Efficient Industrial Point Cloud Anomaly Detection via Mamba and Selective Anomalous Feature Generation

Dinh-Cuong Hoang[a,*], Phan Xuan Tan[b,*], Anh-Nhat Nguyen[c], Hoang-Nam Duong[a], Minh-Duc Cao[a], Duc-Huy Ngo[a], Ta Huu Anh Duong[a], Tuan-Minh Huynh[a], Duc-Manh Nguyen[a], Van-Thiep Nguyen[c], Thu-Uyen Nguyen[c], Minh-Quang Do[c], Xuan-Tung Dinh[c], Van-Hiep Duong[c] and Ngoc-Anh Hoang[c]

[a]*Greenwich Vietnam, FPT University, Hanoi, 10000, Vietnam*
[b]*College of Engineering, Shibaura Institute of Technology, Tokyo, 135-8548, Japan*
[c]*ICT Department, FPT University, Hanoi, 10000, Vietnam*

ARTICLE INFO

Keywords:
Industrial anomaly detection
Defect recognition
Object Anomaly Segmentation

ABSTRACT

Automated detection of surface defects on three-dimensional (3D) parts is vital for ensuring product quality and safety in manufacturing. However, three key challenges hinder reliable detection: geometric context ambiguity across complex part shapes, domain mismatch between generic pretrained features and industrial scans (with their unique noise and reflectivity), and the scarcity of diverse defect examples for training. To overcome these issues, we propose a novel single-forward-pass framework for point cloud anomaly detection, comprising three new modules: (1) Spatial Context Aggregation, which grounds each local patch in a set of learned global prototypes via an optimal-transport alignment to resolve context ambiguity; (2) Feature Adaptor, a lightweight two-layer multilayer perceptron (MLP) that fine-tunes self-supervised Point-MAE embeddings to the specific characteristics of industrial scans; and (3) Selective Anomalous Feature Generator, which synthesizes realistic hard negatives by corrupting random subsets of feature tokens, thus mitigating the need for extensive defect labels. An attention-based discriminator trained with patch-wise supervision learns to distinguish these hard negatives from genuine defect-free patterns. At inference, our pipeline delivers dense per-point anomaly scores in a single pass at up to 13.5 frames per second (FPS). On the Real3D-AD benchmark, we observe point-level improvements of 2.8% in area under the receiver operating characteristic curve (AUROC) and 5.7% in area under the precision-recall curve (AUPR), with object-level gains of 3.0% (AUROC) and 3.5% (AUPR). Evaluated on our newly released Industrial3D-AD dataset, which captures realistic sensor noise and reflective materials, we see similar enhancements (2.9%/5.3% point-level, 2.8%/3.3% object-level).

1. Related Work

1.1. Industrial Image Anomaly Detection

Industrial image anomaly detection (IAD) aims to identify and localize visual defects in manufacturing and inspection images. Prior research in this field can be broadly categorized according to the level of supervision available during training, namely supervised, semi-supervised, and unsupervised methods. Each category reflects a different trade-off between annotation cost, generalization ability, and detection performance.

Supervised methods formulate defect detection as a conventional classification or segmentation problem that relies on explicit anomaly annotations. Early approaches employ convolutional neural networks (CNNs) to learn discriminative representations of defective versus normal samples. Attention-based CNNs are designed to emphasize regions most relevant to defects [1], while transformer architectures leverage their ability to capture long-range contextual dependencies for irregularity detection [2]. When sufficient labeled anomalies are available, supervised methods can achieve very high accuracy. However, collecting comprehensive labeled datasets that cover the wide

*Corresponding author
cuonghd12@fe.edu.vn (D. Hoang); tanpx@shibaura-it.ac.jp (P.X. Tan); nhatna3@fe.edu.vn (A. Nguyen); Dn6175j@gre.ac.uk (H. Duong); duccmgch230132@fpt.edu.vn (M. Cao); huyndgch230154@fpt.edu.vn (D. Ngo); duongthagch220838@fpt.edu.vn (T.H.A. Duong); minhhtgch230186@fpt.edu.vn (T. Huynh); manhndgch220466@fpt.edu.vn (D. Nguyen); thiepnvhe173027@fpt.edu.vn (V. Nguyen); uyennthe176614@fpt.edu.vn (T. Nguyen); quangdmhe180854@fpt.edu.vn (M. Do); tungdxhe172611@fpt.edu.vn (X. Dinh); hiepdvhe181185@fpt.edu.vn (V. Duong); anhhnhe186401@fpt.edu.vn (N. Hoang)
ORCID(s): 0000-0001-6058-2426 (D. Hoang)

diversity of defect types and appearance variations is often impractical in real industrial scenarios. As a result, these methods are limited by their dependence on exhaustive annotations.

Semi-supervised approaches attempt to bridge the gap between supervised and unsupervised paradigms by combining a small number of labeled anomalies with a large corpus of defect-free images. Common strategies include pseudo-labeling, class-imbalance reweighting, and hybrid pipelines that integrate unsupervised representation learning with supervised fine-tuning [3, 4]. For example, neural network adaptations can learn from sparse anomaly labels while preserving representations of normal data distributions. These methods often outperform purely unsupervised approaches when reliable anomaly labels exist. Nevertheless, their performance still heavily depends on the diversity and quality of the labeled anomalies, and it tends to degrade when anomalies exhibit substantial variation or unseen patterns.

Unsupervised IAD methods eliminate the need for anomaly annotations entirely. They are trained exclusively on defect-free images and are designed to detect anomalies by identifying deviations from learned normal patterns during inference. Two main paradigms dominate this category. The first focuses on reconstruction-based learning, where models such as autoencoders, generative adversarial networks (GANs), diffusion models, and vision transformers are trained to reconstruct normal samples under the assumption that defects will result in higher reconstruction errors. Representative examples include multi-scale feature-guided autoencoders [5], divide-and-assemble memory-based models [6], joint reconstruction–discrimination embeddings [7], and dual subspace re-projection (DSR) [8] that simulates near-in-distribution defects. Recently, diffusion and transformer-based inpainting frameworks such as masked transformers [9], dual attention architectures [10], adaptive inpainting networks (AMI-Net) [11], and dynamic diffusion systems [12, 13, 14] have further improved reconstruction fidelity and localization precision. Although these approaches are conceptually simple and widely applicable, they require careful architectural design and regularization to prevent over-smoothing or the unintended reconstruction of defects.

The second major family of unsupervised methods operates in feature space rather than pixel space. These methods rely on pretrained visual encoders to extract semantic embeddings of normal images and then measure deviations from these embeddings during testing. Teacher–student distillation frameworks train a student network to imitate the feature representations of a fixed teacher model, where large deviations between the two networks indicate potential anomalies [15, 16, 17, 18, 19]. Memory-based approaches maintain a feature bank of representative normal descriptors and identify anomalies using nearest-neighbor search or probabilistic distance metrics [20, 21, 22, 23, 24]. These feature-based methods benefit from the semantic richness of large pretrained models but are sensitive to domain shift, since industrial imagery often differs substantially from natural images such as those in ImageNet. As a consequence, mismatched feature distributions can limit anomaly sensitivity in industrial applications.

A complementary research direction focuses on synthesizing pseudo-anomalies to enable discriminative training without requiring true defect labels. Pixel-level anomaly generation methods such as CutPaste [25] and discriminative synthetic augmentation frameworks [7] create artificial anomalies on clean images and train a classifier to distinguish between the original and augmented samples. While simple and efficient, these pixel-space synthesis techniques often fail to match the visual and semantic characteristics of real industrial defects. Recent work addresses this limitation by synthesizing anomalies directly in the feature space or by adapting pretrained networks to the target domain. For instance, feature adaptors fine-tune pretrained CNNs to reduce domain bias and improve the relevance of anomaly signals [26]. Other approaches generate near-in-distribution perturbations that better approximate realistic defect patterns [8, 27, 28].

Despite these advances, several open challenges remain. The most prominent include achieving representations that are both discriminative for subtle local defects and robust under domain shift, generating realistic synthetic anomalies that provide meaningful supervision, and designing architectures that balance detection accuracy with real-time efficiency. Addressing these challenges motivates recent hybrid approaches that integrate geometry-aware representation learning, efficient context aggregation, and realistic feature-space anomaly synthesis, which form the foundation for our proposed method.

1.2. Industrial 3D Anomaly Detection

Anomaly detection in three-dimensional (3D) data is an emerging research area that complements the more mature literature on two-dimensional (2D) methods [29]. Depth sensors provide rich geometric and structural cues that can improve defect localization and diagnosis, but they also impose unique algorithmic and practical challenges. Recent work has addressed these opportunities and challenges using multimodal fusion, purely geometric reasoning, self-supervised reconstruction, and generative denoising paradigms.

One line of work combines geometric descriptors with image-based semantic cues to leverage complementary modalities. The Complementary Pseudo Multimodal Feature (CPMF) framework integrates handcrafted local point cloud features with global semantic information obtained from pseudo-view projections processed by pretrained networks [30]. Related multimodal methods fuse geometric and photometric information to increase robustness under varying capture conditions and to improve localization accuracy [31, 32]. These approaches exploit the strengths of both modalities, but they may depend on careful alignment between 2D and 3D domains and on the availability of reliable color or intensity channels.

A second category focuses on methods that operate directly in the 3D domain and that therefore avoid cross-modal alignment issues. The 3D Student-Teacher method (3D-ST) extended the student–teacher paradigm to point clouds and demonstrated that a student trained to imitate a self-supervised teacher can localize geometric anomalies with a single forward pass [33]. Registration- and memory-based schemes, such as Reg3D-AD, combine raw coordinates with learned features from masked point cloud encoders (for example PointMAE) to build neighborhood-sensitive memory banks that represent normal geometry [34]. Group3AD represents anomalies with group-level feature prototypes and enforces inter-cluster uniformity and intra-cluster alignment to improve discriminability [35]. These geometric approaches remove dependencies on image-based cues and can provide fine-grained localization, but several such methods incur substantial computational and memory overhead due to large memory banks, registration steps, or complex clustering procedures.

Self-supervised reconstruction and generative models form a third family of approaches for 3D anomaly detection. IMRNet uses iterative mask reconstruction with geometry-aware sampling to preserve potentially anomalous structures during downsampling and employs a transformer to reconstruct masked patches in a self-supervised manner [36]. R3D-AD adopts a diffusion-based reconstruction strategy that learns point-wise displacements to convert anomalous inputs into their normal counterparts [37]. These generative methods can produce high-fidelity reconstructions and enable pixel- or point-level anomaly scoring, but they typically require iterative inference, careful regularization, and substantial compute to achieve stable results.

Despite these advances, practical deployment in industrial settings remains challenging. First, many state-of-the-art 3D methods are computationally intensive or memory hungry, which complicates integration into production inspection pipelines that require high throughput. Second, publicly available 3D anomaly datasets are limited in scale and diversity, and this scarcity of labeled anomalies places a premium on robust self-supervised and synthetic-augmentation strategies. Third, real-world data often exhibits occlusion, variable sampling density, and sensor noise, all of which can confound methods that assume well-behaved geometric inputs. These limitations motivate designs that explicitly address computational efficiency, domain adaptation, and robustness to capture artifacts.

Motivated by the simplicity and efficiency of SimpleNet in the 2D setting [26], we aim to develop an equally compact and practical architecture for 3D point cloud anomaly detection. Directly adapting SimpleNet to point clouds is not straightforward because point clouds are unordered and irregular, and because they require permutation-invariant operations and explicit modeling of local geometric relationships. In addition, 3D scans present higher dimensionality, variable point density, and sensor noise that demand specialized feature extraction and domain adaptation. To address this we ...

1.3. State Space Model

State-space models (SSMs) formulate sequence processing as learned linear dynamical systems and have recently been adapted into neural layers for long-range dependency modeling [38, 39, 40, 41, 42]. Early foundational work introduced HiPPO-style projection frameworks that give principled recurrent memory for very long horizons [38]. Building on these ideas, the Structured State Space for Sequence modeling (S4) family exploited algebraic structure and frequency-domain computation to make deep SSMs both expressive and efficient on long-horizon benchmarks [39]. Follow-up studies improved practical usability by reparameterizing layers, adopting multi-input multi-output constructions, and stabilizing initialization so that SSM layers can be trained in parallel and at scale [40, 41, 42].

Parallel to advances in attention and convolutional architectures, researchers have investigated how SSM blocks can serve as alternatives or complements in vision pipelines. Compared to self-attention-based transformers [43], SSM layers can offer lower memory footprints and near-linear runtime when applied to long token sequences, which is attractive for high-resolution inputs. Recent papers adapt the basic SSM toolkit to two-dimensional signals by converting images or feature maps into ordered token streams, adding positional encodings, and designing bidirectional or multi-scale scans to recover spatial context [44, 45, 46]. These vision-focused adaptations report competitive results

on classification and restoration tasks while reducing per-batch memory and speeding throughput in many settings [44, 45, 46].

SSM-driven designs have also been extended to geometric and low-level vision problems. For image restoration, lightweight SSM variants applied on multi-scale features achieve good fidelity-versus-latency trade-offs by exploiting the linear recurrence to propagate information across large receptive fields. For point-cloud processing, several works linearize spatial neighborhoods using carefully chosen traversal orders and then apply SSM modules to capture long-range geometric relationships; additional local aggregation or constrained parameterizations are used to preserve fine-grained geometry [47, 48, 49]. These studies demonstrate that state-space layers are versatile primitives that can be tailored to both dense pixel-level tasks and irregular 3D data.

Our contribution is complementary to these lines of work. Rather than replacing pretrained visual backbones or retraining large models from scratch, we design a compact, parameter-efficient adapter that embeds Mamba-style SSM processing into intermediate stages of an off-the-shelf backbone. The adapter fuses cross-stage representations and propagates long-range context with little extra compute, enabling improved feature integration while keeping the backbone weights frozen. In this way, our proposal bridges algorithmic advances in SSMs and practical, low-cost integration into large-scale vision systems.

2. Methodology

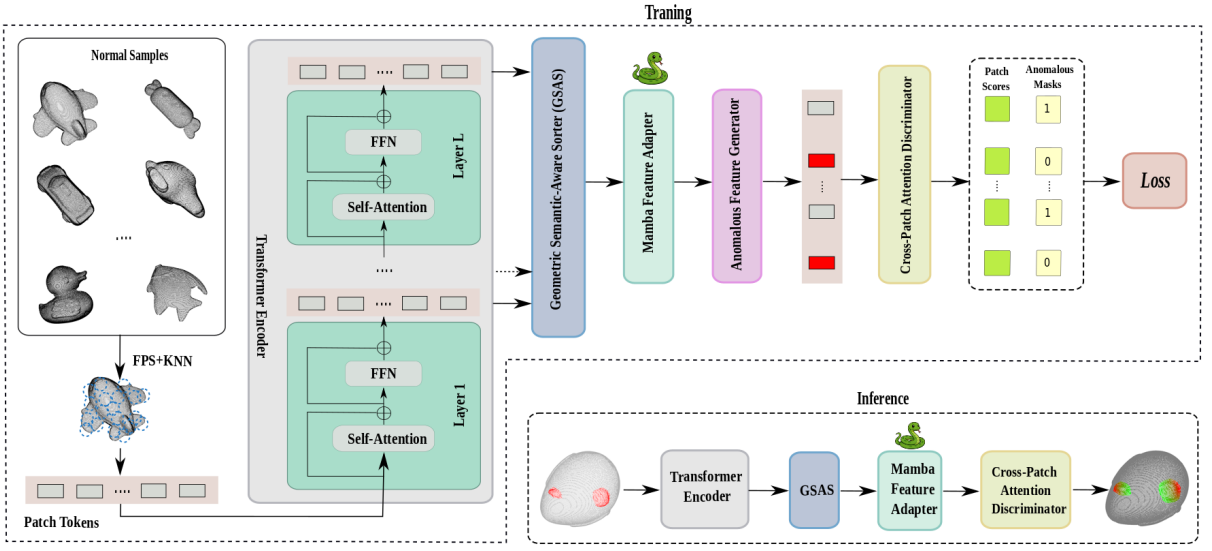

Figure 1: Overview of the 3D anomaly detection pipeline.

Given an input point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^{N}$, the proposed framework leverages a frozen pre-trained point-cloud Transformer and a small set of lightweight trainable adapters to obtain spatially coherent and anomaly-sensitive representations. The Transformer backbone consists of L encoder layers, each producing M patch tokens of dimension D. We denote the patch-token matrix and class token at Transformer layer i as $\mathbf{T}_i \in \mathbb{R}^{M\times D}$ and $c_i \in \mathbb{R}^{1\times D}$, respectively. Collectively, the hierarchical token set is $\mathbf{T} = \{\mathbf{T}_1, \ldots, \mathbf{T}_L\}$ with a total of $T = L \cdot M$ tokens. Our goal is to transform these unordered layer-wise tokens into a semantically ordered sequence $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T\times D}$ via the proposed **Geometric Semantic-Aware Sorter (GSAS)**, fuse them using a state-space adapter (Mamba) to produce enriched contextual features $y_{1:T} \in \mathbb{R}^{T\times D}$, and finally apply a lightweight attention-based discriminator for per-patch anomaly localization. To maintain parameter efficiency, all Transformer weights remain frozen; only the GSAS, Mamba adapters, the selective anomalous feature generator, and the cross-patch attention discriminator are trainable.

2.1. Backbone

Local patch extraction is performed once per input cloud and the resulting patch identities are preserved across Transformer layers. Specifically, we apply Farthest Point Sampling (FPS) to select M patch centers $\{p_j \in \mathbb{R}^3\}_{j=1}^{M}$

(indices j refer to patch centers), and for each center we collect a fixed-size local neighborhood via K-nearest neighbors (KNN). Each neighborhood is embedded by a lightweight PointNet encoder followed by a small positional-encoding MLP to produce an initial patch embedding $x_j^{(0)} \in \mathbb{R}^D$. These M patch embeddings (plus a global class token) are then fed into the frozen Point-MAE Transformer [50] pre-trained on ShapeNet [51]. Formally, the i-th Transformer block $\ell_i(\cdot)$ computes

$$[c_i;\, \mathbf{T}_i] \;=\; \ell_i([c_{i-1};\, \mathbf{T}_{i-1}]), \qquad i = 1, \ldots, L, \tag{1}$$

where $\mathbf{T}_i = [t_{i,1}; \ldots; t_{i,M}] \in \mathbb{R}^{M\times D}$ and $c_i \in \mathbb{R}^{1\times D}$. Importantly, the spatial coordinates $\{p_j\}$ (the patch centers) are associated with the corresponding patch index j and are reused for all Transformer layers, so that token $t_{i,j}$ at layer i corresponds to the same geometric patch center p_j.

2.2. Geometric Semantic-Aware Sorter (GSAS)

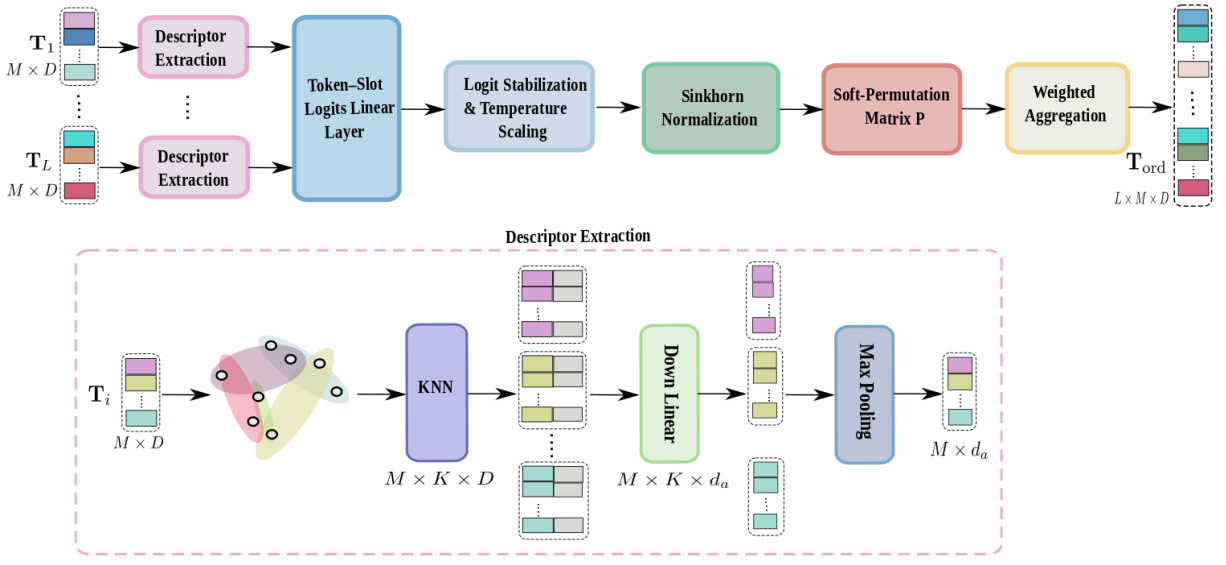

Figure 2: GSAS.

Point clouds admit no canonical linear ordering in $\mathbb{R}^3$; nevertheless, sequence models (e.g., state-space architectures) require an ordered input. The Geometric Semantic-Aware Sorter (GSAS) produces a differentiable, geometry- and semantics-aware ordering of the full set of Transformer patch tokens such that the resulting ordered sequence preserves local surface adjacency and semantic affinity while remaining end-to-end trainable. Below we give a precise, dimensionally consistent description of GSAS and the auxiliary regularizers that prevent degenerate assignments.

We assume patch extraction (FPS + KNN) is performed once per input cloud and that the resulting M unique patch centers $\{p_j \in \mathbb{R}^3\}_{j=1}^M$ are reused across Transformer layers. The frozen Transformer produces per-layer patch tokens $t_{i,j} \in \mathbb{R}^D$ for layer $i \in \{1, \ldots, L\}$ and patch index $j \in \{1, \ldots, M\}$. We enumerate the full set of $T = L \cdot M$ tokens with a single global index $t \in \{1, \ldots, T\}$ via a bijection $(i, j) \mapsto t$ (and denote the inverse mappings $i(t), j(t)$ when needed). The patch center associated with global token t is therefore $p_{j(t)}$ (and we may write p_t as shorthand, noting $p_t = p_{j(t)}$). GSAS maps the unordered set $\{t_{i,j}\}$ to an ordered sequence $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T\times D}$ using a differentiable soft-permutation matrix $P \in [0, 1]^{T\times T}$ (rows = tokens, columns = ordered slots).

Permutation-invariant local descriptors (graph-on-patches, within-layer aggregation). Construct a kNN graph on the unique patch centers $\{p_j\}_{j=1}^M$; denote the neighbor set of patch j by $\mathcal{N}_{\text{patch}}(j) \subset \{1, \ldots, M\}$. For a given token t corresponding to (i, j) (i.e., $i = i(t), j = j(t)$), we compute a permutation-invariant local descriptor by aggregating features from the same Transformer layer i of neighboring patches (i.e., tokens $t_{i,u}$ for $u \in \mathcal{N}_{\text{patch}}(j)$). Explicitly:

$$u_{i,j} \;=\; \text{MaxPool}\Big(\big\{ W_\downarrow\, t_{i,u} + b_\downarrow \;\big|\; u \in \mathcal{N}_{\text{patch}}(j) \big\} \Big) \in \mathbb{R}^{d_a}, \tag{2}$$

where $t_{i,u} \in \mathbb{R}^D$ is the token at layer i of neighbor patch u, $W_\downarrow \in \mathbb{R}^{d_a \times D}$, $b_\downarrow \in \mathbb{R}^{d_a}$, and the MaxPool is element-wise across the neighbor set. Using the bijection $(i, j) \mapsto t$ we set $E_{t,:} = u_{i(t),j(t)}$ to collect descriptors into $E \in \mathbb{R}^{T \times d_a}$.

Token-slot affinities and differentiable soft-permutation. We set the number of ordered slots equal to the total token count, $S_{\text{slots}} = T$. The raw token-slot logits (affinities) are computed as

$$\mathbf{G} \;=\; E\, W_\uparrow + \mathbf{1}\, b_\uparrow^\top \in \mathbb{R}^{T \times T}, \tag{3}$$

where $W_\uparrow \in \mathbb{R}^{d_a \times T}$, $b_\uparrow \in \mathbb{R}^T$, and $\mathbf{1} \in \mathbb{R}^{T \times 1}$ is an all-ones column. Entry $\mathbf{G}_{t,s}$ is the unnormalized affinity of token t to ordered slot s.

To obtain a differentiable approximately-permutation matrix we apply temperature-scaled exponentiation followed by the Sinkhorn operator. For numeric stability subtract the column-wise maximum before exponentiation: let $v \in \mathbb{R}^T$ be the vector with entries $v_s = \max_t \mathbf{G}_{t,s}$. Then

$$\tilde{\mathbf{G}} \;=\; (\mathbf{G} - \mathbf{1}\, v^\top)/\tau \in \mathbb{R}^{T \times T}, \quad \text{and} \quad P \;=\; \text{Sinkhorn}\big(\exp(\tilde{\mathbf{G}}),\, K_{\text{sink}}\big) \in [0,1]^{T \times T}, \tag{4}$$

where $\tau > 0$ is a temperature (smaller τ yields sharper assignments) and $\text{Sinkhorn}(\cdot, K_{\text{sink}})$ denotes K_{sink} iterations of alternating row- and column-normalization producing an (approximately) doubly-stochastic matrix. During training this soft-permutation is differentiable; at inference time one may optionally discretize P (e.g., via the Hungarian algorithm on $-\mathbf{G}$) if a hard permutation is required.

Weighted aggregation into ordered slots. Given the token feature matrix $X \in \mathbb{R}^{T \times D}$ with rows $X_t = t_{i(t),j(t)}$, the ordered-token matrix is obtained by weighted aggregation into slots:

$$\mathbf{T}_{\text{ord}} \;=\; P^\top X \in \mathbb{R}^{T \times D}, \qquad \mathbf{T}_{\text{ord}}[s, :] = \sum_{t=1}^{T} P_{t,s}\, X_t. \tag{5}$$

Because P is (approximately) doubly-stochastic, this yields one feature vector per ordered slot; the weighted aggregation is differentiable so gradients propagate to all GSAS parameters $W_\downarrow, b_\downarrow, W_\uparrow, b_\uparrow$.

Auxiliary regularizers for locality and assignment sharpness. To avoid degenerate (diffuse or collapsed) assignments and to encourage geometric locality in the induced order we add two auxiliary regularizers.

Entropy penalty. A column-wise entropy penalty discourages diffuse assignments into a given slot:

$$\mathcal{L}_{\text{ent}} \;=\; \frac{1}{T} \sum_{s=1}^{T} H(P_{:,s}), \qquad H(\pi) = -\sum_t \pi_t \log(\pi_t + \epsilon_{\text{ent}}), \tag{6}$$

where $\epsilon_{\text{ent}} > 0$ is a small constant for numerical stability (and the natural logarithm is used). This term encourages each slot to receive a concentrated assignment.

Locality penalty (normalized). Let $\tilde{s} = \dfrac{s-1}{\max(1, T-1)} \in [0,1]$ denote the normalized slot coordinate (so slot positions are scale-invariant to T and safely defined when $T = 1$), and define the expected (normalized) slot index for token t as

$$\mu_t \;=\; \sum_{s=1}^{T} \tilde{s}\, P_{t,s} \in [0,1]. \tag{7}$$

Denote by $j(t)$ the patch index associated with token t. Define a geometric affinity between patches via

$$w_{t,u} \;=\; \exp\big(-\|p_{j(t)} - p_{j(u)}\|^2/\sigma_p^2\big), \tag{8}$$

where σ_p is a bandwidth parameter. To avoid density bias we average the locality penalty per token:

$$\mathcal{L}_{\text{loc}} \;=\; \sum_{t=1}^{T} \frac{1}{\sum_{u=1}^{T} w_{t,u} + \epsilon_{\text{norm}}} \sum_{u=1}^{T} w_{t,u} \left(\mu_t - \mu_u\right)^2, \tag{9}$$

where $\epsilon_{\text{norm}} > 0$ is a small constant for stability in the normalization. Because $w_{t,u}$ depends only on the underlying patch centers $\{p_j\}$, tokens corresponding to the same patch (different layers) naturally receive maximal affinity; if a design choice requires discounting same-patch across-layer pairs one may set $w_{t,u} = 0$ when $j(t) = j(u)$ — otherwise the above encourages tokens from the same patch to be placed nearby in the final order.

Both $\mathcal{L}_{\text{ent}}$ and $\mathcal{L}_{\text{loc}}$ are weighted by small coefficients $\alpha_{\text{ent}}, \alpha_{\text{loc}} > 0$ and incorporated into the overall training loss (see Sec. 2.5).

Remarks on shapes and stability. All shapes and operations above are dimensionally consistent: $E \in \mathbb{R}^{T\times d_a}$, $W_\uparrow \in \mathbb{R}^{d_a\times T}$ producing $\mathbf{G} \in \mathbb{R}^{T\times T}$, $\exp(\cdot)$ is elementwise, Sinkhorn returns $P \in \mathbb{R}^{T\times T}$, and $P^\top X$ yields $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T\times D}$. The kNN graph is explicitly constructed over the unique patch centers $\{p_j\}$ (shared across layers) and neighborhood aggregation pools neighbor tokens *within the same Transformer layer* by design, which enforces geometric consistency while keeping layer semantics separated during descriptor computation. For numerical stability we recommend subtracting the per-column maximum prior to exponentiation (as shown), using ϵ_{ent} inside the entropy log, and using ϵ_{norm} in the locality normalization.

2.2.1. Mamba Feature Adapter

After GSAS produces the ordered token matrix $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T\times D}$, let the row-wise sequence be

$$x_{1:T}, \qquad x_t = \mathbf{T}_{\text{ord}}[t, :] \in \mathbb{R}^D, \quad t = 1, \dots, T.$$

The Mamba adapter is formulated as a state-space model that fuses long-range, layer-wise context along the GSAS-induced ordering. Let the latent dimension be S. The recurrence and output equations are defined as

$$h_t = A\, h_{t-1} + B\, x_t, \qquad h_0 = \mathbf{0} \in \mathbb{R}^S, \tag{10}$$

$$\tilde{q}_t = C\, h_t + D\, x_t \in \mathbb{R}^D, \tag{11}$$

where

$$A \in \mathbb{R}^{S\times S}, \quad B \in \mathbb{R}^{S\times D}, \quad C \in \mathbb{R}^{D\times S}, \quad D \in \mathbb{R}^{D\times D}.$$

The sequence of adapted outputs is collected as

$$Q = [\tilde{q}_1, \dots, \tilde{q}_T]^\top \in \mathbb{R}^{T\times D},$$

and, for consistency with the subsequent modules, $q_t \equiv \tilde{q}_t$.

The GSAS module determines the sequence order $x_{1:T}$, while the Mamba adapter performs contextual integration along this induced order without altering the token arrangement. The parameters $\{A, B, C, D\}$ are shared across all time steps, ensuring parameter efficiency and preserving the linear computational complexity with respect to sequence length T. Specifically, Mamba operates in $\mathcal{O}(T)$ time and requires $\mathcal{O}(S)$ additional memory per step, in contrast to the $\mathcal{O}(T^2)$ cost of full self-attention. The residual term $D\, x_t$ maintains per-token fidelity, while $C\, h_t$ introduces long-range contextual dependencies through the recurrent hidden state. The Mamba parameters are optimized jointly with the GSAS parameters, positional MLP, anomaly-scaling γ, and discriminator weights, whereas the Transformer backbone remains frozen.

2.3. Anomalous Feature Generator

Industrial defects in 3D point clouds are typically sparse, subtle, and spatially localized. To simulate such defects during training without requiring external anomaly examples, we introduce a lightweight, selective feature-space perturbation that synthesizes pseudo-anomalous tokens by injecting noise into a sparse subset of adapted token embeddings. Given output from Mamba feature adapter Y.

We sample an independent Bernoulli mask per token to determine corruption:

$$m_t \sim \text{Bernoulli}(p), \qquad m_t \in \{0, 1\}, \tag{12}$$

where $p \in (0, 1)$ controls expected corruption sparsity and is selected on validation (to reflect realistic, sparse defect rates we typically choose $p \ll 0.5$ and tune it on held-out data). When $m_t = 1$ the token is corrupted by an additive isotropic Gaussian perturbation in the canonical token space:

$$\varepsilon_t \sim \mathcal{N}\left(0, \sigma^2 I_D\right), \tag{13}$$

where $\sigma > 0$ controls perturbation scale (we validate σ; a practical value is $\sigma = 0.1$). The pseudo-anomalous token is defined as

$$\tilde{q}_t \;=\; q_t + \gamma\, m_t\, \varepsilon_t, \tag{14}$$

where $\gamma \geq 0$ is a learnable scalar that adaptively rescales the injected perturbation (initialized to a small positive value). When $m_t = 0$ we have $\tilde{q}_t = q_t$. This corruption mechanism is applied only during training; at inference time the generator is disabled ($m_t \equiv 0$) and all tokens remain clean.

The selective (sparse) corruption contrasts with indiscriminate global noise: by corrupting a small fraction of slots we emulate localized defects while preserving the normal manifold for the majority of features. Because sparse corruption can induce class imbalance, we (i) recommend validating p on held-out data and (ii) mitigate imbalance via weighted loss or controlled sampling (details in the training section).

2.4. Cross-Patch Attention Discriminator

Detecting defects often requires contextual comparison across patches. We therefore employ a lightweight attention-based discriminator that jointly processes all patch slots and outputs a scalar logit per slot. To preserve spatial information and to generalize across varying patch arrangements, positional embeddings are derived from patch center coordinates rather than learned per-slot indices. Let $p_t \in \mathbb{R}^3$ denote the patch center associated with slot t; the positional embedding is computed by a small coordinate MLP:

$$e_t \;=\; \text{PE-MLP}(p_t) \in \mathbb{R}^D. \tag{15}$$

During training we construct a single mixed input sequence for the discriminator in which corrupted tokens replace clean tokens in-place. The discriminator input for slot t is therefore

$$\hat{q}_t \;=\; \big(m_t\, \tilde{q}_t + (1 - m_t)\, q_t\big) + e_t \;=\; q_t + m_t(\gamma \varepsilon_t) + e_t, \tag{16}$$

and the supervision label is $y_t = m_t$. Inference uses the same pipeline but with $m_t \equiv 0$ so that $\hat{q}_t = q_t + e_t$.

Let H denote the number of attention heads and $d_{\text{head}} = D/H$ the per-head dimension; in practice we choose D divisible by H or insert a thin linear projection to meet this constraint. The discriminator applies a single multi-head self-attention layer followed by a compact shared MLP head that maps each token to a scalar logit:

$$Z \;=\; \text{MHA}(\{\hat{q}_t\}_{t=1}^{T}) \in \mathbb{R}^{T \times D}, \tag{17}$$

$$s_t \;=\; \text{MLP}_{\text{head}}(Z_t) \in \mathbb{R}. \tag{18}$$

Here $\text{MHA}(\cdot)$ denotes standard multi-head self-attention with appropriate query/key/value projections, scaled dot-product attention, residual connections, dropout and layer normalization; MLP_{head} is a two-layer feedforward network (one hidden layer) shared across slots and producing a scalar logit. Layer normalization and dropout are included to stabilize training. Because the discriminator processes all tokens jointly, each output s_t reflects both local evidence and global context, improving sensitivity to structured or context-dependent defects compared to independent per-slot scoring.

At test time the discriminator receives only clean tokens $\{q_t\}$ augmented with positional embeddings e_t and outputs per-slot logits s_t. These per-slot scores are reprojected to the original point cloud by assigning each point the maximal score of the patches that contain it (see Inference section).

2.5. Loss Function and Training

We jointly optimize the trainable components (GSAS parameters, Mamba adapter parameters, positional-embedding MLP, the anomaly-scaling γ, and discriminator parameters) using a patch-wise binary classification objective. Let Θ denote the set of all trainable parameters. For a training batch we accumulate P supervised token slots (summed across batch and sequence). We use a numerically-stable logits-based binary cross-entropy implemented as BCE-with-logits. The per-slot logits-based loss (stable form) is:

$$\ell_{\text{BCElogits}}(s, y) \;=\; \max(s, 0) - s\, y + \log\big(1 + \exp(-|s|)\big), \tag{19}$$

and the unweighted batch loss is $\mathcal{L}_{\text{BCE}} = \frac{1}{P} \sum_{t=1}^{P} \ell_{\text{BCElogits}}(s_t, y_t)$. To mitigate class imbalance when corruption is sparse we optionally use a positive-class weighting factor $w_+ > 0$ (e.g., $w_+ = (1-p)/p$ or a value tuned on validation)

via the standard `pos_weight` mechanism of BCE-with-logits; this multiplies the contribution of positive (corrupted) examples.

We complement $\mathcal{L}_{\text{BCE}}$ with the GSAS regularizers $\mathcal{L}_{\text{ent}}$ and $\mathcal{L}_{\text{loc}}$ (defined in Sec. 2.2) to discourage diffuse soft assignments and to encourage geometric locality in the induced ordering. The total objective is therefore

$$\mathcal{L} \;=\; \mathcal{L}_{\text{BCE}} \;+\; \alpha_{\text{ent}}\,\mathcal{L}_{\text{ent}} \;+\; \alpha_{\text{loc}}\,\mathcal{L}_{\text{loc}}, \tag{20}$$

where $\alpha_{\text{ent}}, \alpha_{\text{loc}} \geq 0$ are small weighting coefficients selected on validation. Weight decay and parameter regularization are handled via AdamW's decoupled weight-decay (i.e., we pass the weight-decay hyperparameter to the optimizer rather than applying an explicit ℓ_2 penalty inside $\mathcal{L}$). In practice we use AdamW with initial learning rate 10^{-4}, cosine annealing over 100 epochs, batch size 8, and gradient clipping (norm limit 1). Standard 3D augmentations (random yaw, point jittering, scaling) are applied prior to patch extraction.

During training the discriminator receives a single mixed set of tokens (some corrupted in-place, others clean) and predicts per-slot logits s_t with labels $y_t = m_t$. We do not present both clean and corrupted duplicates of the same cloud in the same forward pass; the single-stream mixed-input design matches inference-time behavior and simplifies batching.

2.6. Inference and Scoring Function

At inference the Anomalous Feature Generator is disabled (i.e., $m_t \equiv 0$ and $\tilde{q}_t = q_t$). The deterministic pipeline is: extract patches and patch centers $\{p_j\}$ (shared across layers), compute Transformer tokens and apply GSAS to obtain an ordered sequence, run the Mamba adapter to obtain adapted tokens q_t, compute positional embeddings $e_t = \text{PE-MLP}(p_t)$, and evaluate the trained discriminator to obtain logits s_t. Optionally convert logits to probabilistic intensities via $\sigma(s_t)$.

To reproject per-slot scores to the original point cloud, assign each point $\mathbf{x}$ the maximum logit among patches that contain it:

$$s(\mathbf{x}) \;=\; \max_{t:\,\mathbf{x}\in\mathcal{P}_t} s_t, \tag{21}$$

where $\mathcal{P}_t$ denotes the set of input points belonging to patch t (from the KNN used at patch extraction). A 3D median filter or voxel-grid median aggregation may be applied to the resulting heatmap for spatial smoothing; the filter kernel size or voxel resolution is selected on validation. For segmentation the heatmap is thresholded at τ_{seg} (chosen on validation), and for global anomaly detection we compute the cloud score

$$s_{\text{cloud}} \;=\; \max_{t} s_t, \tag{22}$$

declaring the cloud anomalous if $s_{\text{cloud}} > \tau_{\text{pc}}$.

Implementation notes: we maintain a single canonical token dimension D throughout; positional embeddings are coordinate-derived via PE-MLP to generalize across patch arrangements; the discriminator head count H is chosen so D is divisible by H (or a thin projection is applied); and hyperparameters $p, \sigma, \gamma, \alpha_{\text{ent}}, \alpha_{\text{loc}}, w_+$ and median-filter settings are selected on validation and reported in the experimental section.

3. Evaluation

3.1. Datasets

We evaluate our method on two complementary benchmarks that together cover synthetic, large-scale variation and high-fidelity real scans. First, Anomaly-ShapeNet is a synthetic point-cloud benchmark built on top of ShapeNet-CoreV2 and intended to provide diverse, controllable defect examples for 3D anomaly detection [36, 51]. The released benchmark used in this paper contains 1,600 samples across 40 object categories. The authors synthesize six realistic defect types, namely bulge, concavity, hole, break, bending, and crack, with the anomalous region occupying about one to ten percent of the points in affected samples. Defects are created using Blender sculpting tools and the corresponding point-level ground truth masks are generated by geometric comparison tools and exported with CloudCompare [36].

Second, Real3D-AD is a high-precision, real-scan dataset collected for industrial 3D anomaly detection and benchmarking [34]. Real3D-AD contains 1,254 scanned objects spanning 12 categories (for example, airplane, car, candybar, diamond, seahorse and toffees). The point clouds are high density, with per-object point counts ranging from tens of thousands to on the order of two million points, and a point-to-point sampling resolution on the order of

0.0010 mm to 0.0015 mm. Scans were captured using a blue-light structured-light scanner (PMAX-S130) on a rotating turntable to obtain full 360 degree coverage; defects were labeled using a CloudCompare-based pipeline that leverages octree comparison plus manual verification to produce per-point anomaly masks [34]. The dataset is organized in a prototype-based training setup: each class provides a small set of pristine prototypes for training and separate test folders containing both normal and defective scans together with ground-truth masks and text annotations, which makes Real3D-AD suitable for evaluating prototype-driven and registration-based anomaly methods.

3.2. Implementation Details

We implement the complete framework in PyTorch[1] and execute all experiments on a workstation equipped with four NVIDIA RTX 3090 GPUs (24 GB each). Training is performed with mixed precision using NVIDIA AMP and with PyTorch Distributed Data Parallel (DDP) across the four devices. To improve reproducibility we fix the global random seed to 42 and control nondeterministic sources where possible; by default we keep cuDNN's nondeterministic kernels enabled for throughput, but experiments that require bitwise reproducibility can force deterministic behavior. Checkpoints, training logs, and evaluation scripts are provided in the public repository.

All raw scans are preprocessed with voxel-grid filtering to keep memory and compute tractable: we use a voxel size of 0.0005 m for Real3D-AD and 0.001 m for Anomaly-ShapeNet, matching the dataset recommendations. After voxelization each cloud is translated to zero mean and scaled to unit radius. During training we apply on-the-fly augmentations before patch extraction: random yaw rotation sampled uniformly in $[0, 2\pi)$, isotropic Gaussian jitter with standard deviation $\sigma = 0.005$ m (clipped at 2σ), uniform scaling in the range $[0.95, 1.05]$, light point dropout up to 5% of points, and a small random translation ± 0.002 m. These augmentations increase robustness while preserving patch identity across Transformer layers within a forward pass.

Patch extraction uses Farthest Point Sampling (FPS) to select $M = 256$ patch centers and K-nearest neighbors with $K = 512$ points per patch, unless noted otherwise. Each neighborhood is encoded by a lightweight PointNet-style encoder followed by a two-layer positional-encoding MLP to produce patch embeddings of dimension D, where D is set to match the pretrained Transformer token size (Point-MAE defaults to $D = 768$ in our experiments). The Transformer backbone (Point-MAE pretrained on ShapeNet) is kept frozen in all runs; only the Geometric Semantic-Aware Sorter (GSAS), the Mamba adapter, the positional-MLP, the anomaly-scaling scalar γ, and the attention discriminator are trainable. For extremely large scans (e.g., Real3D objects that produce very high T after unfolding layers and patches) we reduce M or increase the voxel size during preprocessing to keep memory usage bounded.

The GSAS intra-layer aggregation descriptor dimension is $d_a = 128$. Token-slot logits are produced by a linear layer $W_\uparrow \in \mathbb{R}^{d_a \times T}$ where $T = L \cdot M$ is the total token count across all backbone layers; in practice T ranges from a few thousand up to tens of thousands depending on the backbone depth L and patch count M. For the Sinkhorn soft-permutation we use temperature $\tau = 0.05$ and $K_{\text{sink}} = 20$ iterations; the geometric-affinity bandwidth is set to $\sigma_p = 0.05$ in unit-radius coordinates, and numerical stability values are $\epsilon_{\text{ent}} = \epsilon_{\text{norm}} = 10^{-8}$. Reasonable default weights for the auxiliary regularizers are $\alpha_{\text{ent}} = 10^{-2}$ and $\alpha_{\text{loc}} = 10^{-1}$; these coefficients are tuned on the validation split. When a crisper ordering is desired for deployment we lower τ and/or increase K_{sink}, and for very long sequences the Sinkhorn iterations can be truncated or replaced by blockwise assignment to trade off runtime and ordering accuracy.

The Mamba adapter uses a latent state size $S = 256$ in our default configuration. To promote stable recurrent dynamics we initialize A as a scaled identity with spectral radius close to one (we use $A \approx 0.98\, I_S$ with small Gaussian perturbations) and initialize B, C, D with Kaiming normal initialization and small biases. These simple initialization choices proved reliable across datasets; substituting HiPPO-inspired parameterizations is also supported if desired. With the chosen S and D, the adapter contributes a modest parameter overhead, typically under 5% relative to the frozen backbone.

During training the anomalous feature generator corrupts tokens sparsely to emulate localized defects. We set the corruption probability to $p = 0.05$ by default and sample isotropic Gaussian perturbations with scale $\sigma = 0.1$; the adaptive rescaling parameter γ is learned but initialized to 0.1. To mitigate class imbalance induced by sparse corruption we either set the BCE-with-logits positive-class weight `pos_weight` to $(1-p)/p$ or tune a scalar w_+ on the validation split. The primary training objective is the numerically-stable logits-based binary cross-entropy averaged across supervised slots, augmented by the entropy and locality regularizers from GSAS.

[1] Code and dataset: https://github.com/hoangcuongbk80/Mamba3dAD

The attention-based discriminator uses $H = 8$ heads and a two-layer MLP head with hidden size $D/2$ and GELU activation. Dropout with rate 0.1 and layer normalization are applied inside the attention block and MLP head to stabilize optimization. If D is not divisible by H we apply a thin linear projection to the nearest divisible dimension to avoid per-head mismatches. All trainable modules are optimized jointly with AdamW using initial learning rate 1×10^{-4}, weight decay 1×10^{-2}, and $\beta = (0.9, 0.999)$. We perform a 2-epoch linear warmup from 1×10^{-5} followed by cosine annealing to zero over 100 epochs. Batch size is 8 total across GPUs in our standard runs (two samples per GPU); we clip gradients to norm 1.0 and use 8 dataloader workers per GPU to balance IO and CPU preprocessing.

At inference time the anomalous feature generator is disabled and the pipeline is deterministic: extract voxelized patches and centers, compute frozen-transformer tokens, apply GSAS to obtain an ordering, run the Mamba adapter to produce adapted tokens, add coordinate-derived positional embeddings, and evaluate the attention discriminator to obtain per-slot logits. We map per-slot logits back to points by taking the maximum score among the patches that contain each point and optionally apply a 3D median filter in voxel space for spatial smoothing; voxel-filter parameters and detection/segmentation thresholds (τ_{pc}, τ_{seg}) are selected on validation. For deployments that require deterministic hard orderings we optionally discretize the soft permutation using the Hungarian algorithm on $-\mathbf{G}$; for latency-sensitive settings one may instead reduce K_{sink}, run GSAS blockwise, or cache the patch-extraction step for static objects.

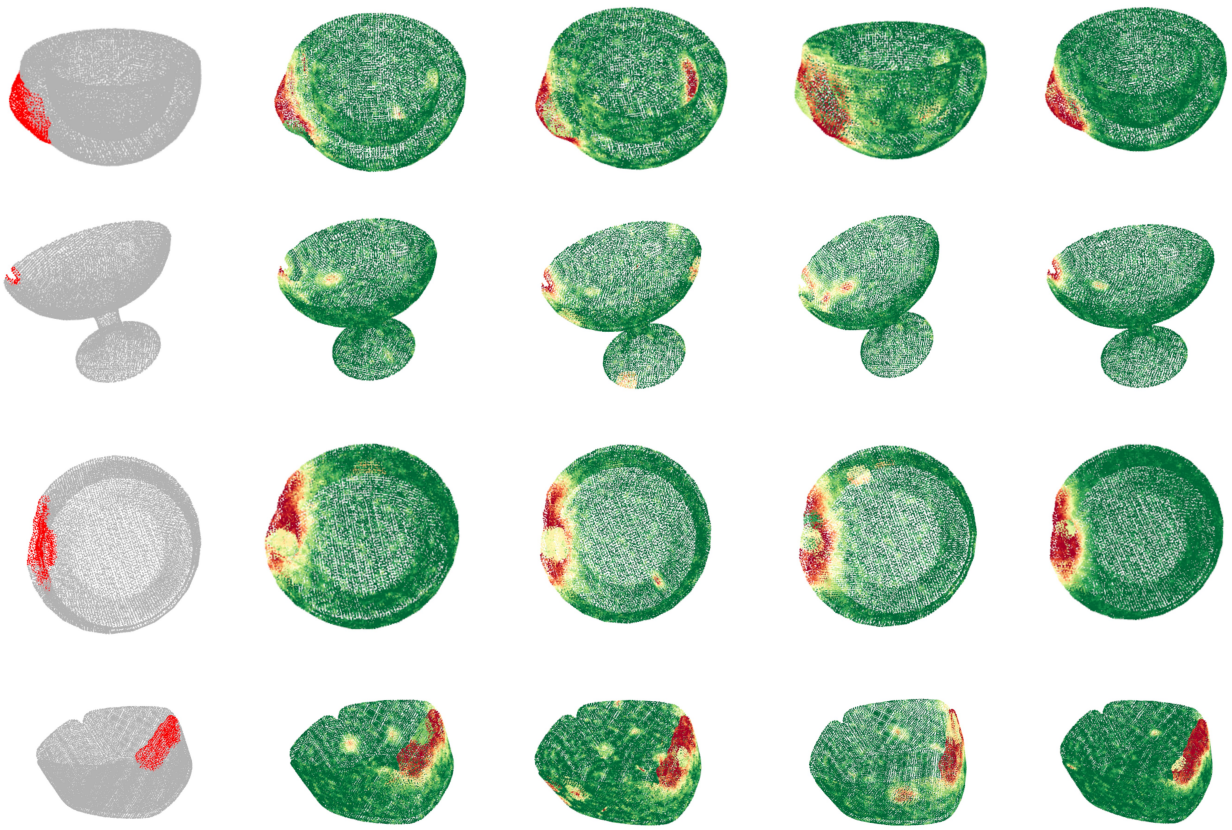

Figure 3: Qualitative results on the newly collected Anomaly-ShapeNet [36]. From left to right: input point clouds and ground truth annotations of anomalous points in red, and anomaly scores for each 3D point predicted by the proposed method.

3.3. Evaluation Metrics

We assess the performance of anomaly detection models using both object-level and point-level evaluation metrics derived from the receiver operating characteristic (ROC) and precision–recall (PR) curves. These complementary

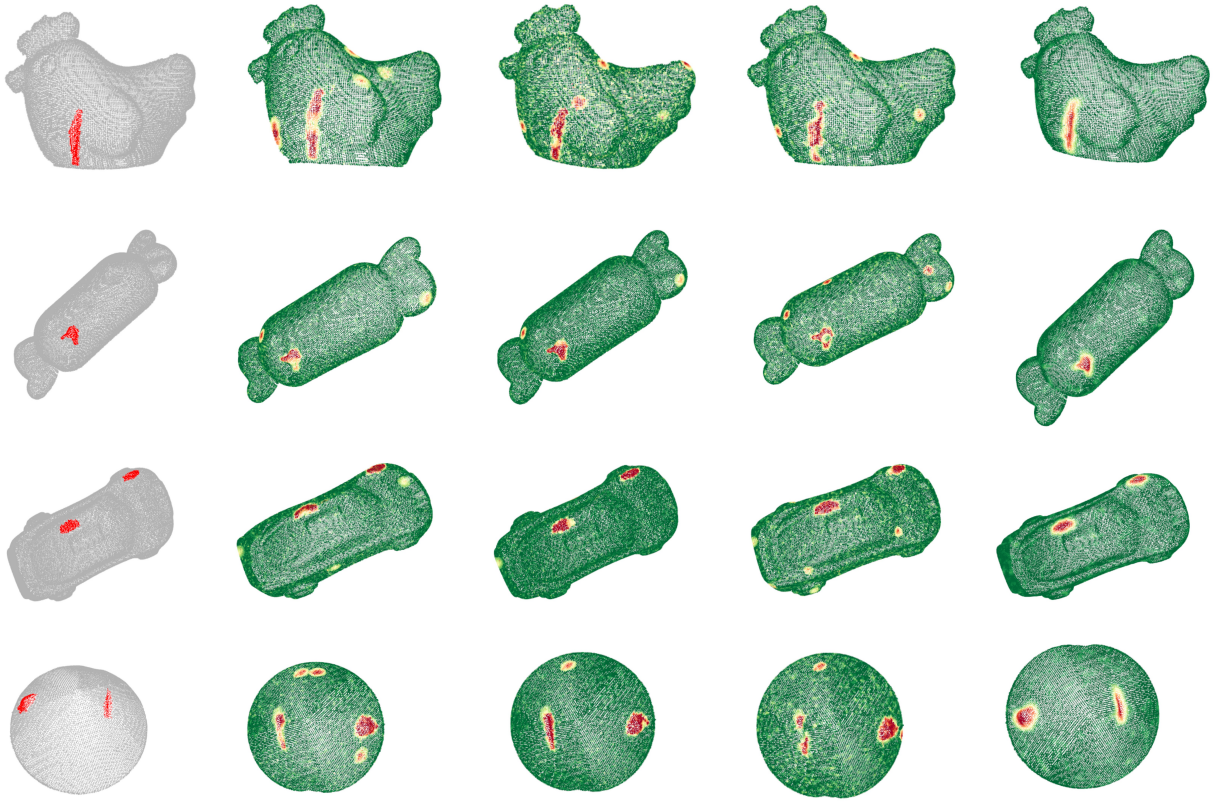

Figure 4: Qualitative results on Real3D-AD dataset. From left to right: input point clouds, Ground truth annotations of anomalous points in red, and anomaly scores for each 3D point predicted by the proposed method.

measures capture different aspects of detection performance: ROC-based metrics emphasize overall discrimination capability, while PR-based metrics are more sensitive to rare positive instances, as is typical in anomaly detection.

The ROC curve characterizes the trade-off between sensitivity and specificity by plotting the true positive rate (TPR) against the false positive rate (FPR) as the decision threshold varies. Let TP, FP, TN, and FN denote the number of true positives, false positives, true negatives, and false negatives, respectively. The TPR and FPR are defined as

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{23}$$

The area under the ROC curve (AUROC) summarizes the ROC curve into a single scalar measure,

$$\text{AUROC} = \int_0^1 \text{TPR}(\text{FPR})\, \text{dFPR}, \tag{24}$$

where an AUROC of 0.5 indicates random guessing, and a score of 1.0 denotes perfect discrimination. AUROC is threshold-independent and robust to class imbalance, which makes it widely used for both binary classification and anomaly detection.

While AUROC evaluates general separability between normal and anomalous samples, the PR curve focuses on the model's effectiveness in identifying the positive (anomalous) class. Precision and recall are defined as

$$\text{Precision } (P) = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{Recall } (R) = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{25}$$

Table 1
Average Results on Real3D-AD.

	Point Level		Object Level		Speed
	AUROC	AUPR	AUROC	AUPR	FPS
BTF(Raw)	0.571	0.022	0.603	0.611	2.05
BTF(FPFH)	0.730	0.064	0.635	0.614	1.01
M3DM(PointMAE)	0.637	0.046	0.552	0.572	0.30
M3DM(PointBERT)	0.636	0.052	0.538	0.581	0.50
PatchCore(FPFH)	0.577	0.071	0.593	0.591	0.09
PatchCore(FPFH+Raw)	0.680	0.123	0.682	0.667	0.09
PatchCore(PointMAE)	0.642	0.058	0.594	0.633	0.10
3D-ST [33]	0.705	0.109	0.645	0.723	1.50
Reg3D-AD [34]	0.705	0.109	0.704	0.723	0.08
Group3AD [35]	0.735	0.137	0.751	0.740	2.50
IMRNet [36]	0.725	0.166	0.725	0.625	5.60
R3D-AD [37]	0.592	0.041	0.734	0.632	7.12
Ours	0.763	0.194	0.781	0.775	13.52

The area under the precision–recall curve (AUPR) is computed as

$$\mathrm{AUPR} = \int_0^1 P(R)\,\mathrm{d}R, \tag{26}$$

providing a threshold-independent measure of anomaly retrieval performance. Because the baseline precision corresponds to the fraction of anomalies in the dataset, AUPR is particularly informative in highly imbalanced scenarios, reflecting a model's ability to retrieve true anomalies while minimizing false detections.

At the **object level**, each point cloud is treated as a single instance. A global anomaly score is typically obtained by aggregating per-point anomaly scores, for example using the maximum response across all points in the cloud. Object-level AUROC and AUPR are then computed over the entire set of test point clouds. At the **point level**, each point's predicted score is directly compared with its binary ground-truth label, yielding fine-grained evaluation of spatial localization performance. Together, these metrics provide a comprehensive assessment of both detection accuracy and localization precision across scales.

3.4. Result

Figures **??** and **??** present qualitative segmentation results on the Real3D-AD and Industrial3D-AD benchmarks, respectively, illustrating the model's ability to localize both bulges and sinks in high-precision scans and to robustly highlight subtle scratches, dents, and occlusion-induced artifacts under factory-like capture conditions. Table 1 presents a comparison between the proposed method and state-of-the-art approaches. BTF(Raw) refers to the use of only the raw 3D coordinate features (x, y, z) within the Back-to-Front (BTF) framework [29]. In contrast, BTF(FPFH) augments the same pipeline with Fast Point Feature Histograms (FPFH) [52]. The entries M3DM(PointMAE) and M3DM(PointBERT) correspond to the model [31] configured to ignore its RGB branch and instead extract point cloud features with PointMAE [50] or PointBERT [53], respectively. For PatchCore variants, PatchCore(FPFH) replaces the usual ResNet-based feature extractor with FPFH descriptors [52] before feeding them into the PatchCore anomaly scoring pipeline [21]. PatchCore(FPFH+Raw) further concatenates the raw spatial coordinates to each FPFH feature vector, and PatchCore(PointMAE) uses the PointMAE network [50] as the backbone feature extractor within the PatchCore framework.

At the point level, our model achieves an AUROC of 0.763 and an AUPR of 0.194, improving over the previous best, Group3AD [35], which attains 0.735 AUROC and 0.137 AUPR, by +2.8% and +5.7%, respectively. Notably, simpler baselines such as BTF(Raw) and BTF(FPFH) remain far behind (AUROC 0.571 and 0.730), indicating the necessity of integrating learned spatialdescriptor representations rather than relying solely on raw coordinates or hand-crafted FPFH features. At the object level, we observe a similar margin of improvement: our method yields an AUROC of 0.781 and an AUPR of 0.775, compared to Group3AD's 0.751 AUROC and 0.740 AUPR, corresponding to gains of +3.0% and +3.5%. Transformer-based or distillation-based approaches such as M3DM(PointMAE) [31], M3DM(PointBERT)

Table 2
Performance comparison on the newly collected Industrial3D-AD dataset in terms of point-level and object-level AUROC, AUPR, and inference speed (FPS).

	Point Level		Object Level		Speed
	AUROC	AUPR	AUROC	AUPR	FPS
BTF(Raw)	0.545	0.021	0.577	0.585	2.05
BTF(FPFH)	0.702	0.061	0.610	0.586	1.01
M3DM(PointMAE)	0.612	0.044	0.528	0.545	0.30
M3DM(PointBERT)	0.608	0.049	0.512	0.553	0.50
PatchCore(FPFH)	0.548	0.068	0.563	0.561	0.09
PatchCore(FPFH+Raw)	0.648	0.117	0.650	0.637	0.09
PatchCore(PointMAE)	0.615	0.055	0.567	0.604	0.10
3D-ST [33]	0.670	0.104	0.614	0.689	1.50
Reg3D-AD [34]	0.678	0.104	0.670	0.690	0.08
Group3AD [35]	0.701	0.131	0.716	0.708	2.50
IMRNet [36]	0.691	0.158	0.691	0.597	5.60
R3D-AD [37]	0.563	0.039	0.699	0.603	7.12
Ours	0.730	0.184	0.744	0.741	13.52

[53], and 3D-ST [33] show competitive localization accuracy but do not surpass our framework's balanced precision-recall trade-off, particularly in recall, as evidenced by their lower AUPR scores.

Table 2 shows results on our newly collected Industrial3D-AD dataset, which features a wider range of component geometries, finer defect scales, and material surface variations compared to Real3D-AD. At the point level, our method achieves 0.730 AUROC and 0.184 AUPR, outperforming the strongest baseline, Group3AD [35] (0.701 AUROC, 0.131 AUPR), by +2.9% and +5.3%. However, these scores are slightly lower than on Real3D-AD (0.763 AUROC, 0.194 AUPR), reflecting the increased challenge of detecting subtle anomalies on shiny or reflective industrial surfaces and irregular sensor noise patterns. At the object level, we observe 0.744 AUROC and 0.741 AUPR, again surpassing Group3AD (0.716 AUROC, 0.708 AUPR) by +2.8% and +3.3%, but trailing our Real3D-AD result (0.781 AUROC, 0.775 AUPR). The relative drop (4% in AUROC and 3% in AUPR) can be attributed to two factors: (1) the higher within-class variance in industrial parts, where small scratches or dents produce weaker point-wise cues that diffuse across object boundaries and (2) more pronounced domain shift between training and test partitions. New materials and lighting conditions appear in the evaluation split.

In terms of inference speed, our pipeline achieves 13.52 FPS, which is more than twice as fast as the next best method, R3D-AD [37] (7.12 FPS), and up to two orders of magnitude faster than memory-bank-based approaches such as the PatchCore variants [21], or reconstruction distillation methods including Reg3D-AD [34], which runs at 0.08 FPS. This level of efficiency highlights the lightweight design of our Spatial Context Aggregation and Feature Adaptor modules, both of which aggregate local and global context without relying on computationally expensive feature matching or multi-pass inference.

3.5. Ablation Study

To quantify the impact of every design choice and hyperparameter in our pipeline, we conduct all ablations under the same training setups and baseline settings (frozen Point-MAE backbone; Spatial Context Aggregation with $\bar{n} = 0.1n$, $K = 10$, $\tau = 0.5$, $\tau_c = 0.1$; Feature Adaptor MLP hidden size $C/2$, residual weight $\alpha = 0.5$; Anomalous Feature Generator corruption probability $p = 0.5$ and noise standard deviation $\sigma = 0.1$; 3D median filter radius 3 voxels at inference). We evaluate: (i) module removals: – SCA (omit Spatial Context Aggregation) and – FA (bypass the Feature Adaptor); (ii) SCA internals: No global propagation (skip inter-prototype fusion), No prototype snapping (disable final cosine re-assignment), Pure geometry (fuse only W^g) and Pure feature (fuse only W^z); (iii) learning setup: No pretraining (train transformer end-to-end); and (iv) core module baselines: AFG (SimpleNet), which perturbs all tokens with Gaussian noise rather than selectively, and Discriminator (SimpleNet), which replaces the attention-based discriminator with a per-token two-layer MLP.

Figure 5 illustrates that our full model consistently delivers the best and most stable point- and object-level AUROC/AUPR across all 22 test objects. Table 3 reports the impact of each variant on average point- and object-level AUROC/AUPR as well as inference speed.

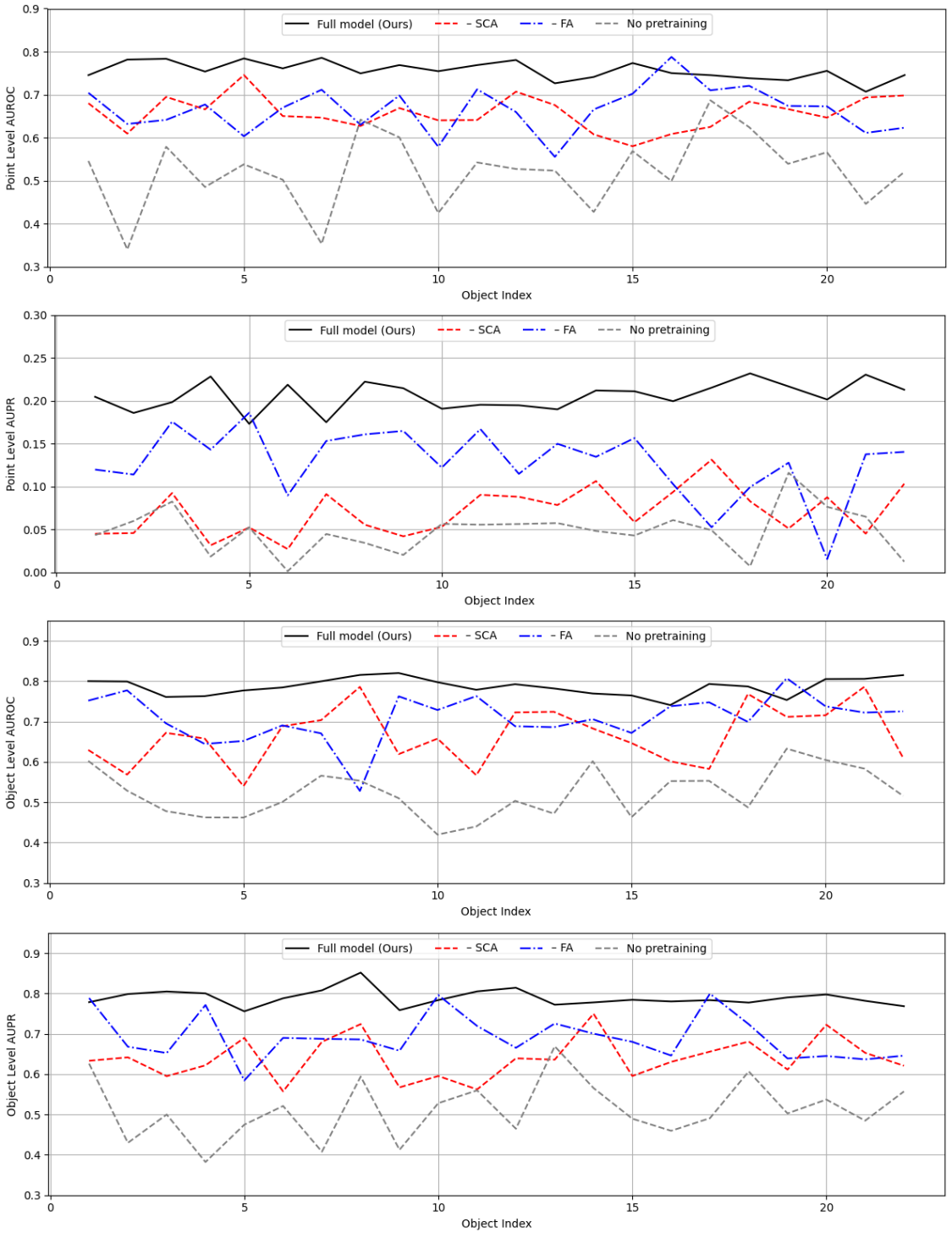

Figure 5: Object- and point-level performance comparison across 22 test objects. Four line plots show (top left) point-level AUROC, (top right) point-level AUPR, (bottom left) object-level AUROC, and (bottom right) object-level AUPR for the full model (black solid) and three ablated variants (- SCA in red dashed; - FA in blue dash-dot; no pretraining in gray dashed).

Table 3
Ablation Study: Average Results Across Datasets (AFG always present).

	Point Level		Object Level		Speed
	AUROC	AUPR	AUROC	AUPR	FPS
Full model (Ours)	0.750	0.181	0.770	0.775	13.52
- SCA	0.719	0.103	0.715	0.707	18.66
- FA	0.735	0.169	0.752	0.756	14.79
AFG (SimpleNet)	0.722	0.143	0.731	0.734	13.52
Discriminator (SimpleNet)	0.724	0.132	0.725	0.729	14.15
No global propagation	0.725	0.143	0.733	0.734	16.52
No prototype snapping	0.743	0.165	0.755	0.760	14.22
Pure geometry (no W^z)	0.731	0.149	0.734	0.736	14.46
Pure feature (no W^g)	0.733	0.145	0.736	0.732	14.72
No pretraining	0.504	0.053	0.525	0.530	13.52

The full model achieves the highest performance across all metrics, attaining a point-level AUROC of 0.750 and AUPR of 0.181, along with object-level scores of 0.770 AUROC and 0.775 AUPR, while operating at 13.52 frames per second (FPS). Among all variants, removing the Spatial Context Aggregation (SCA) module leads to the most substantial performance degradation. Specifically, point-level AUROC and AUPR drop to 0.719 and 0.103, respectively, and object-level AUROC and AUPR fall to 0.715 and 0.707. Although inference speed improves to 18.66 FPS without SCA, the significant accuracy loss underscores the essential role of local-global fusion in enhancing the expressiveness of geometric features and capturing subtle surface anomalies.

Eliminating the Feature Adaptor (FA) results in a more moderate decline in detection quality. In this configuration, point-level AUROC and AUPR decrease to 0.735 and 0.169, while object-level AUROC and AUPR reduce to 0.752 and 0.756. The slight improvement in inference speed to 14.79 FPS suggests that the adaptor introduces only a marginal computational cost. Nevertheless, these results indicate that aligning the pretrained backbone features with the target industrial domain enhances anomaly separability and provides measurable gains in precision, even if it is not strictly indispensable.

We also compare our selective Anomalous Feature Generator with a variant inspired by SimpleNet, which perturbs all patch tokens uniformly by adding Gaussian noise. This modification leads to a clear performance drop: point-level AUPR declines from 0.181 to 0.143 and object-level AUPR from 0.775 to 0.734. These results validate the advantage of our selective corruption strategy, where only a random subset of tokens is perturbed. By introducing sparse and localized feature distortions, the generator produces harder negative samples that more accurately simulate realistic defects and improve the training signal for the discriminator.

Replacing our attention-based Discriminator with a SimpleNet-style per-token two-layer MLP leads to further degradation in performance. The point-level AUPR drops to 0.132 and object-level AUPR to 0.729, with a modest increase in speed to 15.85 FPS. This decline highlights the importance of joint reasoning across patch tokens: cross-patch attention enables the model to capture spatial dependencies and contextual relationships between patches, which are particularly important for identifying semantic inconsistencies and subtle geometric anomalies. In contrast, treating each token independently limits the model's ability to distinguish structural outliers that manifest only in relation to their neighbors.

Further ablations explore the internal mechanisms of the Spatial Context Aggregation module. Disabling global propagation between prototypes results in notable performance loss, with AUPR reduced to 0.143 at the point level and 0.734 at the object level, even though speed increases to 16.52 FPS. This confirms that long-range context transfer among prototypes contributes meaningfully to the model's ability to reason about distant but structurally related regions. Disabling the final prototype snapping or restricting fusion to only geometric (W^g) or only learned feature (W^z) affinities yields milder declines in detection quality. For all these variants, point-level AUROC remains above 0.731 and AUPR above 0.145, while object-level scores also stay above 0.732. These findings indicate that both affinity modalities are beneficial and that the soft re-assignment step, though not critical alone, adds refinement that enhances overall robustness.

Finally, removing the pretraining on ShapeNet and training the Point-MAE backbone from scratch results in a complete collapse in detection performance. The point-level AUROC and AUPR drop to 0.504 and 0.053, while

object-level scores fall to 0.525 and 0.530, with no change in runtime. This stark contrast underscores the critical role of large-scale self-supervised pretraining in capturing generic 3D structural priors. Without this initialization, the model struggles to learn a meaningful notion of normalcy, especially under the weak supervision available in unsupervised anomaly detection.

This ablation study provides strong empirical support for our architectural choices. The Spatial Context Aggregation module and pretrained backbone are essential for accurate and reliable anomaly detection. The Feature Adaptor and global prototype interactions offer measurable improvements at minimal computational cost. Most notably, our selective pseudo-anomaly generation and attention-based discrimination outperform the corresponding SimpleNet variants by a clear margin, validating the core innovations of our method in terms of both accuracy and robustness in industrial 3D anomaly detection scenarios.

4. Conclusion

We have introduced an efficient framework for unsupervised 3D anomaly detection tailored for industrial applications. Our method combines a parameter-free Spatial Context Aggregation module, a lightweight Feature Adaptor for domain adaptation, and an Anomalous Feature Generator that synthesizes hard negatives in feature space. Built upon a pretrained Masked Autoencoder, the proposed approach effectively balances accuracy and efficiency without relying on memory banks or multi-pass inference. Our method achieves state-of-the-art results on both the Real3D-AD and Industrial3D-AD benchmarks. On Real3D-AD, we report a point-level AUROC of 0.763 and AUPR of 0.194, along with an object-level AUROC of 0.781 and AUPR of 0.775. On Industrial3D-AD, our framework achieves 0.730 AUROC and 0.184 AUPR at the point level, and 0.744 AUROC and 0.741 AUPR at the object level. These results reflect consistent improvements over the strongest prior baselines, with gains of up to 5.7% in AUPR and 3.0% in AUROC. Our pipeline also runs at 13.52 FPS, which, while not strictly real-time, represents a substantial speedup compared to existing 3D methods and is suitable for near-real-time inspection in industrial environments. In future work, we will optimize the model architecture and quantization schemes for deployment on edge devices to improve throughput in high speed production scenarios. We will also evaluate our approach with a wider variety of 3D sensors, such as structured light and time of flight cameras, to assess robustness under different noise and resolution characteristics. Finally, we plan to integrate our anomaly detector into robotic inspection systems and combine it with downstream tasks including 3D semantic mapping and automated manipulation to enable end to end quality control and corrective action in smart manufacturing environments.

References

[1] S. Venkataramanan, K.-C. Peng, R. V. Singh, A. Mahalanobis, Attention guided anomaly localization in images, in: European Conference on Computer Vision, Springer, 2020, pp. 485–503.

[2] C. Ding, G. Pang, C. Shen, Catching both gray and black swans: Open-set supervised anomaly detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 7388–7398.

[3] G. S. Chadha, A. Rabbani, A. Schwung, Comparison of semi-supervised deep neural networks for anomaly detection in industrial processes, in: 2019 IEEE 17th international conference on industrial informatics (INDIN), Vol. 1, IEEE, 2019, pp. 214–219.

[4] W.-H. Chu, K. M. Kitani, Neural batch sampling with reinforcement learning for semi-supervised anomaly detection, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16, Springer, 2020, pp. 751–766.

[5] Y. Shi, J. Yang, Z. Qi, DFR: Deep feature reconstruction for unsupervised anomaly segmentation, Neurocomputing 424 (2021) 9–22.

[6] J. Hou, Y. Zhang, Q. Zhong, D. Xie, S. Pu, H. Zhou, Divide-and-Assemble: Learning block-wise memory for unsupervised anomaly detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 8791–8800.

[7] V. Zavrtanik, M. Kristan, D. Skočaj, DRAEM-a discriminatively trained reconstruction embedding for surface anomaly detection, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 8330–8339.

[8] V. Zavrtanik, M. Kristan, D. Skočaj, DSR–a dual subspace re-projection network for surface anomaly detection, in: European conference on computer vision, Springer, 2022, pp. 539–554.

[9] A. De Nardin, P. Mishra, G. L. Foresti, C. Piciarelli, Masked transformer for image anomaly localization, International Journal of Neural Systems 32 (07) (2022) 2250030.

[10] X. Yao, R. Li, Z. Qian, Y. Luo, C. Zhang, Focus the discrepancy: Intra-and inter-correlation learning for image anomaly detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 6803–6813.

[11] W. Luo, H. Yao, W. Yu, Z. Li, AMI-Net: Adaptive mask inpainting network for industrial anomaly detection and localization, IEEE Transactions on Automation Science and Engineering (2024).

[12] X. Zhang, N. Li, J. Li, T. Dai, Y. Jiang, S.-T. Xia, Unsupervised surface anomaly detection with diffusion probabilistic model, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 6782–6791.

[13] F. Lu, X. Yao, C.-W. Fu, J. Jia, Removing anomalies as noises for industrial defect localization, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 16166–16175.

[14] J. Tebbe, J. Tayyub, Dynamic addition of noise in a diffusion model for anomaly detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 3940–3949.
[15] P. Bergmann, M. Fauser, D. Sattlegger, C. Steger, Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 4183–4192.
[16] M. Salehi, N. Sadjadi, S. Baselizadeh, M. H. Rohban, H. R. Rabiee, Multiresolution knowledge distillation for anomaly detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 14902–14912.
[17] Q. Wan, L. Gao, X. Li, L. Wen, Unsupervised image anomaly detection and segmentation based on pretrained feature mapping, IEEE Transactions on Industrial Informatics 19 (3) (2022) 2330–2339.
[18] Z. Gu, L. Liu, X. Chen, R. Yi, J. Zhang, Y. Wang, C. Wang, A. Shu, G. Jiang, L. Ma, Remembering normality: Memory-guided knowledge distillation for unsupervised anomaly detection, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 16401–16409.
[19] G. Tong, Q. Li, Y. Song, Enhanced multi-scale features mutual mapping fusion based on reverse knowledge distillation for industrial anomaly detection and localization, IEEE Transactions on Big Data 10 (4) (2024) 498–513.
[20] T. Defard, A. Setkov, A. Loesch, R. Audigier, PaDiM: a patch distribution modeling framework for anomaly detection and localization, in: International conference on pattern recognition, Springer, 2021, pp. 475–489.
[21] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, P. Gehler, Towards total recall in industrial anomaly detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 14318–14328.
[22] J. Bae, J.-H. Lee, S. Kim, PNI: Industrial anomaly detection using position and neighborhood information, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 6373–6383.
[23] Z. Zuo, Z. Wu, B. Chen, X. Zhong, A reconstruction-based feature adaptation for anomaly detection with self-supervised multi-scale aggregation, in: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2024, pp. 5840–5844.
[24] J. Hyun, S. Kim, G. Jeon, S. H. Kim, K. Bae, B. J. Kang, ReConPatch: Contrastive patch representation learning for industrial anomaly detection, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 2052–2061.
[25] C.-L. Li, K. Sohn, J. Yoon, T. Pfister, CutPaste: Self-supervised learning for anomaly detection and localization, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 9664–9674.
[26] Z. Liu, Y. Zhou, Y. Xu, Z. Wang, SimpleNet: A simple network for image anomaly detection and localization, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 20402–20411.
[27] Z. Fang, X. Wang, H. Li, J. Liu, Q. Hu, J. Xiao, FastRecon: Few-shot industrial anomaly detection via fast feature reconstruction, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 17481–17490.
[28] X. Zhang, M. Xu, X. Zhou, RealNet: A feature selection network with realistic synthetic anomaly for anomaly detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2024, pp. 16699–16708.
[29] E. Horwitz, Y. Hoshen, Back to the feature: classical 3d features are (almost) all you need for 3d anomaly detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 2968–2977.
[30] Y. Cao, X. Xu, W. Shen, Complementary pseudo multimodal feature for point cloud anomaly detection, Pattern Recognition 156 (2024) 110761.
[31] Y. Wang, J. Peng, J. Zhang, R. Yi, Y. Wang, C. Wang, Multimodal industrial anomaly detection via hybrid fusion, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 8032–8041.
[32] M. Rudolph, T. Wehrbein, B. Rosenhahn, B. Wandt, Asymmetric student-teacher networks for industrial anomaly detection, in: Proceedings of the IEEE/CVF winter conference on applications of computer vision, 2023, pp. 2592–2602.
[33] P. Bergmann, D. Sattlegger, Anomaly detection in 3d point clouds using deep geometric descriptors, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 2613–2623.
[34] J. Liu, G. Xie, R. Chen, X. Li, J. Wang, Y. Liu, C. Wang, F. Zheng, Real3D-AD: A dataset of point cloud anomaly detection, Advances in Neural Information Processing Systems 36 (2023) 30402–30415.
[35] H. Zhu, G. Xie, C. Hou, T. Dai, C. Gao, J. Wang, L. Shen, Towards high-resolution 3d anomaly detection via group-level feature contrastive learning, in: Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 4680–4689.
[36] W. Li, X. Xu, Y. Gu, B. Zheng, S. Gao, Y. Wu, 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, 2024, pp. 22207–22216.
[37] Z. Zhou, L. Wang, N. Fang, Z. Wang, L. Qiu, S. Zhang, R3D-AD: Reconstruction via diffusion for 3d anomaly detection, in: European Conference on Computer Vision, Springer, 2024, pp. 91–107.
[38] A. Gu, T. Dao, S. Ermon, A. Rudra, C. Ré, Hippo: Recurrent memory with optimal polynomial projections, Advances in neural information processing systems 33 (2020) 1474–1487.
[39] A. Gu, K. Goel, C. Ré, Efficiently modeling long sequences with structured state spaces, arXiv preprint arXiv:2111.00396 (2021).
[40] A. Gu, I. Johnson, K. Goel, K. Saab, T. Dao, A. Rudra, C. Ré, Combining recurrent, convolutional, and continuous-time models with linear state space layers, Advances in neural information processing systems 34 (2021) 572–585.
[41] A. Gu, K. Goel, A. Gupta, C. Ré, On the parameterization and initialization of diagonal state space models, Advances in Neural Information Processing Systems 35 (2022) 35971–35983.
[42] J. T. Smith, A. Warrington, S. W. Linderman, Simplified state space layers for sequence modeling, arXiv preprint arXiv:2208.04933 (2022).
[43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
[44] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, X. Wang, Vision mamba: efficient visual representation learning with bidirectional state space model, in: Proceedings of the 41st International Conference on Machine Learning, 2024, pp. 62429–62442.

[45] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, J. Jiao, Y. Liu, Vmamba: Visual state space model, Advances in neural information processing systems 37 (2024) 103031–103063.
[46] H. Guo, J. Li, T. Dai, Z. Ouyang, X. Ren, S.-T. Xia, Mambair: A simple baseline for image restoration with state-space model, in: European conference on computer vision, Springer, 2024, pp. 222–241.
[47] D. Liang, X. Zhou, W. Xu, X. Zhu, Z. Zou, X. Ye, X. Tan, X. Bai, Pointmamba: A simple state space model for point cloud analysis, Advances in neural information processing systems 37 (2024) 32653–32677.
[48] X. Han, Y. Tang, Z. Wang, X. Li, Mamba3d: Enhancing local features for 3d point cloud analysis via state space model, in: Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 4995–5004.
[49] Y. Zha, N. Li, Y. Wang, T. Dai, H. Guo, B. Chen, Z. Wang, Z. Ouyang, S.-T. Xia, Lcm: Locally constrained compact point cloud model for masked point modeling, Advances in Neural Information Processing Systems 37 (2024) 104816–104842.
[50] Y. Pang, W. Wang, F. E. Tay, W. Liu, Y. Tian, L. Yuan, Masked autoencoders for point cloud self-supervised learning, in: European conference on computer vision, Springer, 2022, pp. 604–621.
[51] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al., Shapenet: An information-rich 3d model repository, arXiv preprint arXiv:1512.03012 (2015).
[52] R. B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3d registration, in: 2009 IEEE international conference on robotics and automation, IEEE, 2009, pp. 3212–3217.
[53] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, J. Lu, Point-BERT: Pre-training 3d point cloud transformers with masked point modeling, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 19313–19322.