point. Whereas, local surface structure and broader shape context can help resolve this ambiguity, particularly when sampling is sparse. We therefore extract geometric features independently of the camera. Following NeuVis Wang et al. (2025)'s setting, we employ O-CNN Wang et al. (2017) as the geometric backbone, allowing improvements to be attributed primarily to *projection-induced reasoning* and *projective geometry-view fusion* rather than to a stronger 3D encoder:

$$\mathbf{f}_i^{\text{geom}} = E_{\text{geom}}(\mathcal{P})_i \in \mathbb{R}^{d_g}. \tag{7}$$

These features describe the point cloud's geometry without conditioning on the camera. They can therefore be reused across viewpoints for an unchanged point cloud. And these features complement the camera-dependent evidence from the view branch and are subsequently integrated through projective geometry-view fusion.

3.3 Projective Geometry-View Fusion

The two branches process information in complementary ways: the geometry branch encodes 3D geometric context, while the view branch aggregates view-specific features over projected neighborhoods. Our cross-cue fusion module integrates them at two levels. Point-wise cross-cue interaction first combines the features for each point. Neighborhood-level information exchange then propagates the fused features across the projected neighborhood graph, enabling visibility predictions to leverage geometric context from potential occluders as well as from the target point itself.

Point-wise cross-cue interaction. We first project the geometric and view-dependent features in a common d_f-dimensional feature space:

$$\mathbf{h}_i^{\text{geom}} = \phi_g(\mathbf{f}_i^{\text{geom}}), \qquad \mathbf{h}_i^{\text{view}} = \phi_v(\mathbf{f}_i^{\text{view}}), \tag{8}$$

where $\phi_g : \mathbb{R}^{d_g} \to \mathbb{R}^{d_f}$ and $\phi_v : \mathbb{R}^{d_v} \to \mathbb{R}^{d_f}$ are MLP projections. We then compute a channel-wise gate from both cues and use it to modulate a geometric update to the view feature:

$$\begin{aligned} \boldsymbol{\alpha}_i &= \sigma\big(\phi_{\text{gate}}([\mathbf{h}_i^{\text{view}} \| \mathbf{h}_i^{\text{geom}}])\big), \\ \mathbf{h}_i^{\text{fuse}} &= \mathbf{h}_i^{\text{view}} + \boldsymbol{\alpha}_i \odot \mathbf{h}_i^{\text{geom}}. \end{aligned} \tag{9}$$

Here $\phi_{\text{gate}} : \mathbb{R}^{2d_f} \to \mathbb{R}^{d_f}$ is an MLP, σ is the sigmoid function, and $\odot$ denotes element-wise multiplication. We also include a direct path from the geometry feature when computing the point-wise fused representation:

$$\bar{\mathbf{h}}_i = \phi_{\text{fuse}}([\mathbf{h}_i^{\text{fuse}} \| \mathbf{h}_i^{\text{geom}}]) \in \mathbb{R}^{d_f}, \tag{10}$$

where $\phi_{\text{fuse}} : \mathbb{R}^{2d_f} \to \mathbb{R}^{d_f}$ is an MLP. This direct path allows the fusion MLP to access the geometric feature independently of the gate values in the residual update.

Neighborhood-level information exchange. Point-wise fusion incorporates geometry into each node's representation, but does not itself exchange these updated features between nodes. We therefore apply a local Point Transformer layer (Zhao et al., 2021a) over the same projected neighborhood graph:

$$\mathbf{f}_i^{\text{joint}} = \mathcal{T}\big(\bar{\mathbf{h}}_i, \mathbf{p}_i, \{(\bar{\mathbf{h}}_j, \mathbf{p}_j) : j \in \mathcal{N}_\Pi^{\text{in}}(i)\}\big) \in \mathbb{R}^{d_f}. \tag{11}$$

The operator computes local attention from the fused features and relative 3D positions using the same graph $\mathcal{G}_\Pi$. The earlier view-branch updates process view-dependent evidence, and this stage exchanges representations that also contain 3D geometric context. It produces the joint features used to predict visibility.

3.4 Visibility Prediction and Training

The joint representation is mapped to a scalar visibility logit by a classifier MLP with widths $d_f \to 128 \to 64 \to 1$, converting the joint representation into a visibility probability:

$$z_i = \phi_{\text{cls}}(\mathbf{f}_i^{\text{joint}}), \qquad \hat{p}_i = \sigma(z_i). \tag{12}$$

LayerNorm and ReLU follow each hidden linear layer. Finally, we obtain binary visibility predictions by thresholding $\hat{p}_i$ at 0.5.

We train the network using binary cross-entropy loss applied to the predicted visibility logits.

4 Dataset Preparation

We construct our dataset from 37,995 raw ShapeNet meshes Chang et al. (2015), which are preprocessed to remove topological artifacts such as degenerate faces, duplicated vertices, and inconsistent normals. During training, surface points are randomly sampled from these clean meshes at varying resolutions (1k, 2k, 8k, 16k and 32k). Camera poses are drawn from a uniform distribution over a viewing hemisphere centered at the object. Each object is annotated from 24 calibrated viewpoints, with ground-truth visibility labels obtained via GPU-accelerated ray casting. To simulate real-world scans, we apply online geometric augmentations during dataset generation, including random rotations, scaling, and point dropout. Specifically, we split the dataset into 30,396 models for training, and allot 3,799 each for validation and testing. In total, the dataset contains 911,880 point-view visibility labels. *We will release the complete implementation, including all code, weights, the dataset, and the full data generation pipeline upon acceptance.*

5 Results & Discussion

We evaluate VISNET from three perspectives: how well our method compares with existing SOTA methods, the contribution of its architectural components, and generalization to out-of-distribution datasets and varying point cloud densities and noisy inputs.

5.1 Comparative Analysis

We compared VISNET against three baseline methods: the classic Hidden Point Removal (HPR) operator Katz et al. (2007), the optimized HPRO Katz & Tal (2025), and the recent learning-based method NeuVis Wang et al. (2025). *Note that, to ensure a fair comparison, we retrained Wang et al. (2025) on our training dataset prior to evaluation.*

As shown in Table 1, VISNET consistently outperforms all baselines across all reported metrics and sampling densities (1k to 32k). At the sparse resolutions of 1k and 2k points, our method achieves

Table 1: Visibility prediction on ShapeNet Chang et al. (2015) across varying point densities.

Method	Accuracy	Visible IoU	Occluded IoU	mIoU	F1 (Visible)	Macro-F1	Balanced Accuracy
HPR (1k)	84.6	62.52	79.22	70.87	76.94	82.67	84.45
HPRO (1k)	75.5	48.98	67.91	58.44	65.76	73.32	75.90
NeuVis (1k)	94.7	84.05	92.65	88.35	91.33	93.76	93.74
Ours (1k)	**96.2**	**88.07**	**94.71**	**91.39**	**93.66**	**95.47**	**94.97**
HPR (2k)	88.1	67.38	84.14	75.76	80.51	85.95	86.11
HPRO (2k)	79.5	51.18	69.25	60.21	67.71	74.77	77.71
NeuVis (2k)	93.3	80.47	90.69	85.58	89.18	92.15	92.69
Ours (2k)	**96.7**	**89.57**	**95.44**	**92.50**	**94.50**	**96.08**	**95.56**
HPR (8k)	91.5	74.69	88.67	81.68	85.51	89.75	89.01
HPRO (8k)	83.7	57.75	79.58	68.66	73.21	80.92	80.62
NeuVis (8k)	96.0	87.29	94.26	90.77	93.21	95.13	95.14
Ours (8k)	**97.4**	**91.49**	**96.27**	**93.88**	**95.56**	**96.83**	**96.51**
HPR (16k)	92.4	78.53	90.00	84.27	87.97	91.36	91.39
HPRO (16k)	85.1	58.72	81.81	70.26	73.99	81.99	80.50
NeuVis (16k)	96.4	88.33	94.74	91.53	93.80	95.55	95.50
Ours (16k)	**97.8**	**92.94**	**96.88**	**94.91**	**96.34**	**97.38**	**97.22**
HPR (32k)	93.7	81.47	91.86	86.67	89.79	92.77	91.96
HPRO (32k)	85.4	55.69	82.30	69.00	71.54	80.92	78.30
NeuVis (32k)	96.5	88.53	94.87	91.70	93.92	95.64	95.47
Ours (32k)	**98.0**	**93.73**	**97.22**	**95.48**	**96.76**	**97.68**	**97.61**

Table 2: Accuracy comparison under different noise levels.

Method	0%	0.5%	1%	Method	0%	0.5%	1%
HPR (2k)	88.4	87.3	86.6	HPR (16k)	91.5	89.6	87.9
HPRO (2k)	85.5	84.4	83.0	HPRO (16k)	86.4	85.0	83.9
NeuVis (2k)	91.4	88.9	85.1	NeuVis (16k)	95.0	93.3	85.8
Ours (2k)	**92.6**	**92.3**	**89.1**	**Ours (16k)**	**96.5**	**95.1**	**88.5**
HPR (8k)	90.8	89.1	87.5	HPR (32k)	85.6	83.7	82.6
HPRO (8k)	87.7	86.1	85.6	HPRO (32k)	85.2	83.2	82.6
NeuVis (8k)	94.2	92.5	86.5	NeuVis (32k)	95.8	93.4	86.7
Ours (8k)	**95.3**	**94.3**	**89.3**	**Ours (32k)**	**97.4**	**95.9**	**89.1**

Table 3: Ablation study on the impact of the Geometry branch, View branch, and Fusion.

Method	2k	8k	16k	32k
W/o Fusion module	95.6	95.7	96.4	96.8
W/o Geometry branch	87.4	90.7	91.9	92.6
W/o View branch	91.3	90.8	91.2	91.6
VISNET	**96.7**	**97.4**	**97.8**	**98.0**

accuracies of 96.2% and 96.7%, compared with 94.7% and 93.3% for NeuVis, respectively. This corresponds to improvements of 1.5 and 3.4 percentage points in accuracy, 3.04 and 6.92 percentage points in mIoU, and 1.23 and 2.87 percentage points in balanced accuracy, respectively. These results highlight VISNET's effectiveness when sparse surface samples provide limited geometric information for visibility prediction. The per-category results in Appendix Table 4 further show consistent improvements across object categories. Appendix Figure 13 provides qualitative results on ShapeNet across 2k–32k inputs, with alternative views and close-ups highlighting visibility predictions around fine structures and occlusion boundaries. Qualitatively, Appendix Figure 5 illustrates that baseline methods frequently suffer from false positives (red points) on complex topologies, such as the inner loops of the *Dragon*. In contrast, VISNET maintains high fidelity at boundaries, minimizing both false positives and false negatives (red and blue points, respectively).

5.2 ABLATION STUDY

To evaluate the contribution of each component in our architecture, we conducted an ablation study (Table 3). The core hypothesis of VISNET is that visibility inference requires both 3D geometric context and 2D projective evidence.

Directed vs. undirected graph. We first compare the directed projected neighborhood graph with an undirected variant that allows bidirectional message passing between projected neighbors. As shown in Appendix Table 5, the undirected variant improves all reported metrics at 2k points, including an increase in mIoU from 92.50% to 93.21%. However, this improvement is not consistent across densities: the directed graph achieves higher mIoU, visible F1, and Macro-F1 at 8k, 16k, and 32k points. Averaged over the four densities, the undirected variant increases accuracy from 97.48% to 97.55%, a gain of approximately 0.08 percentage points. Its inference latency is 22.96–69.37% higher over 2k–32k inputs and 150.82% higher at 200k points (Appendix Figure 4). Given the weak improvement in predictive performance and the consistently higher inference cost of undirected graph, we use the directed graph as the default configuration of VisNet.

Impact of the Geometry Branch: We then study the contribution of the geometry branch by removing the geometry encoder from VisNet. As shown in the "W/o Geometry branch" row, performance drops significantly across all densties (e.g., from 96.7% to 87.4% at 2k). This confirms that the local geometric features are essential for visibility prediction, especially when the input point

cloud is sparse, which provides surface-level information such as local neighborhoods and shape structures to resolve ambiguities that the view-graph alone cannot handle.

Impact of the Fusion Module: We further evaluated the efficacy of our fusion module by replacing it with simple concatenation. The "W/o Fusion module" setting results in a notable performance degradation at all densities(e.g., from 97.4% to 95.7% at 8k). Although the degradation is smaller than removing either the geometry branch, the consistent gap demonstrates that properly integrating geometric and view-dependent features is important and simple feature concatenation is insufficient.

Impact of Channel dimension: Appendix Table 8 compares how the feature dimension C (=32 or =64) affects the trade-off between accuracy and model size. Increasing C consistently improves performance at all densities, though the model size grows from 10.0MB to 39.9MB: for instance, accuracy increases, indicating that a larger hidden representation better captures occlusion relations and fine-grained geometric cues. To maximize accuracy, we use $C = 64$.

5.3 Robustness Analysis

Finally, we assess VisNet's robustness to out-of-distribution data and its scalability to dense point clouds.

Out-of-Distribution Performance: We evaluated the model on the ABC Dataset (Appendix Table 7), which contains mechanical parts significantly different from the training categories. VisNet achieves the highest accuracy across all densities, demonstrating that the model generalises well rather than overfitting to specific object classes. Appendix Figure 6 further compares VisNet with NeuVis on the out-of-distribution Lucy model at 64k points, where VisNet produces a cleaner separation between foreground and background regions. Appendix Figure 9 further presents qualitative results on multi-object scenes from Thingi10K. These scenes contain varying numbers of disconnected components, making visibility estimation more challenging due to inter-object occlusions and complex spatial layouts. VisNet maintains accurate and coherent predictions across these cases, as shown by the high per-scene accuracies of 96.13%–97.99%. More qualitative predictions of out-of-distribution input visibility are provided in the supplementary material. While NeuVis is a strong baseline and outperforms HPR and HPRO across all densities, VisNet consistently outperforms NeuVis by a consistent gain of 1.2–2.6 percentage increase. In addition, VisNet surpasses the non-learning-based HPR method by 3.6–5.1 percentage points, highlighting the effectiveness of our learned geometric-view priors. More qualitative predictions of out-of-distribution input visibility are provided in the supplementary material.

Robustness to Input Noise and Real Scans: We further evaluated robustness in real-world scenarios by simulating Kinect scans using Blensor Gschwandtner et al. (2011). We generated scans with 0%, 0.5%, and 1% noise levels (see Appendix Figure 10 for examples) across varying point cloud densities. As shown in Table 2, VisNet consistently outperformed every alternative for noisy inputs. For instance, the gain over the second-best method, NeuVis, is 1.2%, 1.1%, 1.5%, and 1.6% at densities of 2k, 8k, 16k, and 32k points, respectively; notably, maintaining this edge despite density fluctuation caused by overlapping scan views. More importantly, the advantage becomes more pronounced under stronger perturbations: at a 1% noise level, VisNet outperforms NeuVis by 4.0%, 2.8%, 2.7%, and 2.4% across the four densities. We also evaluate both methods on raw point clouds scanned using an EinScan Pro HD scanner. As shown in Appendix Figure 7, these scans contain non-uniform sampling, missing regions, and complex self-occlusions that are not present in the synthetic training data. Compared with NeuVis, VisNet produces more spatially coherent visibility predictions. The alternative-view and close-up visualizations further show that VisNet better separates visible foreground surfaces from occluded geometry while reducing fragmented predictions. These results demonstrate that projective geometry-view fusion generalizes beyond synthetic inputs and remains robust under both simulated noise and real scanning artifacts.

5.4 Performance and Applications

Scalability and Efficiency: As shown in Appendix Figure 8, VisNet scales well from 2k to 128k points without retraining, despite being trained only up to 32k. Appendix Figure 14 further shows

qualitative results at 64k and 128k points, illustrating generalization to input densities beyond the training range.

The computational scalability is attributed to two complementary design choices: the geometry branch is bounded by the octree depth rather than by n, while the view branch and cross-cue fusion perform local message passing over a projected neighborhood graph with $\mathcal{O}(nk)$ edges, where k is the fixed neighborhood size, yielding an overall $\mathcal{O}(nk)$ computational complexity for graph processing after graph construction, with fixed feature widths and layer counts. Nearest-neighbor search, octree construction, and geometric feature extraction contribute additional costs to end-to-end inference. In contrast, HPR Katz et al. (2007) and HPRO Katz & Tal (2025) rely on convex-hull computation, which remains time-consuming and sensitive to point density. As shown in Figure 3, VISNET reaches 239 ms at 200k points, well beyond our 32k training regime, placing it in an interactive inference time regime alongside NeuVis Wang et al. (2025) while achieving consistently higher accuracy across all densities.

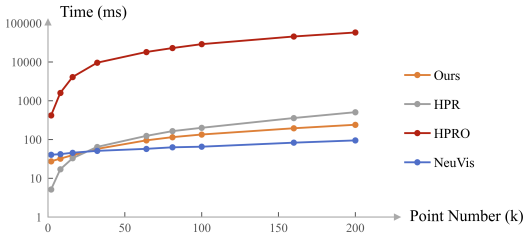

Figure 3: Efficiency comparison. VISNET achieves a favorable accuracy-efficiency trade-off, maintaining competitive inference time while consistently achieving the highest accuracy across varying point densities.

Downstream Applications: Accurate point visibility prediction supports best-view selection, view-dependent surface reconstruction, and shadow casting. Appendix Figure 11 shows surface reconstruction results using visibility predictions from VISNET across input sizes ranging from 1k to 32k points, with finer geometric details recovered as the sampling density increases. We also demonstrate a 2D-to-3D annotation framework that maps user-drawn sketches onto a point cloud by associating each sketch sample with the predicted visible point nearest to it in the projected view plane. As shown in Appendix Figure 12, this transfers the sketch onto the foreground surface without assigning annotations to occluded background geometry. Potential applications include marking incision paths on patient scans, annotating industrial feature lines, and stylizing point clouds.

6 CONCLUSION

We presented VISNET, a framework for per-point visibility prediction based on *projective geometry-view fusion*. Its central design is a projected neighborhood graph that makes projection-induced point interactions explicit, allowing the view branch to learn view-specific representations of potential occlusion. The cross-cue fusion module integrates these representations with 3D geometric context via pointwise cross-cue interactions and neighborhood-level information exchange. Extensive experiments demonstrate that VISNET substantially outperforms traditional geometric heuristics (HPR/HPRO) and state-of-the-art learning-based baseline Wang et al. (2025), especially on challenging sparse inputs, noisy real-world scans, and out-of-distribution shapes.

Limitations & Future Work: Despite its improved performance, VISNET offers several avenues for future research. While our current model combines 3D geometric context with view-specific representations, incorporating additional inputs such as point color, texture, or semantic information could provide valuable semantic and continuity cues. And, while VISNET performs well under a given static configuration, ensuring temporal consistency across dynamic walk-through scenarios remains an open challenge.

AI USE STATEMENT

In this work, we used generative AI tools solely for assisting with the writing and polishing of this manuscript, including grammar improvement, clarity and readability. We have not used generative AI tools for getting the research idea, developing the methodology, designing/conducting all the experiments, analyzing the data, or drawing the conclusions. All AI-assisted revisions were reviewed

and verified by the authors. And all research ideas, technical contributions, experimental results, and interpretations were developed and verified by the authors.

The authors carefully reviewed and edited all AI-assisted content and take full responsibility for the accuracy, originality, and integrity of the manuscript.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. Our study focuses on visibility prediction for 3D point clouds and does not involve human participants, animal subjects, or any personally sensitive information. The experiments use publicly available 3D datasets, including ShapeNet for dataset construction and model evaluation, ABC and Thingi10K for out-of-distribution evaluation, and publicly available real-world object scans for qualitative evaluation. Ground-truth visibility labels are generated automatically through ray casting; therefore, no human annotation or crowd-sourced labor is involved.

We use these datasets for research purposes and acknowledge their original creators. Any release of processed data will respect the licenses and terms of use of the corresponding source datasets.

The proposed method is intended for general-purpose geometric processing applications, such as visibility estimation, surface reconstruction, and view-dependent rendering. It is not designed for identifying individuals or making decisions about people. The authors take full responsibility for the integrity of the research and the claims presented in this paper.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the dataset construction, model architecture, training objective, and evaluation protocol to support the reproduction of our results. Appendix A describes the complete data-preprocessing procedure, including mesh cleaning, multi-resolution surface sampling, camera-pose generation, and the ray-casting process used to generate point-wise visibility labels. Algorithm 1 summarizes this pipeline, while Algorithm 2 specifies the forward computation of VISNET, including projected-graph construction, geometric feature extraction, cross-cue fusion, and visibility prediction.

Our implementation is based on PyTorch, O-CNN, and PyTorch Geometric. The model is trained on NVIDIA A100 GPUs using the Adam optimizer and binary cross-entropy loss, and visibility probabilities are thresholded at 0.5 during inference. We report results across multiple point-cloud densities and include architectural, graph-direction, and feature-width ablations. For fair comparison, the learning-based NeuVis baseline is retrained using the same training dataset. The experiments use publicly available datasets, including ShapeNet Chang et al. (2015) for training and in-distribution evaluation, and ABC Koch et al. (2019) and Thingi10K Zhou & Jacobson (2016) for out-of-distribution evaluation.

Upon acceptance, we will release the source code, pretrained checkpoints, training dataset, environment specifications, data-generation pipeline, and experiment configurations required to reproduce the reported results.

REFERENCES

Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. In *Computer graphics forum*, volume 36, pp. 301–329. Wiley Online Library, 2017.

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.

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3075–3084, 2019.

Daniel Cohen-Or, Yiorgos L Chrysanthou, Claudio T. Silva, and Frédo Durand. A survey of visibility for walkthrough applications. *IEEE Transactions on Visualization and Computer Graphics*, 9(3): 412–431, 2003.

Renan Machado e Silva, Claudio Esperança, and Antonio Oliveira. Efficient hpr-based rendering of point clouds. In *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 126–133. IEEE, 2012.

Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: Blender sensor simulation toolbox. In *International Symposium on Visual Computing*, pp. 199–208. Springer, 2011.

Gael Guennebaud, Loïc Barthe, and Mathias Paulin. Deferred splatting. In *Computer Graphics Forum*, volume 23, pp. 653–660. Wiley Online Library, 2004.

Fei Hou, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He. Iterative poisson surface reconstruction (ipsr) for unoriented points. *ACM Trans. Graph.*, 41(4), July 2022. ISSN 0730-0301.

Sagi Katz and Ayellet Tal. On the visibility of point clouds. In *Proceedings of the IEEE international conference on computer vision*, pp. 1350–1358, 2015.

Sagi Katz and Ayellet Tal. Hpro: Direct visibility of point clouds for optimization. In *Computer Graphics Forum*, pp. e70046. Wiley Online Library, 2025.

Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. *ACM Transactions on Graphics (TOG)*, 26(3):24, 2007.

Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013.

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9601–9611, 2019.

Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8500–8509, 2022.

Guohao Li, Chenxin Xiong, Guocheng Qian, Ali Thabet, and Bernard Ghanem. Deepergcn: Training deeper gcns with generalized aggregation functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13024–13034, 2023.

Peter Lippmann, Gerrit Gerhartz, Roman Remme, and Fred A Hamprecht. Beyond canonicalization: How tensorial messages improve equivariant message passing. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 88067–88087, 2025.

Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022.

Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 922–928. Ieee, 2015.

Ravish Mehra, Pushkar Tripathi, Alla Sheffer, and Niloy J Mitra. Visibility of noisy point cloud data. *Computers & Graphics*, 34(3):219–230, 2010.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 405–421, 2020.

Amal Dev Parakkat, Stefan Ohrhallinger, Elmar Eisemann, and Pooran Memari. Ballmerge: High-quality fast surface reconstruction via voronoi balls. In *Computer Graphics Forum*, volume 43, pp. e15019. Wiley Online Library, 2024.

Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 145–154, 2019.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 652–660, 2017a.

Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5099–5108, 2017b.

Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022.

Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18942–18952, 2022.

Szymon Rusinkiewicz and Marc Levoy. Qsplat: A multiresolution point rendering system for large meshes. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 343–352, 2000.

Shuang Song, Zhaopeng Cui, and Rongjun Qin. Vis2mesh: Efficient mesh reconstruction from unstructured point clouds of large scenes with learned virtual view visibility. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6514–6524, 2021.

Raphael Sulzer, Renaud Marlet, Bruno Vallet, and Loic Landrieu. A survey and benchmark of automatic surface reconstruction from point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Ivan E Sutherland, Robert F Sproull, and Robert A Schumacker. A characterization of ten hidden-surface algorithms. *ACM Computing Surveys (CSUR)*, 6(1):1–55, 1974.

Yonatan Sverdlov and Nadav Dym. On the expressive power of sparse geometric MPNNs. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 43822–43849, 2025.

Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 6411–6420, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jun-Hao Wang, Yi-Yang Tian, Baoquan Chen, and Peng-Shuai Wang. Neural visibility of point sets. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, SA Conference Papers '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400721373.

Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics*, 36(4):1–11, 2017. doi: 10.1145/3072959.3073608.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin Solomon. Dynamic graph cnn for learning on point clouds. In *ACM SIGGRAPH*, pp. 1–12, 2019.

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4840–4851, June 2024.

Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 915–924, 2021.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pp. 5453–5462. PMLR, 2018.

Huizong Yang, Yuxin Sun, Ganesh Sundaramoorthi, and Anthony Yezzi. Steik: Stabilizing the optimization of neural signed distance functions and finer shape representation. *Advances in neural information processing systems*, 36:13993–14004, 2023.

Yaohua Zha, Naiqi Li, Yanzi Wang, Tao Dai, Hang Guo, Bin Chen, Zhi Wang, Zhihao Ouyang, and Shu-Tao Xia. Lcm: Locally constrained compact point cloud model for masked point modeling. *Advances in Neural Information Processing Systems*, 37:104816–104842, 2024.

Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Point transformer. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 16259–16268, 2021a.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16259–16268, 2021b.

Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.

Taiki Yoshino Sirui Tao Ziyang Fu Zimo Wang, Cheng Wang and Tzu-Mao Li. Hotspot: Signed distance function optimization with an asymptotically sufficient condition. In *CVPR*, 2025.

Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 371–378, 2001.

A Implementation and Reproducibility

Dataset preprocessing. Algorithm 1 describes the construction of our visibility dataset. We uniformly sample points over mesh surfaces and obtain point-wise visibility labels by comparing the first ray–mesh intersection distance with the distance from the camera to each sampled point. Lower-resolution point clouds are obtained by random subsampling, with the same indices applied to their visibility labels.

Algorithm 1 Visibility dataset preprocessing

Require: Mesh collection $\mathcal{M}$; dense sample count N_0; target point counts $\mathcal{N}$ with $n < N_0$; number of views V; camera-direction domain Ω; camera distance factor ρ; intrinsics K; ray-intersection tolerance ϵ.

Ensure: Multi-resolution point clouds, camera parameters, and point-wise visibility labels.

for all meshes $M \in \mathcal{M}$ **do**
  Remove degenerate and duplicate triangles and duplicate vertices from M
  $P = \{\mathbf{p}_i\}_{i=1}^{N_0} \leftarrow \mathrm{UniformSurfaceSample}(M, N_0)$
  $\mathbf{c} \leftarrow \frac{1}{N_0} \sum_{i=1}^{N_0} \mathbf{p}_i$
  $r \leftarrow \max_i \|\mathbf{p}_i - \mathbf{c}\|_2$
  Build a ray-casting scene from M
  for $v = 1, \ldots, V$ **do**
    repeat
      $\mathbf{a} \sim \mathcal{N}(\mathbf{0}, I_3)$
      $\mathbf{q} \leftarrow \mathbf{a}/\|\mathbf{a}\|_2$
    until $\mathbf{q} \in \Omega$
    $C_v \leftarrow \mathbf{c} + \rho r \mathbf{q}$
    $T_v \leftarrow \mathrm{LookAt}(C_v, \mathbf{c})$ {Camera-to-world pose}
    for $i = 1, \ldots, N_0$ **do**
      $\ell_{iv} \leftarrow \|\mathbf{p}_i - C_v\|_2$
      $\mathbf{d}_{iv} \leftarrow (\mathbf{p}_i - C_v)/\ell_{iv}$
      $t_{iv} \leftarrow \mathrm{FirstHit}(M, C_v, \mathbf{d}_{iv})$
      $Y_{iv} \leftarrow \mathbf{1}[t_{iv} < \infty \ \wedge \ |t_{iv} - \ell_{iv}| < \epsilon]$
    end for
  end for
  for all $n \in \mathcal{N}$ **do**
    $I_n \leftarrow \mathrm{RandomSubset}(\{1, \ldots, N_0\}, n)$ {Without replacement}
    Save $P[I_n]$, $Y[I_n, :]$, $\{T_v\}_{v=1}^{V}$, K, $\mathbf{c}$, and r
  end for
end for

Implementation. VisNet is implemented using PyTorch, O-CNN, and PyTorch Geometric. Algorithm 2 specifies the forward computation. The geometry encoder is an octree U-Net whose features are interpolated to the input points. The projective branch uses GENConv-based residual blocks, followed by gated geometry fusion and a PointTransformerConv-based residual block. Surface normals are not required as network inputs.

Block definitions. Here, $[\cdot, \cdot]$ denotes feature concatenation and $\odot$ denotes element-wise multiplication. Each residual GEN block applies GENConv with softmax aggregation and a learnable temperature, followed by a residual connection and graph-wise layer normalization. A two-layer feed-forward network with GELU, dropout, and a second normalized residual connection completes the block. The point-transformer block follows the same residual structure, using camera-space positions for positional encoding. The classifier is a three-layer MLP with layer normalization and ReLU after its first two layers. Training is performed on NVIDIA A100 GPUs using Adam. We supervise point-wise visibility predictions using binary cross-entropy:

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \left[y_i \log p_i + (1 - y_i) \log(1 - p_i) \right],$$

Algorithm 2 VisNet forward computation

Require: Point cloud $P = \{\mathbf{p}_i\}_{i=1}^{N}$; camera intrinsics K and camera-to-world pose (R, C); object center $\mathbf{c}$ and radius r; neighborhood size k; message-passing depth L.
Ensure: Point-wise visibility probabilities $\{s_i\}_{i=1}^{N}$.

$\tilde{P} \leftarrow (P - \mathbf{c})/r$; $\tilde{C} \leftarrow (C - \mathbf{c})/r$
$(X, U, d) \leftarrow \mathrm{CameraProjection}(\tilde{P}, K, R, \tilde{C})$ {Camera coordinates, image coordinates, and depth}
$\mathbf{v}_i \leftarrow (\tilde{\mathbf{p}}_i - \tilde{C})/\|\tilde{\mathbf{p}}_i - \tilde{C}\|_2$
$E_0 \leftarrow \mathrm{kNN}(U, k)$ {Exclude self-neighbors}
$E \leftarrow \varnothing$
for all $(i, j) \in E_0$ **do**
  if $d_i \le d_j$ **then**
    $E \leftarrow E \cup \{(i, j)\}$
  else
    $E \leftarrow E \cup \{(j, i)\}$
  end if
end for
$E \leftarrow \mathrm{RemoveDuplicates}(E)$ {Near-to-far message flow}
$G \leftarrow \mathrm{OctreeUNet}(\tilde{P})$ {Interpolated per-point geometry features}
$H^{(0)} \leftarrow \mathrm{Linear}([X, d, U, \gamma(V)])$ {Sinusoidal direction encoding γ}
for $\ell = 1, \ldots, L$ **do**
  $H^{(\ell)} \leftarrow \mathrm{ResidualGENBlock}_\ell(H^{(\ell-1)}, E)$
end for
$H_p \leftarrow \mathrm{Linear}([H^{(1)}, \ldots, H^{(L)}])$
$H_g \leftarrow \mathrm{MLP}_g(G)$
$A \leftarrow \sigma(\mathrm{MLP}_a([H_p, H_g]))$
$H \leftarrow H_p + A \odot H_g$
$H \leftarrow \mathrm{MLP}_f([H, H_g])$
$H \leftarrow \mathrm{ResidualPointTransformer}(H, X, E)$
$s \leftarrow \sigma(\mathrm{Classifier}(H))$
return s

where $\mathcal{B}$ denotes the point–view pairs in a minibatch, $y_i \in \{0, 1\}$ is the ground-truth visibility label, and p_i is the predicted visibility probability. At inference, visibility probabilities are thresholded at 0.5.

Code availability. Upon acceptance, we will release the implementation, pretrained checkpoints, dataset for training, environment specifications, and data-processing and experiment configurations.

B Results & Additional Experiments

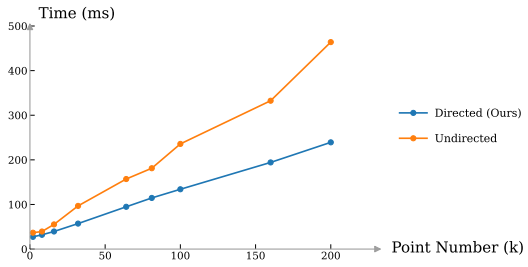

Figure 4: Runtime comparison between the directed and undirected graph variants across different point cloud densities. The directed formulation consistently requires less inference time, with the efficiency gap becoming more pronounced as the number of input points increases.

Table 4: Quantitative comparison on ShapeNet Chang et al. (2015) across 1k–32k input points. We report mean and per-category visibility accuracy (higher is better), with the best results at each density highlighted in bold. VISNET achieves the highest accuracy across all 18 categories at every evaluated density, outperforming HPR, HPRO, and NeuVis. Across all evaluated densities, the improvement over NeuVis ranges from 1.4 to 3.4 percentage points, with the largest gain occurring at 2k points, demonstrating consistent gains across object categories and sampling densities.

Category	**Mean**	Airplane	Bag	Bathtub	Bed	Bench	Bottle	Cabinet	Car	Chair	Display	Lamp	Speaker	Pillow	Rifle	Sofa	Table	Phone	Ship
HPR (1k)	84.6	82.2	87.8	84.2	82.1	77.7	91.9	86.7	86.6	83.6	85.9	83.1	90.6	94.1	86.3	89.2	81.8	91.0	86.8
HPRO (1k)	75.5	75.8	78.2	76.3	72.6	70.3	85.1	75.7	77.9	75.2	73.9	77.1	78.9	84.2	79.1	77.9	71.2	75.2	79.4
NeuVis (1k)	94.7	93.2	95.2	95.0	93.8	90.6	96.6	98.4	94.7	93.9	96.4	91.5	97.5	96.5	93.2	97.2	95.6	98.3	93.4
Ours (1k)	**96.2**	**95.0**	**96.5**	**96.9**	**95.7**	**92.8**	**97.2**	**98.8**	**95.4**	**96.0**	**97.9**	**93.9**	**98.1**	**97.1**	**94.5**	**97.8**	**97.3**	**98.6**	**95.0**
HPR (2k)	88.1	85.1	90.5	87.0	86.9	81.8	94.2	90.4	90.4	86.6	89.7	85.3	93.6	95.7	88.4	92.4	86.3	94.7	88.6
HPRO (2k)	79.5	77.2	80.6	78.6	81.8	76.0	81.5	84.0	87.2	79.1	78.7	74.4	84.1	80.1	77.8	84.5	76.6	78.6	79.7
Neuvis(2k)	93.3	92.4	92.8	92.6	89.7	89.3	95.4	94.7	93.4	91.6	94.9	90.2	94.2	95.7	93.0	96.0	94.0	97.7	92.7
Ours (2k)	**96.7**	**95.4**	**96.9**	**97.4**	**96.8**	**93.4**	**97.4**	**99.0**	**96.2**	**96.6**	**98.3**	**94.5**	**98.6**	**97.5**	**94.8**	**98.2**	**97.7**	**98.9**	**95.5**
HPR (8k)	91.5	88.9	92.9	90.9	90.7	86.4	96.2	93.9	93.1	90.2	93.1	88.9	95.8	96.7	90.9	94.8	90.9	96.9	91.4
HPRO (8k)	83.7	82.2	84.0	83.7	84.1	80.0	87.1	85.8	88.9	83.7	83.7	80.1	87.5	86.2	83.1	88.0	81.1	84.3	84.8
Neuvis(8k)	96.0	94.5	96.1	96.4	95.4	92.8	97.6	98.7	95.9	95.2	97.3	93.2	98.2	97.6	95.0	98.0	96.7	98.7	95.1
Ours (8k)	**97.4**	**96.1**	**97.7**	**97.8**	**97.4**	**94.8**	**98.0**	**99.0**	**96.6**	**97.4**	**98.4**	**95.7**	**98.8**	**98.1**	**96.0**	**98.6**	**98.2**	**98.9**	**96.5**
HPR (16k)	92.4	89.8	93.6	92.1	91.6	87.4	96.9	95.7	93.9	90.9	93.9	89.7	96.6	97.0	91.0	95.4	92.2	97.4	91.6
HPRO (16k)	85.1	83.6	87.1	85.2	84.8	81.5	88.3	86.7	89.4	85.3	85.6	81.4	88.8	88.7	83.8	89.1	82.8	86.0	86.1
Neuvis(16k)	96.4	94.8	96.8	96.8	95.9	93.4	97.9	99.0	96.3	95.7	97.6	93.6	98.4	97.9	95.3	98.2	97.0	99.0	95.5
Ours (16k)	**97.8**	**96.8**	**97.9**	**98.2**	**97.9**	**95.6**	**98.3**	**99.2**	**97.0**	**97.9**	**98.6**	**96.5**	**99.0**	**98.5**	**96.6**	**98.8**	**98.6**	**99.1**	**97.2**
HPR (32k)	93.7	91.9	94.7	92.8	91.8	89.3	97.0	94.1	94.0	93.5	94.9	92.4	96.4	97.6	94.1	96.2	93.1	97.2	93.8
HPRO (32k)	85.4	83.0	86.1	85.6	86.6	82.3	87.0	88.4	91.1	85.8	86.4	79.6	89.1	86.2	81.6	90.0	84.1	85.5	85.1
Neuvis(32k)	96.5	94.8	97.2	96.7	96.0	93.4	98.0	99.1	96.7	95.9	97.6	93.7	98.6	98.1	95.5	98.4	97.1	99.0	95.6
Ours (32k)	**98.0**	**97.1**	**98.4**	**98.3**	**98.1**	**96.0**	**98.5**	**99.3**	**97.4**	**98.1**	**98.7**	**96.9**	**99.1**	**98.6**	**97**	**98.9**	**98.7**	**99.1**	**97.5**

Table 5: Graph direction ablation on ShapeNet across 2k–32k input points. Dir. and Undir. denote directed and undirected graphs, respectively. All metrics are percentages (higher is better), with the best results at each density in bold. The undirected graph performs best across all metrics at 2k points, while the directed graph achieves higher IoU and F1 scores at 8k–32k points. At 32k, both variants reach 98.0% accuracy, but the directed graph improves mIoU by 0.69 percentage points, indicating that undirected message passing provides no consistent advantage across sampling densities.

	2K		8K		16K		32K	
Metric	Dir.	Undir.	Dir.	Undir.	Dir.	Undir.	Dir.	Undir.
Accuracy	96.7	**97.0**	**97.4**	97.3	97.8	**97.9**	**98.0**	**98.0**
Visible IoU	89.57	**90.56**	**91.49**	91.45	**92.94**	92.64	**93.73**	92.78
Occluded IoU	95.44	**95.86**	**96.27**	96.21	**96.88**	96.74	**97.22**	96.79
mIoU	92.50	**93.21**	**93.88**	93.83	**94.91**	94.69	**95.48**	94.79
F1 (Visible)	94.50	**95.05**	**95.56**	95.53	**96.34**	96.18	**96.76**	96.25
Macro-F1	96.08	**96.47**	**96.83**	96.80	**97.38**	97.26	**97.68**	97.31
Balanced Accuracy	95.56	**96.04**	96.51	**96.67**	**97.22**	97.13	**97.61**	97.23

Table 6: Visibility estimation through surface reconstruction. This pipeline first reconstructs a surface using geometric Kazhdan & Hoppe (2013); Hou et al. (2022); Parakkat et al. (2024) or learning-based Yang et al. (2023); Zimo Wang & Li (2025) reconstruction methods, then estimates point visibility from the reconstructed surface. We report visibility accuracy (higher is better) and runtime in seconds (reconstruction + visibility inference) across 2k–32k inputs. Reconstruction dominates the computational cost. But it's unable to consistently generate accurate visibility estimates, particularly for sparse inputs. These results highlight the limitations of relying on an intermediate reconstructed surface for point visibility estimation. We also put the direct prediction with VISNET as a reference.

Method	2K	Time (s)	8K	Time (s)	16K	Time (s)	32K	Time (s)
Screened Poisson	0.75	$4.2 + 0.008$	0.82	$8.1 + 0.012$	0.84	$12.1 + 0.018$	0.86	$22.5 + 0.028$
Iterative Poisson	0.82	$43.5 + 0.070$	0.87	$64.3 + 0.010$	0.90	$83.0 + 0.017$	0.92	$117.4 + 0.030$
BallMerge	0.70	$0.13 + 0.005$	0.74	$0.52 + 0.009$	0.73	$1.11 + 0.016$	0.74	$2.32 + 0.030$
StEik	0.87	$390.7 + 0.390$	0.92	$392.1 + 0.420$	0.92	$397.4 + 0.440$	0.93	$394.8 + 0.440$
HotSpot	0.72	$266.9 + 0.057$	0.92	$270.2 + 0.094$	0.93	$281.8 + 0.099$	0.94	$282.6 + 0.104$
VisNet	0.97	0.04	0.98	0.06	0.98	0.08	0.98	0.09

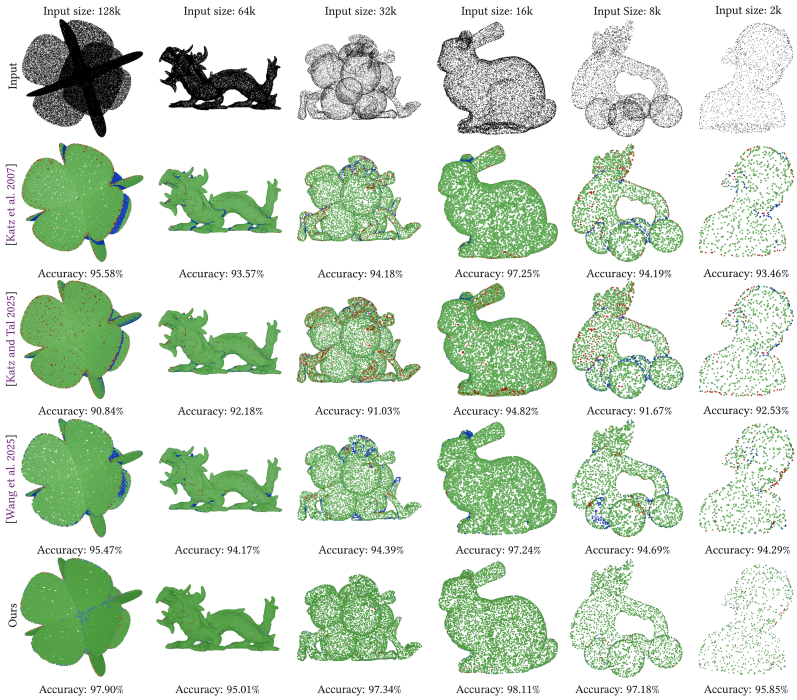

Figure 5: Comparison of point cloud visibility methods across varying densities on out-of-distribution data. We report accuracy (%) across a wide range of scales (2k–128k). Qualitative results (right) highlight true positives (green), false negatives (red), and false positives (blue), with point sizes increased for clarity.

Table 7: Accuracy comparison on the out-of-distribution ABC dataset across varying input densities. Higher values indicate better performance. VISNET achieves the highest accuracy at every density, outperforming NeuVis by 1.2–2.6 percentage points and demonstrating strong generalization to unseen object categories.

Method	**2k**	**8k**	**16k**	**32k**
HPR	90.4	93.2	93.8	92.7
HPRO	79.5	84.2	85.7	85.4
NeuVis	92.9	95.6	95.9	96.0
Ours	**95.5**	**96.8**	**97.4**	**97.7**

Table 8: Ablation study on the feature-channel dimension C. Increasing C from 32 to 64 improves accuracy by 0.9–1.9 percentage points across all input densities, at the cost of increasing the model size from 10.0 MB to 39.9 MB. We therefore use $C = 64$ in our final model.

Model Setting	**2k**	**8k**	**16k**	**32k**
Dim $C = 32$ (10.0 MB)	94.8	95.7	96.4	97.1
Dim $C = 64$ (39.9 MB)	**96.7**	**97.4**	**97.8**	**98.0**

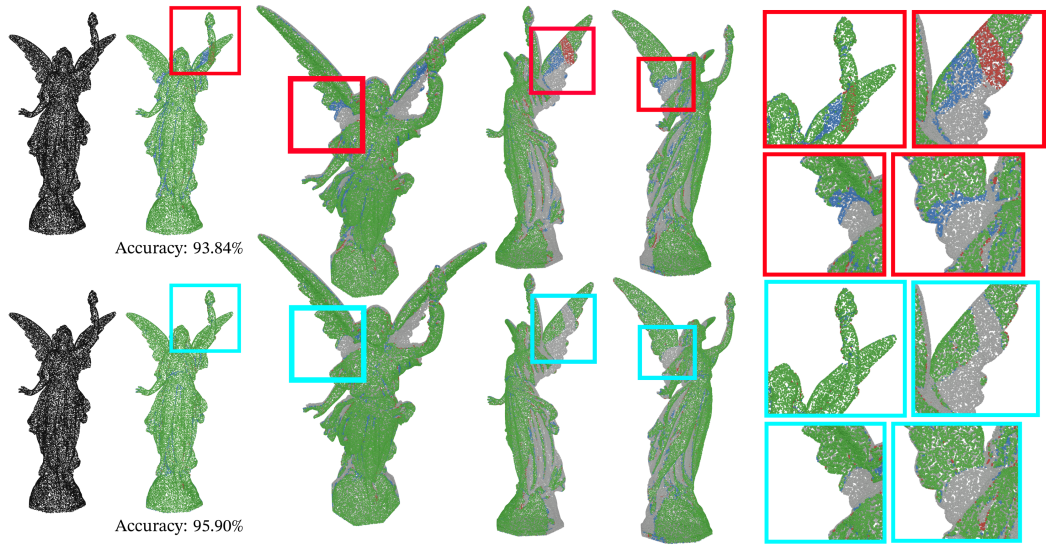

Figure 6: Out-of-distribution comparison on Lucy at 64k point cloud density. **Top:** NeuVis (93.84% accuracy). **Bottom:** VisNet (95.90%). The matching colored boxes (red for NeuVis, cyan for VisNet) localise the same regions on both rows, so the comparison reflects pointwise differences rather than viewpoint differences. VISNET resolves these line-of-sight ambiguities and recovers a near-clean front/back boundary, despite Lucy being far outside the ShapeNet training distribution.

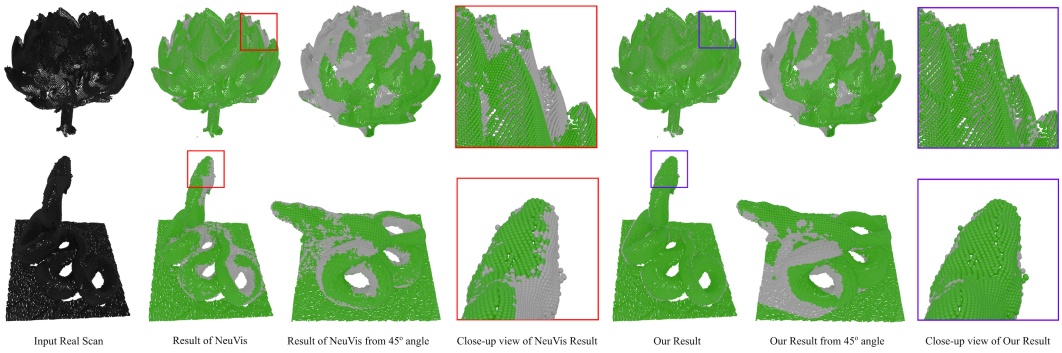

Figure 7: Qualitative comparison on real point clouds captured using an EinScan Pro HD scanner. From left to right, we show the input scan, NeuVis predictions, NeuVis predictions from a 45° alternative view and a corresponding close-up, followed by the same results for our VISNET. Compared with NeuVis, VISNET produces more spatially coherent visibility predictions around thin structures, and sharp depth transitions, even when the input surfaces are only partially captured and contain holes caused by incomplete scanning.

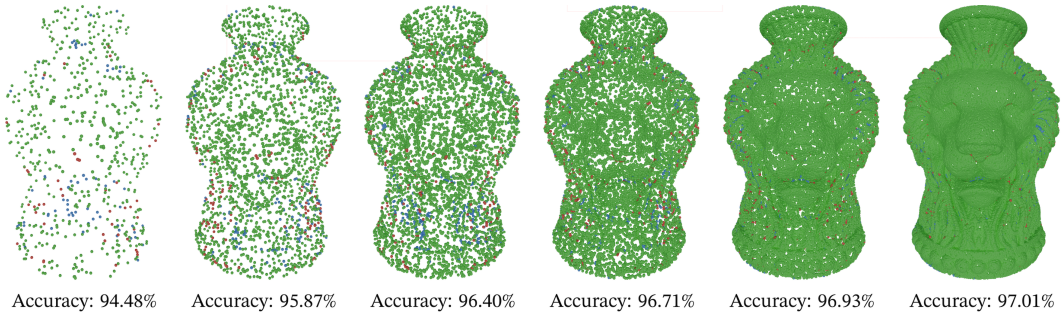

Figure 8: Performance comparison of point cloud visibility methods across varying densities on out-of-distribution data. Our method demonstrates robust scalability and superior accuracy from 2k up to 128k points.

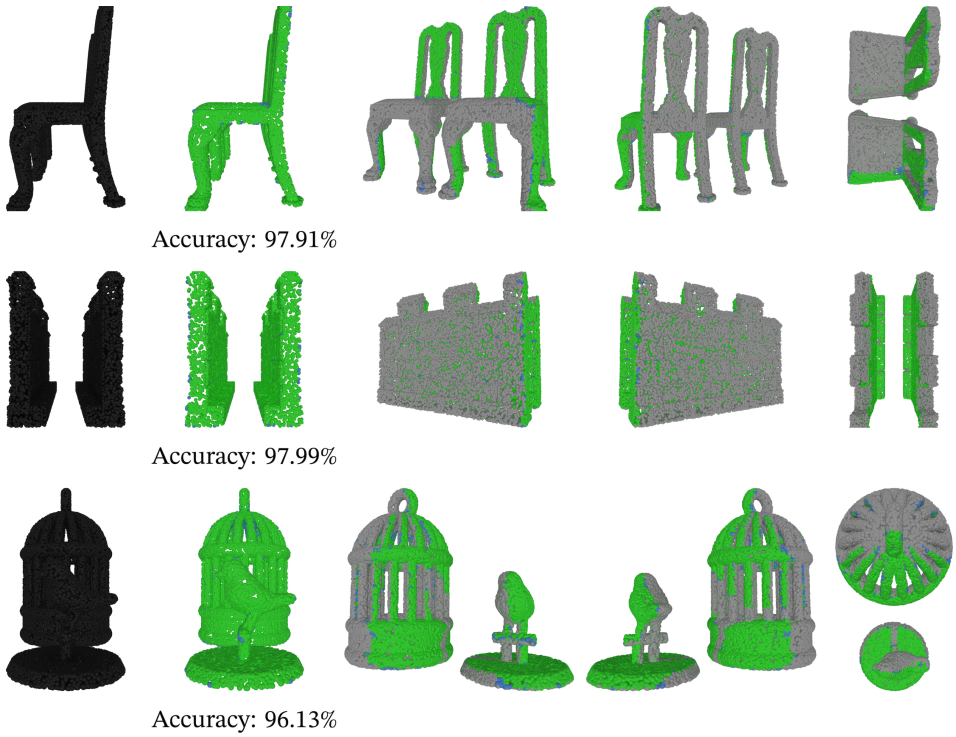

Figure 9: Reconstruction results for multi-object scenes from Thingi10K Zhou & Jacobson (2016). VISNET maintains consistent performance across varying numbers of objects in the scene, demonstrating that the number of components does not degrade the quality of visibility estimation.

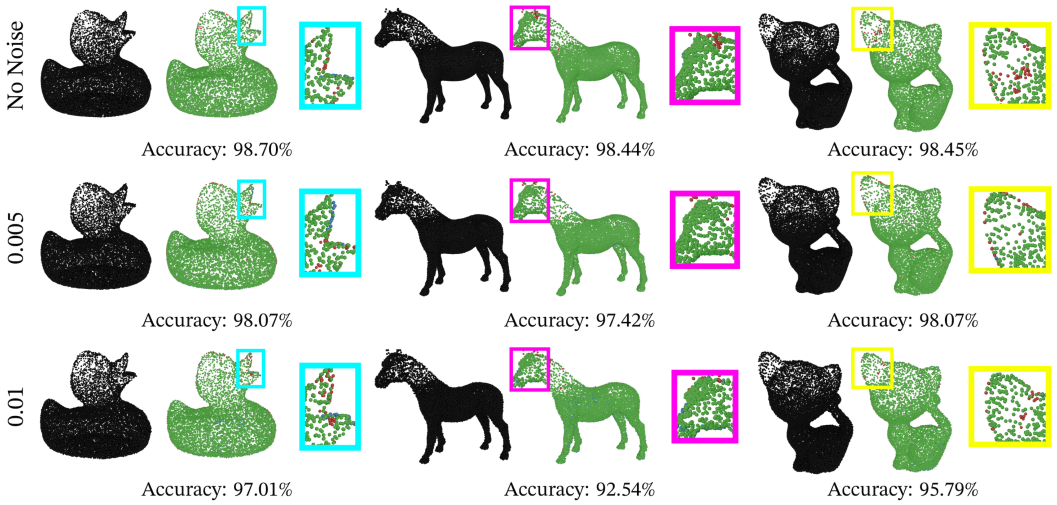

Figure 10: Results of VISNET on real-world scan simulated using Blensor Noise at 32K density. Accuracy degrades gracefully with increasing perturbation, the largest drop occurring on the horse, where the thin limbs amplify the effect of per-point displacement noise.



Figure 11: Surface reconstruction using visibility predictions from VISNET. For each object, input point counts increase from 1k to 2k, 8k, 16k, and 32k from left to right. Higher sampling densities enable finer geometric details and thin structures to be recovered.

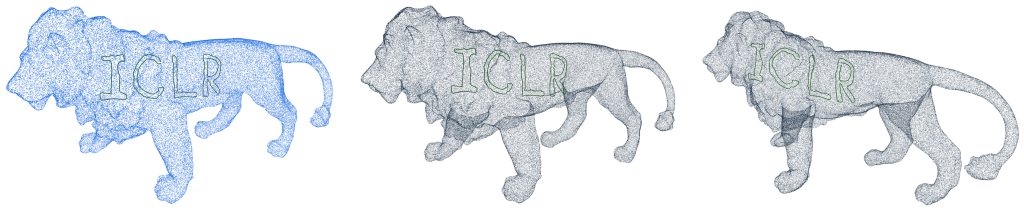

Figure 12: Sketch-based surface decoration: The user scribbles a 2D sketch on the view plane (left). These scribbles can be mapped to visible points to decorate point clouds (3D view of the sketch in the middle and right), ensuring the design follows the underlying 3D topology.

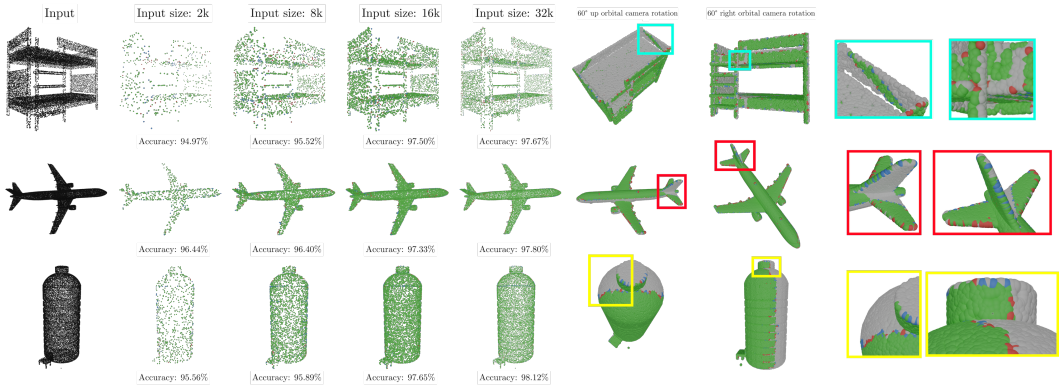

Figure 13: ShapeNet inference at 2k, 8k, 16k, and 32k. Per row (bed, airplane, bottle): the reference input point cloud (black), VisNet's prediction at each density (TP green, FP red, FN blue) with accuracy below, 60° *up* and *right* orbital views exposing back-side errors, and two zoomed insets on fine detail. Accuracy improves monotonically and exceeds 97% at 32k for all three objects.

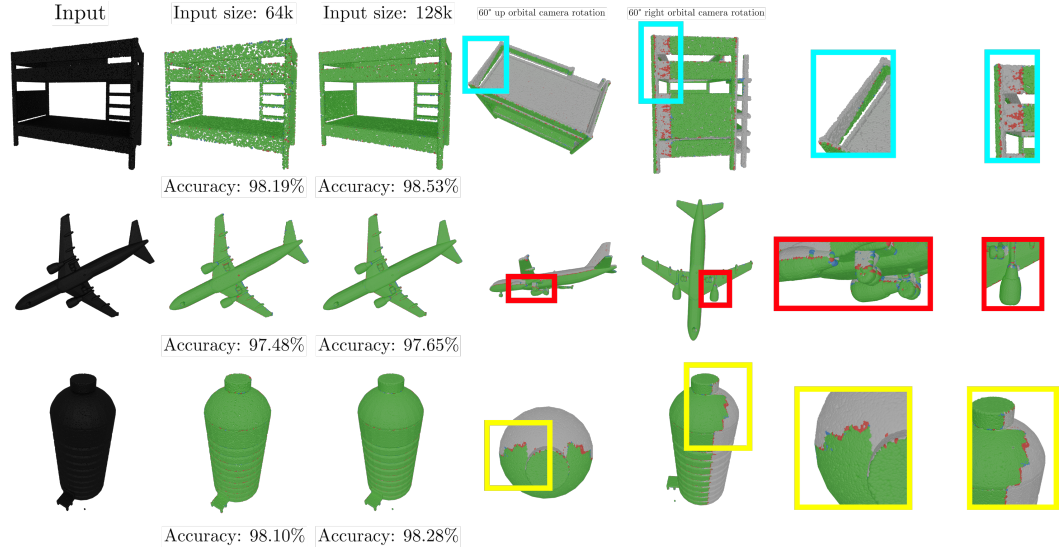

Figure 14: ShapeNet inference at 64k and 128k on the same objects as Figure 13, evaluated from a different viewpoint. Each row shows the input point cloud (black), predictions at 64k and 128k points (TP in green, FP in red, and FN in blue), and orbital views revealing errors on occluded surfaces. Prediction accuracy remains consistently high and improves with increased sampling density.