Hierarchical Point-Patch Fusion with Adaptive Patch Codebook for 3D Shape Anomaly Detection

Supplementary Material

This supplementary document expands on the full implementation pipeline, including pseudo-anomaly augmentation, construction of our industry dataset, multi-scale patchification strategies, codebook organization, and RoPE-enhanced cross-attention fusion. We also provide additional architectural explanations, ablation studies, and sensitivity analyses to offer deeper insight into how patch parameters, multi-scale representations, and modulation mechanisms contribute to robust 3D anomaly localization. We further include extensive qualitative comparisons, point-level quantitative results, anomaly-score behavior, and a detailed runtime breakdown to ensure full transparency and reproducibility of our approach.

1. Implementation Details

1.1. Pseudo Anomaly Augmentation

Although the figures below show mesh visualizations, all training and evaluation on our industry datasets use uniformly downsampled point clouds generated from the mesh faces. During training, to simulate anomaly types that closely resemble real defects, we augment the data with six pseudo-anomaly types in addition to the pseudo-anomalies already provided by the Real3D-AD and Anomaly-ShapeNet datasets. As illustrated in Fig. 1, each class shows the targeted anomaly pattern, and the label in brackets specifies the corresponding pseudo-anomaly method. These include inward normal shifts (sink), outward normal shifts (concavity), sine-wave displacements along normals (bulges), randomly positioned cropped holes, and cut-off regions created by randomly placed cubes or cylinders to emulate angular and planar displacements. For each pseudo-anomaly type, we generate three displacement levels (1e-3, 1e-2, 1e-1) to represent small, medium, and large severities. For every input shape, one of the six pseudo types and one of the three severity levels is sampled. This pseudo-anomaly augmentation enables the model to better discriminate between normal and anomalous patches and point features.

1.2. Industry Dataset Details

To construct the test samples for the industry dataset, which include angular or planar displacements and large part shifts as anomalies, we relied on publicly available 3D printing files of components from the robotic dexterous hand "OCRA" due to copyright restrictions on most industrial parts. We printed these components using a layerwise 3D printer and, through experiments, identified several common displacement-related anomalies. Based on these realistic errors, we manually mapped the corresponding anomaly regions onto the GT CAD meshes to generate meshes containing ground-truth anomalies. Although this process required manual effort, it enabled accurate annotation of real–world displacement defects. As illustrated in Fig. 2, the top part above the dashed line shows the anomaly components with red-highlighted defective regions. In contrast, the bottom part displays the corresponding GT CAD meshes. However, such mapping of anomaly regions can also be autonomous and scalable through 3D scan techniques using mobile devices.

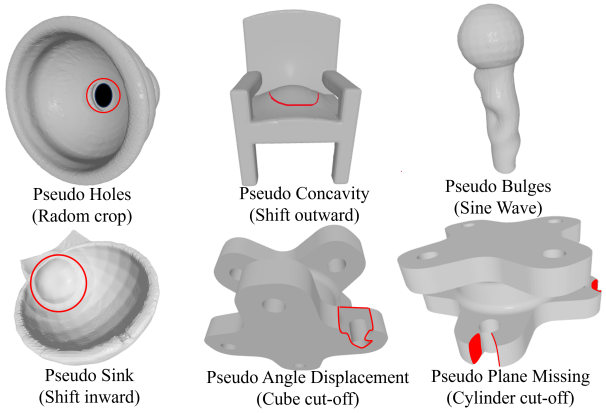

Figure 1. Pseudo anomalies generated through six negative-augmentation types during training. Each mesh shows the red-highlighted simulated anomaly, with the label indicating the anomaly type and its corresponding augmentation method in brackets.

1.3. Patchification Strategy

Fig. 3 presents patchification strategies: (a) input, (b) decomposition methods, (c) color-coded patches, (d) 1D layouts with deviation metrics. Regular grids (Row 1) cause 38% point variation due to geometry misalignment. FPS voxels (Row 2, 15%), FPS spheres (Row 3, 12%), and multi-scale FPS (Row 4, 10-15%) progressively improve distribution balance and spatial coherence. Multi-scale decomposition provides hierarchical features for robust anomaly detection across scales.

Furthermore, we provide semantic part–based patchification results in Fig. 4 using the self-supervised PartField [7]. Although semantic labels allow sampling points by

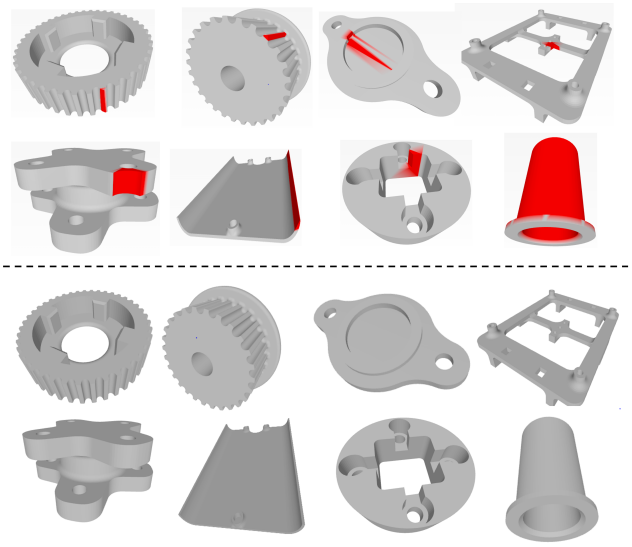

Figure 2. Our 3D anomaly dataset generated from real CAD models of open-source robotic hand components for 3D printing.

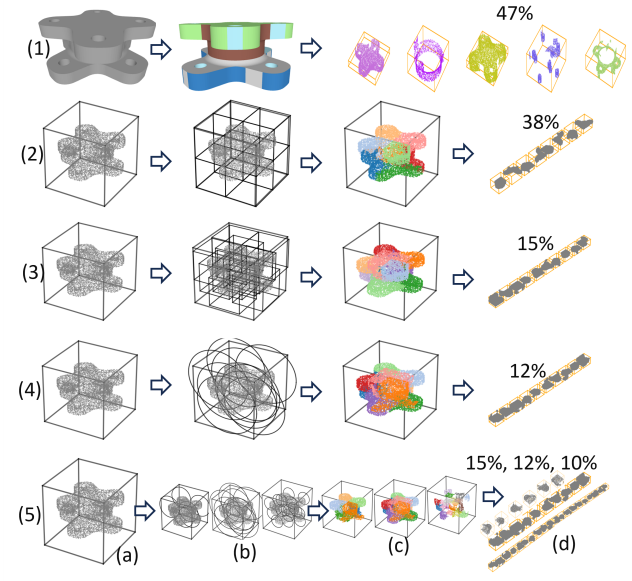

Figure 3. Visualization of different patchification strategies: (a) input point cloud input(b) input point cloud with varying patch sizes, where smaller patch sizes produce proportionally more patches; (c) patch-colored point cloud overlaid in the object's coordinate frame; and (d) a 1D layout of the patched point cloud, where the numbers above the point cloud patches indicate the average point-deviation proportion within each patch. Here we show the five different patchification strategies as mentioned in the main body accordingly.

part regions, the resulting patches are not robust. The part granularity is coarse, many inter-part boundaries are irregular, and some patches split a single semantic region or conflict with the underlying semantics. Only in the industrial dataset (last column) does the anomalous region appear consistently segmented, as in the blue plane. However, failures still occur, such as the misaligned middle gear in the last row (highlighted in red), where the semantic part fails to capture the geometric displacement. Overall, while semantic parts can segment certain large anomaly regions, their coarse resolution prevents reliable detection of small anomalies such as holes, cracks, and bulges. In our tests, semantic part guidance does not consistently improve point-level anomaly prediction, except for limited cases in the industrial dataset with minor metric gains.

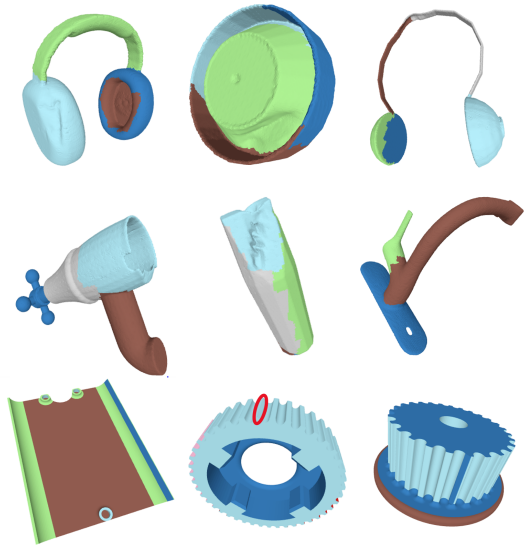

Figure 4. Visualization of semantic parts by using PartField [7].

1.4. Codebook

The diagram in Fig. 5 illustrates how multi-scale patch features are organized inside the codebook. For each of the three hierarchical scales, patches of different spatial sizes are extracted from the input shape and processed by the 3D U-Net to obtain 32-dimensional patch features. Each patch feature is stored together with its spatial key (x, y, z) in the codebook table. The t-SNE view (top right) shows how patch features from different scales populate the embedding space: '+' markers represent the patch-level features, while colored points correspond to the point-level features within each patch. When two patch features are highly similar—exceeding the predefined similarity threshold—they are merged via feature averaging and recorded as a single codebook entry, while still maintaining their individual spatial keys. This yields a compact yet spatially indexed codebook that preserves locality while reducing redundancy across scales.

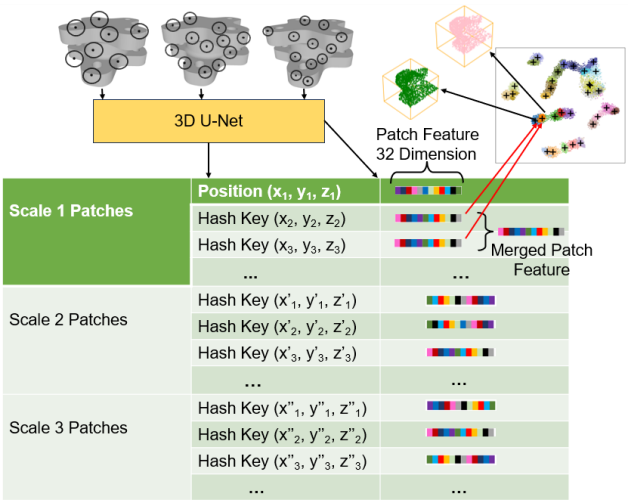

Figure 5. Visualization of the codebook structure. Patch features from the three hierarchical scales—each extracted by querying the 3D U-Net using different patch sizes—are embedded into 32-dimensional vectors and projected via t-SNE (top right). Many patch features overlap in the embedding space; '+' markers denote patch-level features, while the colored dots represent their corresponding point-level features. When two patch features exceed a similarity threshold (i.e., are nearly identical in Hilbert space), they are merged through feature averaging and stored as a single codebook entry, while remaining accessible through their distinct spatial keys (x, y, z).

1.5. RoPE-Enhanced Cross-Attention Fusion

The patch feature tokens $\{\mathbf{p}_j\}$ from adaptive patchification serve as *query* embeddings. In contrast, point feature tokens $\{\mathbf{z}_i\}$ from the pre-trained UNet encoder provide *key* and *value* embeddings in a multi-head cross-attention mechanism. To preserve spatial relationships, we employ Rotary Position Embedding (RoPE) [2], which injects positional information through rotation matrices $\mathbf{R}^d_{\Theta,m}$ applied to each token:

$$\text{RoPE}(\mathbf{z}_m) = \mathbf{R}^d_{\Theta,m}\,\mathbf{z}_m, \tag{1}$$

where $\mathbf{R}^d_{\Theta,m}$ encodes relative positions while preserving feature magnitudes. Following [2], the attention operation integrates RoPE as:

$$\hat{\mathbf{z}}_i = \frac{\sum_{n=1}^{N} \left(\mathbf{R}^d_{\Theta,n}\,\phi(\mathbf{q}_i)\right)^\top \left(\mathbf{R}^d_{\Theta,n}\,\varphi(\mathbf{k}_n)\right)\mathbf{v}_n}{\sum_{n=1}^{N} \phi(\mathbf{q}_i)^\top \varphi(\mathbf{k}_n)}, \tag{2}$$

where $\mathbf{q}_i = \mathbf{W}_Q\mathbf{p}_j$ denotes the query from patch j, $\mathbf{k}_n = \mathbf{W}_K\mathbf{z}_n$ and $\mathbf{v}_n = \mathbf{W}_V\mathbf{z}_n$ are key and value projections from point features, and $\phi(\cdot) = \varphi(\cdot) = \text{elu}(\cdot) + 1$ are non-negative feature maps. Multi-head concatenation and linear projection yield geometry-aware fused representations $\hat{\mathbf{z}}_i$ that encode both local point geometry and regional patch context for robust anomaly localization.

Table 1. RoPE vs vanilla cross-attention performance on Real3D-AD across multi-head configurations. RoPE consistently achieves 4-5% improvements in both metrics.

Method	Heads	Dim	AUC-ROC	AUC-PR
4-head configuration				
Vanilla	4	32	78.5	76.8
RoPE	4	32	**83.4**	**81.5**
Vanilla	4	64	79.1	77.4
RoPE	4	64	**84.2**	**82.7**
8-head configuration				
Vanilla	8	64	80.2	78.3
RoPE	8	64	**85.1**	**83.6**
Vanilla	8	128	80.5	78.7
RoPE	8	128	**85.4**	**84.0**

Tab. 1 compares RoPE-enhanced and vanilla cross-attention. RoPE consistently achieves 4-5% improvements across all configurations, demonstrating robust spatial encoding. Performance increases with more heads ($4 \rightarrow 8$: $+0.9\%$) but saturates at higher dimensions ($64 \rightarrow 128$: $+0.3\%$). The stable gain across settings confirms RoPE's effectiveness in capturing geometric relationships for anomaly detection.

1.6. Patch Feature Modulation

Figure 6 illustrates the patch score modulation and fusion mechanism. The architecture leverages patch-level discrepancies to guide point-level predictions through a dual-branch design. The gate branch ($\boldsymbol{\rho}_i$) determines which patches contribute to anomaly detection, while the modulation branch ($\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i$) adjusts feature magnitudes and biases based on patch similarity. RoPE-enhanced cross-attention enables spatial-aware fusion between point features and their nearest normal patch embeddings. The gated modulation then applies adaptive feature scaling: patches with larger discrepancies receive higher gate weights, amplifying their influence on the final prediction. This hierarchical design ensures that both local point geometry and regional patch context jointly inform the localization of anomalies. The residual connection in the final MLP preserves the attended features while allowing the network to predict refinement offsets, enabling stable training and accurate anomaly offset prediction in $\mathbb{R}^3$ space.

2. Ablation Study

Fig. 7 presents the 3D interaction landscape between patch size and patch number configurations. The surfaces exhibit two key characteristics: **(1) Ridge formation at patch size** 64, where performance consistently peaks across all patch

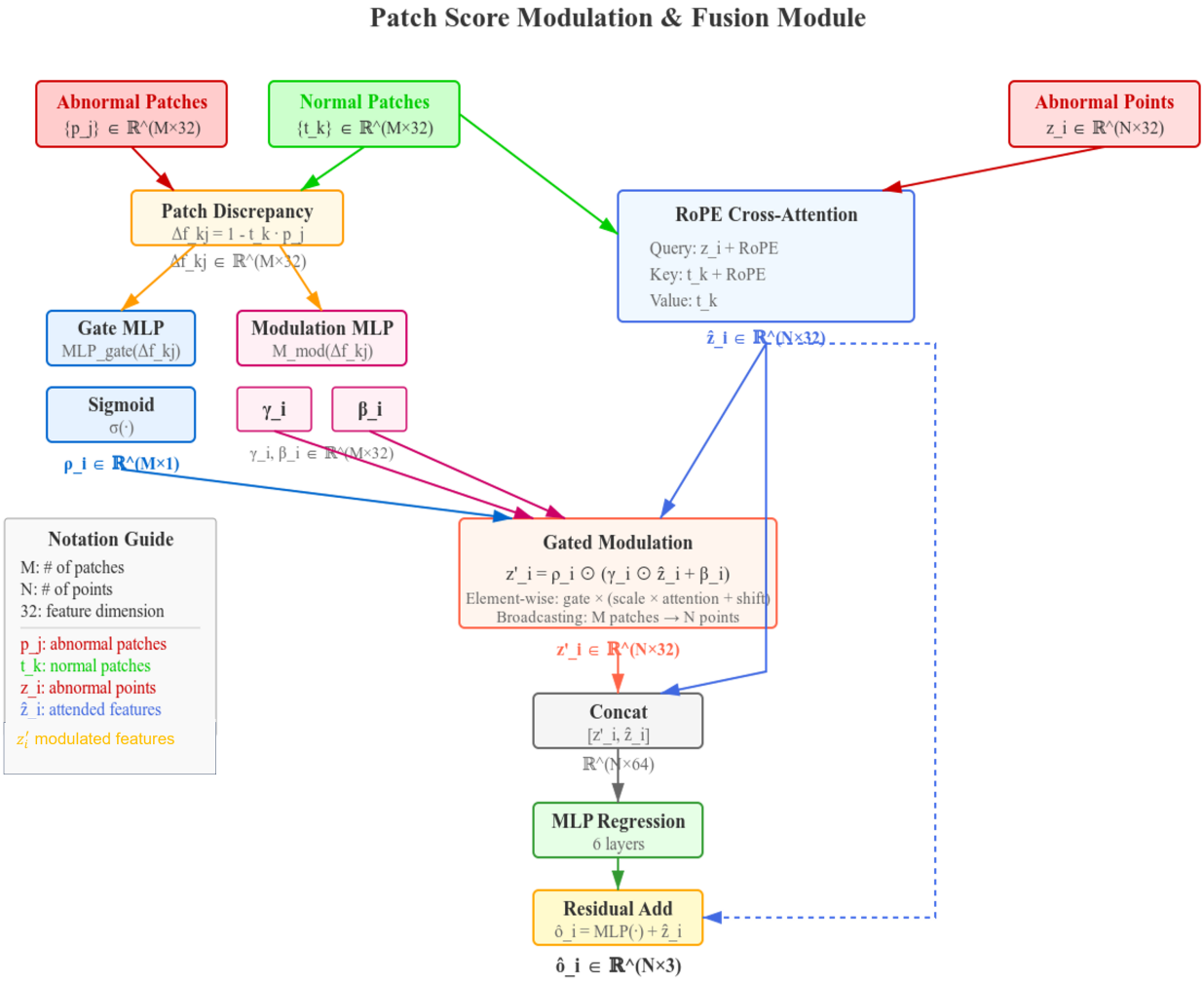

Figure 6. Patch Score Modulation and Fusion Module. From inputs $\{\mathbf{p}_j\}, \{\mathbf{t}_k\} \in \mathbb{R}^{M\times 32}$ (patches) and $\mathbf{z}_i \in \mathbb{R}^{N\times 32}$ (points), the pipeline: (1) computes patch discrepancy $\Delta\mathbf{f}_{kj}$, (2) generates gate weights $\boldsymbol{\rho}_i$ and modulation parameters $(\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i)$, (3) applies RoPE cross-attention $\hat{\mathbf{z}}_i$, (4) performs gated modulation $\mathbf{z}'_i = \boldsymbol{\rho}_i \odot (\boldsymbol{\gamma}_i \odot \hat{\mathbf{z}}_i + \boldsymbol{\beta}_i)$, (5) concatenates features, and (6) predicts offsets $\hat{\mathbf{o}}_i \in \mathbb{R}^{N\times 3}$ via MLP with residual connection. M: patches, N: points.

numbers, demonstrating that medium-sized patches optimally balance local context capture and spatial precision for anomaly localization. **(2) Saturation plateau along the patch number axis**, where performance gains diminish beyond 128 patches, indicating sufficient spatial granularity has been achieved. At smaller patch sizes (< 32), insufficient context limits discriminative power, while larger patches (> 128) over-smooth fine-grained anomaly boundaries, causing gentle performance degradation. The optimal configuration resides at $(64, 256)$ for object-level (84.0%) and $(64, 128)$ for point-level (86.0%), with point-level metrics consistently outperforming object-level by 5-8% due to finer localization granularity. This 3D analysis confirms that the proposed multi-scale framework achieves robust performance across diverse configurations while identifying the sweet spot for practical deployment. But the number of points in each segmented part also varies a lot, leading to a large patch feature difference.

Lastly, to compare how patch size affects different anomaly types and severities, we report the AUC-ROC (%) results in Fig. 8 at a fixed 256-voxel resolution. For the two anomaly cases—(a) holes on the chair and (b) a missing corner on the Franka Connector—the best performance is achieved at patch size 8 (90.4%) and patch size 32 (75.6%), respectively. These results highlight that selecting an appropriate patch size is crucial for achieving optimal performance across varying anomaly scales.

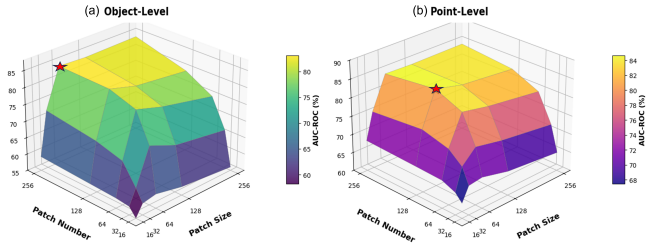

Figure 7. 3D sensitivity analysis of patch size and patch number interactions on Real3D-AD. Left: (a) Object-level AUC-ROC (%). Right: (b) Point-level AUC-ROC (%). The surfaces reveal optimal configurations at patch size 64 across varying patch numbers. Red stars mark peak performance regions.

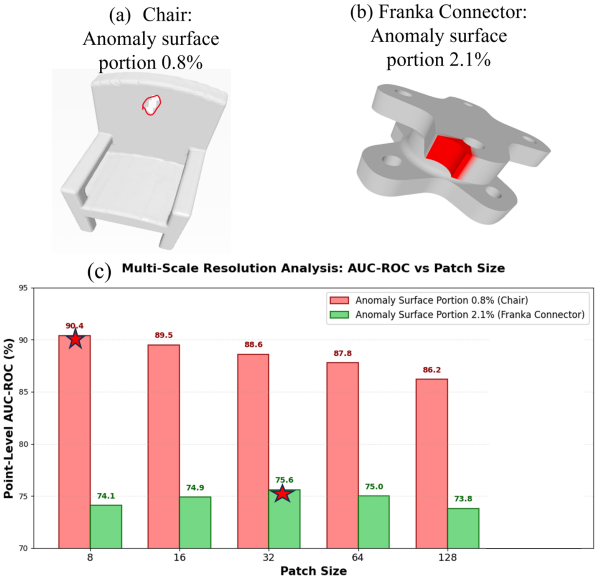

Figure 8. AUC-ROC performance under varying patch sizes and anomaly surface portions. (a) and (b) show the Shari and Franka Connector shapes with anomalies. (c) presents the AUC-ROC curves for both classes—chair in pink bars and connector in green bars—with stars marking the peak performance.

2.1. Multiscale Patch Features

As shown in the t-SNE feature distributions across different patch numbers in Fig. 9, increasing the number of patches reduces the patch size, yielding finer-grained embeddings. For visualization and runtime efficiency, we downsample the input point features to 8,192 points. The duck scan is a half-view scan, whereas the vase is a full 360° scan with strong symmetry, resulting in more overlapping patch clusters due to identical or highly similar features. As the number of patches increases, each cluster becomes smaller, but the overall class-wise distribution remains largely consistent, with patch-level feature clusters becoming increasingly sparse.

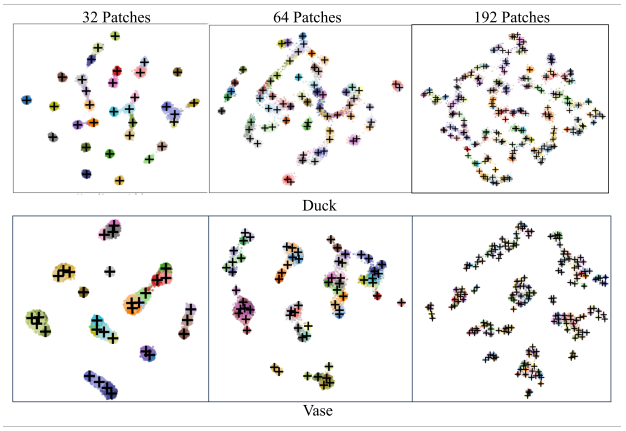

Figure 9. Patch features t-SNE projected onto 2D space with various patch numbers, across $32, 64, 192$ patches for the class of Duck and Vase, while the corresponding path size of each scale level is $64, 32, 8$ respectively.

2.2. Anomaly Score

During inference, each point receives an anomaly score $\hat{\delta}_i \in [0, 1]$. Empirically, scores above 0.82 indicate highly reliable anomaly predictions, as shown in Fig. 10. To obtain stable and noise-free outputs, we further keep only the top 20 points with the highest anomaly scores as the final anomaly detections. This strategy suppresses scattered false positives and consistently highlights the most confident defect regions.

2.3. Time Complexity

The FPS patchification module adopts farthest-point sampling with local neighborhood aggregation to construct stable and spatially uniform patches from the input point cloud. Our FPS sampling is implemented using an open-source fast FPS library based on the FPS-NPDU strategy. To analyze its computational behavior, we separately vary the number of sampled points N and the neighborhood size k, while keeping the other variable fixed. Row 1 of Sec. 2.3 reports the scaling with respect to N under a fixed window size of $k = 32$, showing a near-linear increase consistent with the theoretical complexity $\mathcal{O}(Nk)$. Row 2 reports the scaling with respect to k using a fixed point cloud size of $N = 1024$, demonstrating similarly linear growth.

As the patch feature codebook is pre-computed once during training, inference only requires running FPS patchification on the anomaly input shape. This design removes any additional codebook lookup overhead, and the resulting sampling cost remains lightweight, averaging only 20–50 ms across test point clouds of varying input resolutions. Overall, the module provides an efficient and scalable

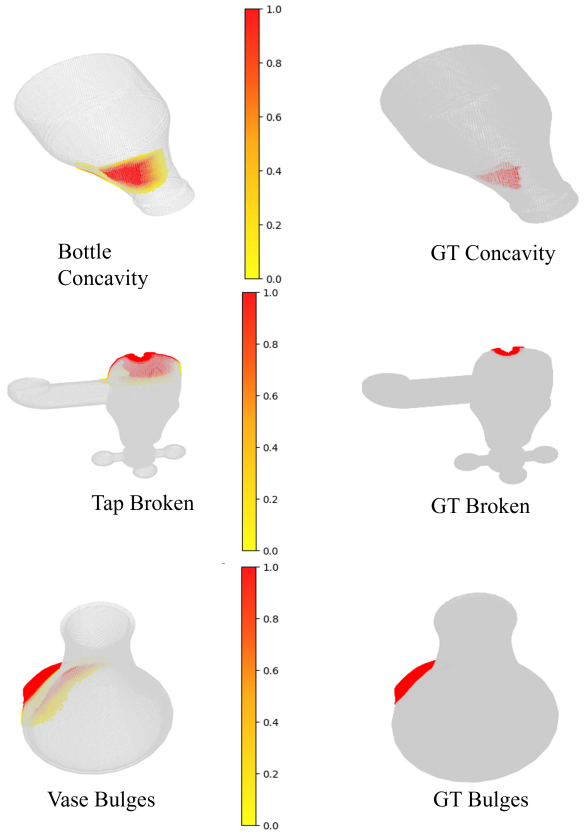

Figure 10. Visualization of predicted anomaly regions and ground-truth defect areas across three representative examples. The left column shows the model's anomaly outputs, where the colormap (yellow → red) indicates the predicted uncertainty level, with yellow representing lower anomaly scores and red indicating higher confidence. Only points with predicted scores above 1×10^{-1} are visualized. The right column displays the corresponding ground-truth (GT) defect areas, where points labeled as anomalies (label $= 1$) are highlighted in red while all remaining points are shown in gray.

patch construction strategy that preserves geometric coverage while maintaining practical runtime efficiency during inference.

Table 2. Time complexity behavior of FPS patchification with respect to point count N and neighborhood size k. Row 1 uses a fixed $k = 32$. Row 2 uses fixed $N = 1024$

Scaling in N	1024	2048	4096	8192	16384
Time (ms)	3.1	6.2	12.6	25.4	51.0
Scaling in k	32	64	192	256	–
Time (ms)	2.8	5.7	17.4	23.2	–

In Tab. 3, our method achieves competitive inference speed (178ms) compared to state-of-the-art approaches, with only 3.5% overhead relative to the fastest method (PO3AD) while maintaining superior detection performance. All methods were tested on an NVIDIA RTX 3090 GPU with identical hardware settings and a batch size of 1 for fair comparison.

Table 3. Inference speed comparison tested on Real3D-AD [6].

Method	**Avg Inference Speed (ms)**
R3DAD [10]	183
PO3AD [9]	172
Ours	178

As shown in Tab. 4, the 3D UNet backbone dominates computation (51.7%), followed by the RoPE cross-attention decoder (29.4%). FPS patchification accounts for 14.4% of inference time, while the lightweight MLP head contributes minimal overhead (4.5%). All timings measured on a single NVIDIA RTX 3090 GPU with batch size 1.

Table 4. Inference time breakdown of model components. Total average inference time is 178ms per sample.

Module	**Time (ms)**	**Percentage (%)**
3D UNet (MinkUNet34C)	92.1	51.7
RoPE Cross-Attention Decoder	52.3	29.4
MLP-based Head	8.0	4.5
FPS Patchification	25.7	14.4
Total	**178.1**	**100.0**

Loss combination. As shown in Tab. 5, $\mathcal{L}_{\text{dist}}$ contributes most significantly when used alone, while removing either $\mathcal{L}_{\text{sim}}$ or $\mathcal{L}_{\text{BCE}}$ leads to a slight performance drop. Combining all three losses yields the highest overall AUC-ROC of 84.2%, confirming their complementary effects.

Table 5. Ablation study on the effect of individual and combined losses (6 types) by Object-level AUC-ROC (%).

Loss Combination	1	2	3	4	5	6
$\mathcal{L}_{\text{dist}}$	✓	✗	✗	✓	✓	✓
$\mathcal{L}_{\text{sim}}$	✗	✓	✗	✓	✗	✓
$\mathcal{L}_{\text{BCE}}$	✗	✗	✓	✗	✓	✓
AUC-ROC (%)	61.8	54.2	58.5	75.9	78.4	**84.2**

3. Baseline Results

For Real3D-AD, the latest PO3AD [9] baseline does not provide point-level metrics, and its released code does not support reliable point-level evaluation. Therefore, we follow ISMP [5] and report their published point-level results

Table 6. Comparison of point-level AUC-ROC results on the Anomaly-ShapeNet dataset.

Method	ashtray0	bag0	bottle0	bottle1	bottle3	bowl0	bowl1	bowl2	bowl3	bowl4	bowl5	bucket0	bucket1	cap0
M3DM [8]	57.7	63.7	66.3	63.7	53.2	65.8	66.3	69.4	65.7	62.4	48.9	69.8	69.9	53.1
CPMF [1]	61.5	65.5	52.1	57.1	43.5	74.5	48.8	63.5	64.1	68.3	68.4	48.6	60.1	60.1
Reg3D-AD [6]	69.8	71.5	88.6	69.6	52.5	77.5	61.5	59.3	65.4	80.0	69.1	61.9	75.2	63.2
IMRNet [3]	67.1	66.8	55.6	70.2	64.1	78.1	70.5	68.4	59.9	57.6	71.5	58.5	77.4	71.5
ISMP [5]	86.5	73.4	72.2	86.9	74.0	76.2	70.2	70.6	85.1	75.3	73.3	54.5	68.3	67.2
PO3AD [9]	96.2	94.9	91.2	84.4	88.0	**97.8**	91.4	91.8	93.5	**96.7**	**94.1**	75.5	89.9	95.7
Ours	**98.5**	**95.2**	**91.6**	**86.3**	**89.4**	96.0	**92.7**	**92.3**	**94.2**	95.0	92.6	**86.0**	**90.5**	**96.1**
Method	cap3	cap4	cap5	cup0	cup1	eraser0	headset0	headset1	helmet0	helmet1	helmet2	helmet3	jar0	micro.
M3DM [8]	60.5	71.8	65.5	71.5	55.6	71.0	58.1	58.5	59.9	42.7	62.3	65.5	54.1	35.8
CPMF [1]	55.1	55.3	55.1	49.7	50.9	68.9	69.9	45.8	55.5	54.2	51.5	52.0	61.1	54.5
Reg3D-AD [6]	71.8	81.5	46.7	68.5	69.8	75.5	58.0	62.6	60.0	62.4	82.5	62.0	59.9	59.9
IMRNet [3]	70.6	75.3	74.2	64.3	68.8	54.8	70.5	47.6	59.8	60.4	64.4	66.3	76.5	74.2
ISMP [5]	77.5	66.1	77.0	55.2	85.1	52.4	47.2	84.3	61.5	60.3	56.8	52.2	66.1	60.0
PO3AD [9]	**94.8**	94.0	86.4	90.9	93.2	**97.4**	82.3	90.7	87.8	94.8	93.2	84.6	87.1	81.0
Ours	92.1	**94.6**	**87.0**	**91.5**	**94.3**	95.0	**85.1**	**91.3**	**88.2**	**95.0**	**93.7**	**87.3**	**88.6**	**83.7**
Method	shelf0	tap0	tap1	vase0	vase1	vase2	vase3	vase4	vase5	vase7	vase8	vase9	Average	
M3DM [8]	55.4	65.4	71.2	60.8	60.2	73.7	65.8	65.5	64.2	51.7	55.1	66.3	61.6	
CPMF [1]	78.3	45.8	65.7	45.8	48.6	58.2	58.2	51.4	65.1	50.4	52.9	54.5	57.3	
Reg3D-AD [6]	68.8	58.9	74.1	54.8	60.2	40.5	51.1	75.5	62.4	88.1	81.1	69.4	66.8	
IMRNet [3]	60.5	68.1	69.9	53.5	68.5	61.4	40.1	52.4	68.2	59.3	63.5	69.1	65.0	
ISMP [5]	70.1	84.4	67.8	68.7	53.4	77.3	62.2	54.6	58.0	74.7	73.6	82.3	69.1	
PO3AD [9]	66.3	78.3	69.2	95.5	88.2	**97.8**	88.4	90.2	93.7	**98.2**	95.0	95.2	89.8	
Ours	**75.2**	**85.4**	**78.5**	**95.8**	**89.1**	95.5	**89.7**	**91.3**	**94.6**	96.0	**95.4**	**95.9**	**91.2**	

Table 7. Point-level AUCROC(↑) on Real3DAD (zoom-in).

Method	**Airplane**	**Car**	**Candy**	**Chicken**	**Diamond**	**Duck**	**Fish**	**Stone**	**Seahorse**	**Shell**	**Starfish**	**Toffees**	**Avg**
BTF(Raw)	0.564	0.647	0.735	0.609	0.563	0.601	0.514	0.597	0.520	0.489	0.392	0.623	0.571
BTF(FPFH)	0.738	0.708	0.864	0.735	0.882	0.875	0.709	**0.891**	0.512	0.571	0.501	0.815	0.733
M3DM	0.547	0.602	0.679	0.678	0.608	0.667	0.606	0.674	0.560	0.738	0.532	0.682	0.631
PatchCore(F)	0.562	0.754	0.780	0.429	0.828	0.264	0.829	**0.910**	0.739	0.739	0.606	0.747	0.682
PatchCore(M)	0.569	0.609	0.627	0.729	0.718	0.528	0.717	0.444	0.633	0.709	0.580	0.580	0.620
RegAD	0.631	0.718	0.724	0.676	0.835	0.503	0.826	0.545	**0.817**	0.811	0.617	0.759	0.705
ISMP	**0.753**	**0.836**	**0.907**	**0.798**	**0.926**	**0.876**	**0.886**	0.857	0.813	**0.839**	**0.641**	**0.895**	**0.836**
Ours	**0.904**	**0.789**	**0.912**	**0.848**	**0.915**	**0.903**	**0.910**	0.869	**0.856**	**0.851**	**0.823**	**0.902**	**0.874**

on Real3D-AD for comparison with our method in Tab. 7.

In addition to the qualitative results in the main paper, we provide extended visualizations covering other major anomaly types in Fig. 11. All remaining test samples from our industry dataset are also included in Fig. 11 for completeness. Furthermore, we report the point-level AUC-ROC results on Anomaly-ShapeNet [3] in Tab. 6, complementing the object-level metrics presented in the main text. For ISMP [4], the released code contains issues that prevent successful training and testing. Therefore, we present their object-level results in the main paper and point-level results in the supplementary material, though these numbers may not be fully reliable.

References

[1] Yunkang Cao, Xiaohao Xu, and Weiming Shen. Complementary pseudo multimodal feature for point cloud anomaly detection. *Pattern Recognition*, 156:110761, 2024. 7

[2] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020. 3

[3] Wenqiao Li and etal. Towards scalable 3d anomaly detection and localization: A benchmark via 3d anomaly synthesis and a self-supervised learning network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22207–22216, 2024. 7

[4] Hanzhe Liang and etal. Look inside for more: Internal spatial modality perception for 3d anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5146–5154, 2025. 7

[5] Hanzhe Liang, Guoyang Xie, Chengbin Hou, Bingshu Wang, Can Gao, and Jinbao Wang. Look inside for more: Internal spatial modality perception for 3d anomaly detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(5):5146–5154, 2025. 6, 7

[6] Jiaqi Liu, Guoyang Xie, Ruitao Chen, Xinpeng Li, Jinbao Wang, Yong Liu, Chengjie Wang, and Feng Zheng. Real3d-ad: A dataset of point cloud anomaly detection. *Advances in Neural Information Processing Systems*, 36:30402–30415, 2023. 6, 7

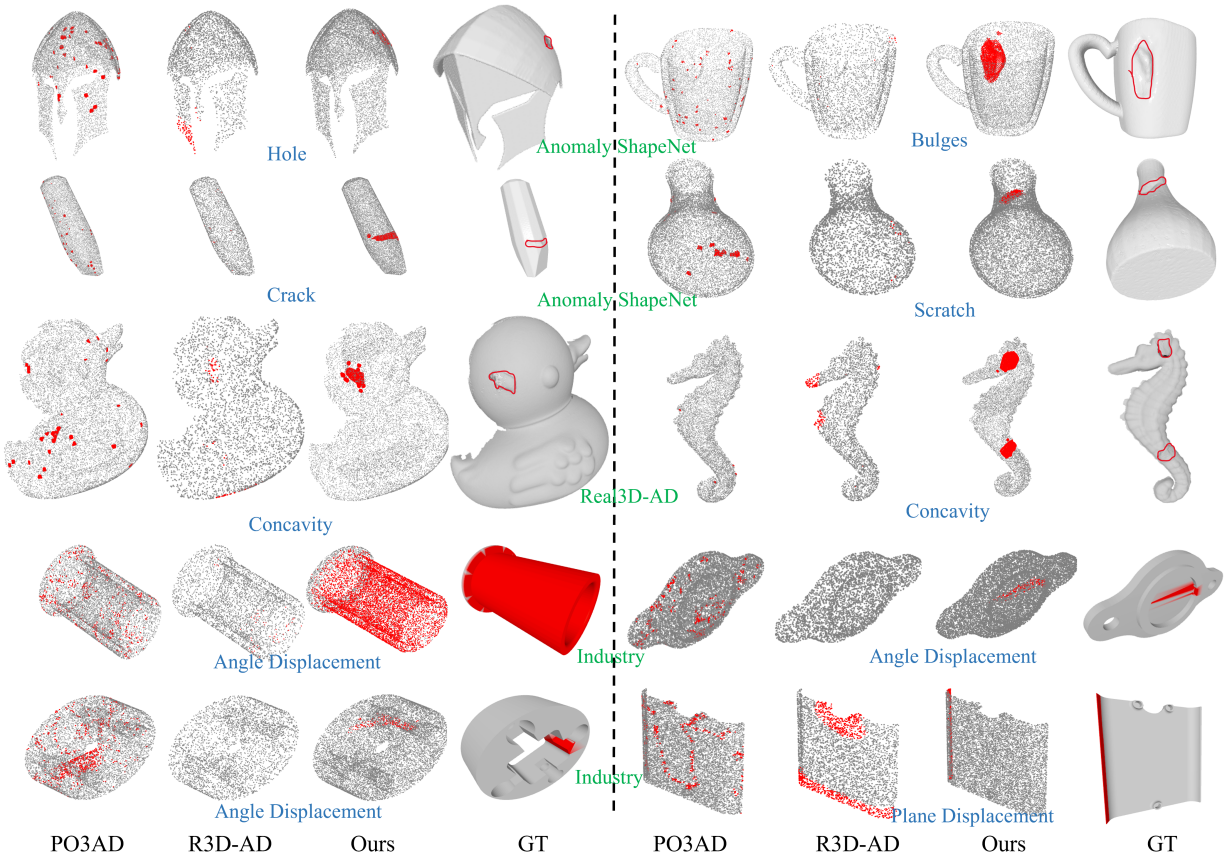

Figure 11. Qualitative comparison on three datasets (green letters): Anomaly-ShapeNet (1st–2nd rows), Real3D-AD (3rd), and our industrial components (4th-5th rows). The dashed line indicates different object classes in each row. Anomaly types (in blue) include displacement, crack, concavity, holes, and bulges. Anomaly points are highlighted in red. Ground truth uses red masks (last two rows) or red contours (first three rows) to indicate anomaly regions overlaid with an anomalous mesh. Each column is the result of a method.

[7] Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9704–9715, 2025. 1, 2

[8] Chengjie Wang, Haokun Zhu, Jinlong Peng, Yue Wang, Ran Yi, Yunsheng Wu, Lizhuang Ma, and Jiangning Zhang. M3dm-nr: Rgb-3d noisy-resistant industrial anomaly detection via multimodal denoising. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 7

[9] Jianan Ye and etal. Po3ad: Predicting point offsets toward better 3d point cloud anomaly detection. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1353–1362, 2025. 6, 7

[10] Zheyuan Zhou, Le Wang, Naiyu Fang, Zili Wang, Lemiao Qiu, and Shuyou Zhang. R3d-ad: Reconstruction via diffusion for 3d anomaly detection. In *European conference on computer vision*, pages 91–107. Springer, 2024. 6