Multi-Scale Token Fusion with Mamba Adapters for Efficient 3D Industrial Defect Detection

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).

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 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

*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)

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

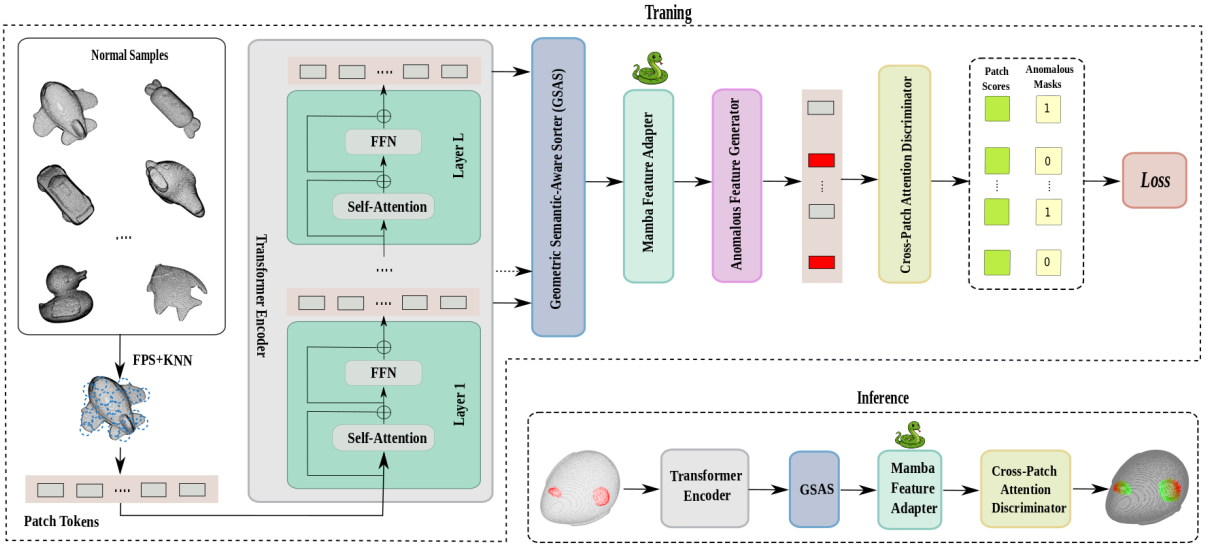

Figure 1: Overview of the proposed 3D anomaly detection pipeline.

We address efficient 3D anomaly detection that requires minimal task-specific training while preserving geometric structure and long-range contextual reasoning. As illustrated in Figure 1, given an input point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^{N}$, the proposed pipeline leverages a frozen pre-trained point cloud Transformer together with a compact set of lightweight adapters to generate spatially coherent and anomaly-sensitive representations. The backbone consists of L encoder layers, each producing M patch tokens of dimension D; we denote the patch-token matrix and class token at layer i as $\mathbf{T}_i \in \mathbb{R}^{M\times D}$ and $c_i \in \mathbb{R}^{1\times D}$, respectively, and collect the hierarchical token set $\mathbf{T} = \{\mathbf{T}_1, \dots, \mathbf{T}_L\}$ with a total of $T = L \cdot M$ tokens. These unordered layer-wise tokens are transformed into a semantically ordered sequence $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T\times D}$, subsequently fused into enriched contextual features $y_{1:T} \in \mathbb{R}^{T\times D}$, and finally scored by a lightweight attention-based discriminator for per-patch anomaly localization. The geometric semantic-aware sorter (GSAS) establishes a differentiable soft-permutation that orders the hierarchical tokens while preserving geometric locality and semantic consistency. The Mamba adapter, a linear-time state-space model, efficiently fuses the ordered sequence to yield context-enhanced features $y_{1:T}$ that retain both local fidelity and global awareness. A selective anomaly generator operates during training to inject sparse feature-space perturbations, enabling effective supervision without real defect labels. The cross-patch discriminator then aggregates contextual cues through a compact attention mechanism to output patch-level anomaly logits, which are reprojected to points for precise localization.

2.1. Backbone

Given an input point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^{N}$, the backbone produces local patch embeddings and a hierarchical set of Transformer tokens that preserve patch identities across layers. We apply Farthest Point Sampling (FPS) to select M patch centers $\{p_j \in \mathbb{R}^3\}_{j=1}^{M}$, where index j denotes a patch center, and for each center we collect a fixed-size neighborhood of K points via K-nearest neighbors (KNN). Each neighborhood is embedded by a lightweight PointNet encoder followed by a positional-encoding MLP to yield an initial patch embedding $x_j^{(0)} \in \mathbb{R}^D$, for $j = 1, \dots, M$. The M initial embeddings are concatenated with a global class token $c_0 \in \mathbb{R}^{1\times D}$ to form the Transformer input $[c_0; \mathbf{T}_0]$, where $\mathbf{T}_0 = [x_1^{(0)}; \dots; x_M^{(0)}] \in \mathbb{R}^{M\times D}$. These are passed to the frozen Point-MAE Transformer pretrained on ShapeNet [50, 51]. Formally, the i-th Transformer block $\mathscr{E}_i(\cdot)$ computes

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

with $\mathbf{T}_i = [t_{i,1}; \dots; t_{i,M}] \in \mathbb{R}^{M\times D}$ and $c_i \in \mathbb{R}^{1\times D}$. The spatial coordinates $\{p_j\}$ remain associated with their patch index j and are reused at every Transformer layer; consequently token $t_{i,j}$ at layer i corresponds to the same geometric

patch center p_j. The backbone therefore outputs the initial patch embeddings $\{x_j^{(0)}\}_{j=1}^{M}$ with $x_j^{(0)} \in \mathbb{R}^D$, the hierarchical tokens $\{\mathbf{T}_i\}_{i=1}^{L}$ with $\mathbf{T}_i \in \mathbb{R}^{M\times D}$, the sequence of class tokens $\{c_i\}_{i=1}^{L}$ with $c_i \in \mathbb{R}^{1\times D}$, and the retained patch centers $\{p_j\}_{j=1}^{M} \in \mathbb{R}^3$. Optionally, these tokens may be viewed as a flattened set $\mathbf{T} = \{\mathbf{T}_1, \dots, \mathbf{T}_L\}$ of $T = L \cdot M$ tokens in $\mathbb{R}^{T\times D}$.

2.2. Geometric Semantic-Aware Sorter (GSAS)

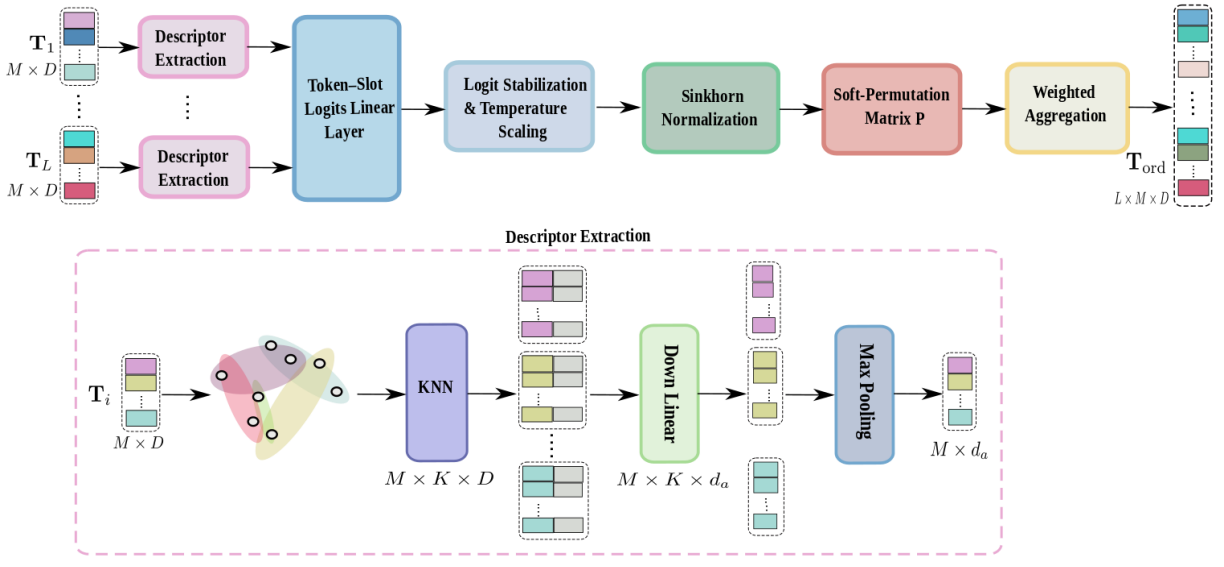

Figure 2: Overview of geometric semantic-aware sorter (GSAS).

Given input tokens $X \in \mathbb{R}^{T\times D}$, where X denotes the flattened collection of all per-layer tokens $t_{i,j}$ produced by the backbone (so that each $\mathbf{T}_i = [t_{i,1}; \dots; t_{i,M}] \in \mathbb{R}^{M\times D}$ and $T = L \cdot M$), and given the unique patch centers $\{p_j \in \mathbb{R}^3\}_{j=1}^{M}$, GSAS produces a differentiable ordering of the T tokens that preserves local surface adjacency and semantic affinity. As illustrated in Figure 2, we first summarizes local geometry and same-layer semantics, then computes token-slot affinities, converts affinities to an approximately doubly-stochastic assignment, and finally aggregates tokens into ordered slots for downstream fusion.

We construct a kNN graph on the unique patch centers and denotes the neighbor set of patch j by $\mathcal{N}_{\text{patch}}(j) \subset \{1, \dots, M\}$. For a token t corresponding to (i, j) with $i = i(t)$ and $j = j(t)$, GSAS computes a permutation-invariant local descriptor by aggregating features from the same Transformer layer i of neighboring patches:

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

where $W_{\downarrow} \in \mathbb{R}^{d_a\times D}$ and $b_{\downarrow} \in \mathbb{R}^{d_a}$. The module collects these descriptors into $E \in \mathbb{R}^{T\times d_a}$ by setting $E_{t,:} = u_{i(t),j(t)}$. The descriptor E summarizes local geometry while preserving layer-specific semantics, and GSAS uses E to inform the affinity computation in the next step. We computes raw token-slot affinities from the descriptors:

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

with $W_{\uparrow} \in \mathbb{R}^{d_a\times T}$, $b_{\uparrow} \in \mathbb{R}^{T}$, and $\mathbf{1} \in \mathbb{R}^{T\times 1}$. The matrix $\mathbf{G}$ encodes unnormalized affinities between tokens and ordered slots, and GSAS processes $\mathbf{G}$ to produce a differentiable assignment that can be optimized end-to-end.

For numerical stability we subtract the column-wise maximum $v \in \mathbb{R}^T$ with entries $v_s = \max_t \mathbf{G}_{t,s}$, applies temperature scaling and exponentiation, and normalizes with the Sinkhorn operator:

$$\tilde{\mathbf{G}} \;=\; (\mathbf{G} - \mathbf{1}\, v^{\top})/\tau, \tag{4}$$

$$P \;=\; \mathrm{Sinkhorn}\big(\exp(\tilde{\mathbf{G}}),\, K_{\mathrm{sink}}\big) \in [0,1]^{T\times T}, \tag{5}$$

where $\tau > 0$ denotes the temperature and K_{sink} the number of iterations. The soft-permutation P is approximately doubly-stochastic and differentiable, and GSAS uses P to aggregate token features into a semantically coherent ordered sequence. We aggregate tokens into ordered slots by weighted summation:

$$\mathbf{T}_{\mathrm{ord}} \;=\; P^{\top} X \in \mathbb{R}^{T\times D}, \tag{6}$$

$$\mathbf{T}_{\mathrm{ord}}[s,:] = \sum_{t=1}^{T} P_{t,s}\, X_t. \tag{7}$$

The aggregation yields one feature vector per ordered slot while preserving differentiability so that gradients propagate to $W_{\downarrow}, b_{\downarrow}, W_{\uparrow}, b_{\uparrow}$. The ordered tokens $\mathbf{T}_{\mathrm{ord}}$ therefore provide the contiguous, semantically coherent input required by the subsequent fusion module.

We include two auxiliary regularizers to prevent degenerate assignments and to encourage geometric locality. The column-wise entropy penalty discourages diffuse assignments into a given slot:

$$\mathcal{L}_{\mathrm{ent}} \;=\; \frac{1}{T}\sum_{s=1}^{T} H(P_{:,s}), \qquad H(\pi) = -\sum_{t} \pi_t \log(\pi_t + \epsilon_{\mathrm{ent}}), \tag{8}$$

with $\epsilon_{\mathrm{ent}} > 0$ for numerical stability. The entropy penalty directly biases the columns of P toward concentrated assignments and thus improves the quality of aggregation in Eq. for $\mathbf{T}_{\mathrm{ord}}$. The locality penalty measures expected slot positions μ_t and biases them to align with geometric affinities:

$$\mu_t \;=\; \sum_{s=1}^{T} \tilde{s}\, P_{t,s} \in [0,1], \tag{9}$$

$$w_{t,u} \;=\; \exp\big(-\|p_{j(t)} - p_{j(u)}\|^2/\sigma_p^2\big), \tag{10}$$

$$\mathcal{L}_{\mathrm{loc}} \;=\; \sum_{t=1}^{T} \frac{1}{\sum_{u=1}^{T} w_{t,u} + \epsilon_{\mathrm{norm}}} \sum_{u=1}^{T} w_{t,u}\,\big(\mu_t - \mu_u\big)^2, \tag{11}$$

where $\tilde{s} = (s-1)/\max(1, T-1)$, σ_p is a bandwidth parameter, and $\epsilon_{\mathrm{norm}} > 0$ stabilizes the normalization. The locality penalty directly influences μ_t and thus biases the soft-permutation P to place geometrically adjacent tokens into nearby slots.

2.2.1. Mamba Feature Adapter

Given ordered input tokens $\mathbf{T}_{\mathrm{ord}} \in \mathbb{R}^{T\times D}$, we denote the row-wise sequence by

$$x_{1:T}, \qquad x_t = \mathbf{T}_{\mathrm{ord}}[t,:] \in \mathbb{R}^D, \quad t = 1,\ldots,T. \tag{12}$$

The sequence $x_{1:T}$ encodes the GSAS-induced order and serves as the input that the Mamba recurrence integrates to produce context-enhanced outputs for each slot. The Mamba adapter is a state-space model with latent dimension S. The recurrence for the hidden state is

$$h_t = A\, h_{t-1} + B\, x_t, \qquad h_0 = \mathbf{0} \in \mathbb{R}^S, \tag{13}$$

where $A \in \mathbb{R}^{S\times S}$ and $B \in \mathbb{R}^{S\times D}$. The recurrence accumulates long-range, layer-wise context along the GSAS ordering, and the resulting hidden state h_t provides the global contextual information used by the output projection in the next equation. The Mamba output equation combines the recurrent state with a residual connection from the input:

$$\tilde{q}_t = C\, h_t + D\, x_t \in \mathbb{R}^D, \tag{14}$$

with $C \in \mathbb{R}^{D\times S}$ and $D \in \mathbb{R}^{D\times D}$. The term $D\, x_t$ preserves per-token fidelity while the term $C\, h_t$ injects long-range context into the adapted output $\tilde{q}_t$; this adapted output is the quantity consumed by downstream modules for anomaly

scoring. The adapter parameters are shared across time steps and collected as $\{A, B, C, D\}$. The parameter sharing ensures parameter efficiency and linear-time complexity; specifically, the module operates in $\mathcal{O}(T)$ time and requires $\mathcal{O}(S)$ additional memory per step, which contrasts with the $\mathcal{O}(T^2)$ cost of full self-attention. The linear complexity permits practical fusion of the flattened $T = L \cdot M$ tokens produced by the backbone and GSAS. The sequence of adapted outputs is assembled as

$$Q = [\tilde{q}_1, \ldots, \tilde{q}_T]^\top \in \mathbb{R}^{T \times D}, \tag{15}$$

and we set $q_t \equiv \tilde{q}_t$ for consistency with subsequent notation. The matrix Q provides the context-enriched features for each ordered slot and thus forms the direct input to the cross-patch discriminator and remaining heads.

2.3. Anomalous Feature Generator

Given adapted tokens $Q \in \mathbb{R}^{T \times D}$ produced by the Mamba adapter, where the rows are $q_t \in \mathbb{R}^D$ for $t = 1, \ldots, T$, the anomalous feature generator synthesizes sparse, feature-space perturbations to emulate localized defects during training. We generate a binary corruption mask, samples isotropic Gaussian perturbations in token space, and combines these quantities to produce corrupted tokens $\tilde{q}_t$ that serve as positive supervision for the discriminator. The module samples an independent Bernoulli mask per slot:

$$m_t \sim \text{Bernoulli}(p), \qquad m_t \in \{0, 1\}, \tag{16}$$

with $p \in (0, 1)$ controlling expected corruption sparsity. The mask m_t determines which slots receive perturbations; consequently the mask provides binary supervisory labels for corrupted versus clean slots during training and guides the subsequent noise injection. We samples additive isotropic Gaussian noise in the canonical token space:

$$\varepsilon_t \sim \mathcal{N}\big(0, \sigma^2 I_D\big), \tag{17}$$

with scale parameter $\sigma > 0$. The noise ε_t provides a generic, directionally unbiased perturbation in feature space; consequently the module uses this noise, when selected by m_t, to produce pseudo-anomalous embeddings that remain in the same representational space as the adapted tokens q_t. The module forms the pseudo-anomalous token by combining the adapted token, the mask, and a learnable rescaling:

$$\tilde{q}_t \;=\; q_t + \gamma\, m_t\, \varepsilon_t, \tag{18}$$

where $\gamma \geq 0$ denotes a learnable scalar that rescales the injected perturbation. The factor γ enables the model to adapt the effective perturbation magnitude during training; consequently the corrupted token $\tilde{q}_t$ preserves the original feature q_t when $m_t = 0$ and expresses a controlled deviation when $m_t = 1$. The generator is active only during training; at inference time the mask is fixed to $m_t \equiv 0$ and all tokens remain uncorrupted.

The design choice to inject sparse feature-space perturbations targets realistic industrial defects that are spatially localized and semantically subtle. The descriptor and fusion stages output context-enriched features q_t that already encode geometric and semantic information; therefore perturbing q_t in feature space produces anomalies that challenge the discriminator while avoiding unrealistic point-level artifacts. A global noise alternative would not preserve a normal manifold for the majority of slots and would reduce the signal-to-noise ratio for localization, and a per-layer independent corruption would not emulate cross-layer contextual inconsistencies that the discriminator must detect; consequently selective feature-space corruption provides a stronger and more realistic supervisory signal for per-slot anomaly localization. Because sparse corruption can induce class imbalance between corrupted and clean slots, we validate p on held-out data and mitigate imbalance via weighted loss or controlled sampling as described in the training procedure. The mask vector and corrupted tokens are therefore used directly as training inputs and labels for the cross-patch discriminator. The input to the anomalous feature generator is $Q \in \mathbb{R}^{T \times D}$ (rows $q_t \in \mathbb{R}^D$) and the primary outputs are the corrupted token matrix $\tilde{Q} = [\tilde{q}_1, \ldots, \tilde{q}_T]^\top \in \mathbb{R}^{T \times D}$ and the binary mask $m \in \{0, 1\}^T$. The corrupted tokens $\tilde{Q}$ and mask m are then consumed by the cross-patch discriminator for supervised anomaly learning during training; at inference the generator is disabled and $\tilde{Q} = Q$.

2.4. Cross-Patch Attention Discriminator

Given corrupted tokens $\tilde{Q} \in \mathbb{R}^{T \times D}$ and binary mask $m \in \{0, 1\}^T$ produced by the anomalous feature generator (training) or given clean tokens $Q \in \mathbb{R}^{T \times D}$ at inference, and given patch centers $\{p_t \in \mathbb{R}^3\}_{t=1}^T$, the discriminator produces contextualized per-slot logits $s_t \in \mathbb{R}$ for $t = 1, \ldots, T$. The discriminator jointly processes all slots to enable

context-aware anomaly scoring that leverages spatial positional information derived from patch centers. The module computes positional embeddings from patch centers:

$$e_t \;=\; \text{PE-MLP}(p_t) \in \mathbb{R}^D. \tag{19}$$

The positional embedding e_t provides explicit spatial context and the module uses e_t to augment token features before attention. We form the discriminator input by combining corrupted or clean tokens with positional embeddings:

$$\hat{q}_t \;=\; \big(m_t\,\tilde{q}_t + (1-m_t)\,q_t\big) + e_t \;=\; q_t + m_t(\gamma\varepsilon_t) + e_t. \tag{20}$$

The input $\hat{q}_t$ therefore contains the appropriate training-time corruption when $m_t = 1$ and otherwise equals the clean token plus spatial embedding; the module uses $\hat{q}_{1:T}$ as input to the attention layer.

The discriminator applies a single multi-head self-attention layer to jointly contextualize all slot inputs. The module denotes the number of heads by H and the per-head dimension by $d_{\text{head}} = D/H$. The module computes the contextualized features

$$Z \;=\; \text{MHA}(\{\hat{q}_t\}_{t=1}^{T}) \in \mathbb{R}^{T\times D}. \tag{21}$$

The multi-head attention mixes information across slots so that each row Z_t encodes both local evidence and global context; the module then uses Z to produce per-slot anomaly scores. We map contextualized features to scalar logits via a shared MLP head:

$$s_t \;=\; \text{MLP}_{\text{head}}(Z_t) \in \mathbb{R}. \tag{22}$$

The MLP head is shared across slots and the module uses the resulting logits $s_{1:T}$ as supervision targets for the training objective with labels $y_t = m_t$. The module employs standard regularization components in the attention block, including residual connections, layer normalization, dropout, and appropriate linear projections for queries, keys, and values. These components stabilize training and therefore improve the reliability of the contextualized features Z. At inference the anomalous feature generator is disabled so that $m_t \equiv 0$, and the discriminator input reduces to $\hat{q}_t = q_t + e_t$. The discriminator therefore outputs logits s_t that reflect contextual deviations from the learned normal manifold without injected corruption. The input to the discriminator is $\tilde{Q} \in \mathbb{R}^{T\times D}$ and $m \in \{0,1\}^T$ during training (or $Q \in \mathbb{R}^{T\times D}$ at inference) together with patch centers $\{p_t\}_{t=1}^{T}$. The primary outputs are the contextualized features $Z \in \mathbb{R}^{T\times D}$ and the per-slot logits $s \in \mathbb{R}^T$. The per-slot logits s are reprojected to the original point cloud by assigning each point the maximal score among patches that contain it and thereby produce the final point-level anomaly heatmap used for localization.

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. The set of all trainable parameters is denoted Θ. The training procedure accumulates P supervised token slots per batch, where P is the total number of labeled slots summed across batch and sequence, and minimizes a scalar objective with respect to Θ. We adopt a numerically stable logits-based binary cross-entropy implemented as BCE-with-logits. The per-slot stable loss is

$$\ell_{\text{BCElogits}}(s,y) \;=\; \max(s,0) - s\,y + \log\big(1+\exp(-|s|)\big), \tag{23}$$

where $s \in \mathbb{R}$ denotes the predicted logit for a single slot and $y \in \{0,1\}$ denotes the corresponding binary label. The stable form in $\ell_{\text{BCElogits}}$ prevents overflow in the exponential and therefore improves numerical robustness. The per-slot quantity $\ell_{\text{BCElogits}}(s_t, y_t)$ is aggregated to produce the batch-level classification loss used to update Θ. We form the unweighted batch classification loss as

$$\mathcal{L}_{\text{BCE}} \;=\; \frac{1}{P}\sum_{t=1}^{P} \ell_{\text{BCElogits}}(s_t, y_t), \tag{24}$$

where s_t and y_t denote the logit and label for the t-th supervised slot in the batch. The loss $\mathcal{L}_{\text{BCE}}$ provides the principal supervision signal that trains the discriminator and upstream adapters. The aggregated loss $\mathcal{L}_{\text{BCE}}$ therefore drives gradient flow into GSAS and Mamba via the discriminator outputs $s_{1:P}$.

We address class imbalance when corruption is sparse by optionally applying a positive-class weighting factor $w_+ > 0$ via the standard `pos_weight` mechanism of BCE-with-logits. The symbol w_+ denotes the weight applied to positive (corrupted) examples and the corruption probability used by the anomalous generator is denoted p in earlier sections. The positive-class weight w_+ multiplies the contribution of positive examples and therefore compensates for sparsity in y_t; consequently weighted BCE rebalances gradients toward scarce corrupted slots and improves training stability under sparse supervision. We complement the classification loss with GSAS regularizers that discourage diffuse soft assignments and encourage geometric locality. The total training objective is

$$\mathcal{L} \;=\; \mathcal{L}_{\text{BCE}} \;+\; \alpha_{\text{ent}}\,\mathcal{L}_{\text{ent}} \;+\; \alpha_{\text{loc}}\,\mathcal{L}_{\text{loc}}, \tag{25}$$

where $\mathcal{L}_{\text{ent}}$ denotes the column-wise entropy penalty on the soft-permutation P, $\mathcal{L}_{\text{loc}}$ denotes the locality penalty on expected slot positions μ_t, and $\alpha_{\text{ent}}, \alpha_{\text{loc}} \geq 0$ are small weighting coefficients selected on validation. The regularizer $\mathcal{L}_{\text{ent}}$ acts on columns of P to promote concentrated assignments and thus sharp aggregations in $\mathbf{T}_{\text{ord}}$. The regularizer $\mathcal{L}_{\text{loc}}$ acts on μ_t to bias ordering toward geometric adjacency and thus encourages the downstream Mamba fusion to exploit local continuity.

We optimize Θ using AdamW and apply decoupled weight decay through the optimizer rather than an explicit ℓ_2 penalty inside $\mathcal{L}$. The optimizer configuration uses initial learning rate 10^{-4}, cosine annealing over 100 epochs, batch size 8, and gradient clipping with norm limit 1. The optimizer and scheduling choices stabilize convergence and limit overfitting while retaining the deployment efficiency of the lightweight adapters. The training stream presents a single mixed set of tokens to the discriminator in which corrupted slots replace clean slots in-place and the supervision labels are $y_t = m_t$, where m_t denotes the corruption mask from the anomalous feature generator. The single-stream mixed-input design matches inference-time input statistics and reduces memory usage compared to presenting clean and corrupted duplicates in the same forward pass; consequently this design simplifies batching and ensures that the discriminator learns from inputs that reflect deployment conditions. The training inputs therefore comprise the per-slot logits $s \in \mathbb{R}^P$, the binary labels $y \in \{0,1\}^P$ (with $y_t = m_t$ during training), and the set of trainable parameters Θ. The primary optimization output is the scalar loss $\mathcal{L}$ whose minimization yields updated parameters Θ. The trained parameters Θ and the per-slot logits $s_{1:P}$ are then used at inference to produce final per-slot scores that are reprojected to a point-level anomaly heatmap for localization.

2.6. Inference and Scoring Function

At inference the anomalous feature generator is disabled so that $m_t \equiv 0$ and $\tilde{q}_t = q_t$ for all slots t. The system first extracts patches and patch centers $\{p_j\}_{j=1}^{M} \subset \mathbb{R}^3$ using the same FPS and KNN procedure used at training. The extracted patch centers provide spatial coordinates that are reused across Transformer layers and that later supply positional embeddings e_t for the discriminator. The frozen Transformer produces per-layer patch tokens $t_{i,j} \in \mathbb{R}^D$ for layers $i = 1, \ldots, L$ and patch indices $j = 1, \ldots, M$. The GSAS module then computes a differentiable ordering and yields the ordered token matrix $\mathbf{T}_{\text{ord}} \in \mathbb{R}^{T \times D}$, where $T = L \cdot M$. The ordered tokens $\mathbf{T}_{\text{ord}}$ preserves geometric locality and semantic affinity and is therefore suitable as the contiguous input required by the Mamba fusion stage.

The Mamba adapter integrates long-range context along the GSAS-induced order and produces adapted slot features $q_t \in \mathbb{R}^D$ for $t = 1, \ldots, T$. The Mamba outputs $Q = [q_1, \ldots, q_T]^\top \in \mathbb{R}^{T \times D}$ combine local fidelity with global context and are therefore the direct inputs to the discriminator. The positional-encoding MLP then computes spatial embeddings $e_t = \text{PE-MLP}(p_t) \in \mathbb{R}^D$ from the patch center coordinates p_t (here p_t denotes the center associated with slot t); these embeddings provide explicit spatial context that generalizes across patch arrangements. The trained discriminator processes the augmented inputs $\hat{q}_t = q_t + e_t$ and returns a scalar logit $s_t \in \mathbb{R}$ for each slot t. The scalar s_t denotes the uncalibrated anomaly score for slot t and the optional sigmoid $\sigma(s_t) = 1/(1+\exp(-s_t))$ converts logits to probabilistic intensities in $(0, 1)$ when required. The system reprojects per-slot logits to the original point cloud by assigning each input point $\mathbf{x} \in \mathbb{R}^3$ the maximum slot logit among patches that contain it. The per-point score $s(\mathbf{x})$ is therefore

$$s(\mathbf{x}) \;=\; \max_{t:\, \mathbf{x} \in \mathcal{P}_t} s_t, \tag{26}$$

where $\mathcal{P}_t$ denotes the set of input points assigned to patch t during KNN-based patch extraction. The max-reprojection preserves sharp localized responses and therefore highlights the most anomalous covering patch for each point.

We apply a spatial smoothing step to the point-level heatmap $s(\mathbf{x})$. A 3D median filter or voxel-grid median aggregation produces a smoothed heatmap and the smoothing parameters are selected on validation. The method

thresholds the smoothed heatmap at τ_{seg} to produce a binary segmentation mask for anomaly localization, where $\tau_{\text{seg}} \in \mathbb{R}$ denotes the segmentation threshold chosen on validation. We also computes a global cloud score

$$s_{\text{cloud}} = \max_{t=1,\dots,T} s_t, \tag{27}$$

and the method declares the cloud anomalous when $s_{\text{cloud}} > \tau_{\text{pc}}$, where $\tau_{\text{pc}} \in \mathbb{R}$ denotes the point cloud threshold selected on validation. Implementation details are preserved at inference: the canonical feature dimension is D, the discriminator uses H heads with per-head dimension $d_{\text{head}} = D/H$, and positional embeddings are coordinate-derived via PE-MLP to generalize across patch arrangements. The primary inference outputs are the per-slot logits $s \in \mathbb{R}^T$ and the point-level heatmap $s(\mathbf{x})$ defined for each input point $\mathbf{x}$; the per-slot logits s provide the direct scores used for cloud-level decision s_{cloud} and the heatmap $s(\mathbf{x}) \in \mathbb{R}^N$ (for N input points) provides the localization signal for segmentation and visualization.

3. Evaluation

3.1. Datasets

We evaluate our method on two public 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 ShapeNetCoreV2 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. 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

All experiments[1] were conducted on a workstation equipped with four NVIDIA RTX 3090 GPUs. All input scans were preprocessed with voxel-grid filtering to ensure tractable memory and computation. The voxel size was 0.0005 m for Real3D-AD and 0.001 m for Anomaly-ShapeNet. Each point cloud was translated to zero mean and scaled to unit radius after voxelization. The augmentation pipeline ran on the fly prior to patch extraction. The augmentations comprised random yaw rotation uniformly sampled from $[0, 2\pi)$, isotropic Gaussian jitter with standard deviation $\sigma_{\text{jitter}} = 0.005$ m clipped at $2\sigma_{\text{jitter}}$, uniform scaling in $[0.95, 1.05]$, random point dropout up to 5%, and random translation with magnitude at most 0.002 m. The implementation applied augmentations consistently within each forward pass so that patch identity remained stable across backbone layers during that pass.

Patch extraction used Farthest Point Sampling (FPS) to select $M = 256$ patch centers by default. Each center gathered up to $K_{\max} = 512$ nearest neighbors, and the effective neighborhood size was $K_{\text{eff}} = \min(K_{\max}, N)$, where N denotes the number of points remaining after voxelization. The implementation used $K_{\text{eff}} = 256$ as an alternative default for memory-constrained runs. Each neighborhood was encoded by a lightweight PointNet-style encoder followed by a two-layer positional-encoding MLP to produce embeddings in $\mathbb{R}^D$. The Transformer backbone was Point-MAE pretrained on ShapeNet and remained frozen in all experiments. The implementation collected tokens from all $L = 6$ encoder layers with embedding dimension $D = 768$. The implementation mapped the T tokens to a reduced ordered slot bank of size S for scalability. The default slot count was $S = 1024$. The implementation computed intra-layer descriptors $E \in \mathbb{R}^{T \times d_a}$ with $d_a = 128$ and learned a set of S slot queries $Q \in \mathbb{R}^{S \times d_a}$. Token-slot affinities

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

were obtained via a factorized dot product between E and Q with a learned per-slot bias, and the resulting affinities were converted to a numerically stabilized soft-assignment matrix $P \in [0,1]^{T\times S}$ by the Sinkhorn normalization. Ordered slot features were aggregated using the soft assignments in P. The Sinkhorn normalization employed conservative numerical stabilizations. The implementation subtracted column-wise maxima prior to exponentiation and executed Sinkhorn iterations in log-space for large matrices. The default temperature was $\tau = 0.2$ and the default iteration count was $K_{\text{sink}} = 8$. The implementation performed 64-bit accumulation for normalization when $T > 5000$ or $S > 2048$. The locality bandwidth used unit-radius coordinates with $\sigma_p = 0.05$. The stability constants were $\epsilon_{\text{ent}} = \epsilon_{\text{norm}} = 10^{-6}$. The GSAS regularization weights were $\alpha_{\text{ent}} = 10^{-2}$ and $\alpha_{\text{loc}} = 10^{-1}$, and these values were validated on held-out data. For very long sequences the implementation provided a blockwise-assignment fallback that partitions tokens spatially and executes GSAS per block to reduce peak memory.

Each ordered slot was assigned a spatial coordinate computed as the normalized P-weighted mean of the contributing patch centers. The implementation supplied this coordinate to the positional-encoding MLP to produce a positional embedding $e_s \in \mathbb{R}^D$ for each slot. The code maintained a consistent mapping among tokens, patch indices, and slots to enable accurate reprojection and visualization. The Mamba adapter processed the ordered slot sequence of length S. The adapter latent state dimension was $S_h = 256$. The implementation parameterized the transition matrix A as a scaled orthonormal operator with a learnable scalar initialized to 0.9 and with spectral norm constrained to 0.95. The input and output projection matrices B, C and the residual projection D used Kaiming normal initialization. The implementation applied layer normalization to the recurrent state and a gated residual connection between recurrent and residual pathways to stabilize dynamics. The adapter executed in linear time with respect to S and introduced less than 5% parameter overhead relative to the frozen backbone in the default configuration.

The anomalous feature generator sampled sparse corruptions per patch center and propagated the mask to all corresponding tokens across layers. The corruption probability was $p = 0.05$. The implementation normalized adapted slot features with LayerNorm prior to corruption. The isotropic Gaussian noise scale in normalized space was $\sigma_{\text{noise}} = 0.05$, and the adaptive rescaling parameter γ was learnable and initialized to 0.05. The generator was active only during training and was disabled during inference. The discriminator processed the S ordered slots using a single multi-head self-attention layer with $H = 8$ heads and per-head dimension D/H. Residual connections, layer normalization, and dropout with rate 0.1 were applied. The output head was a two-layer MLP with hidden size $D/2$ and GELU activation, producing scalar logits for each slot. Slot-level logits were reprojected to tokens, patches, and points using the soft-assignment matrix P. The implementation computed per-token scores as the weighted sum of slot logits using P, computed per-patch scores by taking the maximum over layer-specific tokens for each patch, and computed per-point scores by taking the maximum across all patches that contained the point. The reprojection pipeline preserved localization precision and remained differentiable with respect to P during training.

The training objective combined the numerically stable logits-based binary cross-entropy loss with the GSAS entropy and locality regularizers. The implementation mitigated class imbalance by setting the positive-class weight in the binary cross-entropy loss to $(1-p)/p$ by default or by tuning the positive-class weight on validation. The optimizer was AdamW with initial learning rate 1×10^{-4}, weight decay 1×10^{-2}, and momentum parameters $\beta = (0.9, 0.999)$. A two-epoch linear warmup from 1×10^{-5} preceded cosine annealing to zero over 100 epochs. The effective batch size was eight across four GPUs (two samples per GPU). Equivalent single-GPU runs used gradient accumulation and the repository documented the accumulation settings. The implementation clipped gradients to a maximum norm of 1.0 and used eight data-loader workers per GPU. At inference the anomalous feature generator was disabled and the pipeline executed deterministically. The implementation applied the same preprocessing, GSAS ordering, Mamba fusion, positional embedding, and discriminator evaluation as during training. The implementation optionally applied a voxel-space median filter with radius three voxels for spatial smoothing. Detection and segmentation thresholds were selected using validation data. The implementation supported two practical options for deterministic ordering in deployment: produce a near-binary assignment P by reducing τ and increasing K_{sink} and then take the column-wise arg max, or sort tokens by their expected slot positions computed from P; the implementation used the arg max strategy for moderate S and the expectation-based sort for very large S.

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 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.

Table 1
Average Result (%) on Anomaly-ShapeNet dataset.

	Point Level (%)		Object Level (%)		Speed
	AUROC	AUPR	AUROC	AUPR	FPS
BTF(Raw)	55.2	16.4	49.5	57.4	2.09
BTF(FPFH)	63.0	21.0	53.1	61.0	1.16
M3DM(PointMAE)	61.8	21.5	55.4	61.4	0.31
M3DM(PointBERT)	60.3	19.5	53.7	59.6	0.52
PatchCore(FPFH)	58.2	21.1	57.1	61.0	0.10
PatchCore(FPFH+Raw)	60.4	22.3	58.6	63.2	0.10
PatchCore(PointMAE)	57.9	19.4	56.4	59.3	0.12
Reg3D-AD [34]	67.0	20.3	57.4	70.3	0.10
3D-ST [33]	63.6	22.6	60.7	60.5	1.52
Group3AD [35]	84.6	25.4	81.4	95.3	2.55
IMRNet [36]	65.3	22.8	66.3	72.7	5.62
R3D-AD [37]	75.1	23.7	75.2	73.6	7.15
Ours	91.2	38.7	86.8	98.7	17.31

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

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}. \tag{28}$$

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

$$\mathrm{AUROC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})\, \mathrm{dFPR}, \tag{29}$$

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

$$\mathrm{Precision}\ (P) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \qquad \mathrm{Recall}\ (R) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{30}$$

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

$$\mathrm{AUPR} = \int_0^1 P(R)\, \mathrm{d}R, \tag{31}$$

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 on Anomaly-ShapeNet

Table 1 and Figure 3 summarize our quantitative and qualitative results on Anomaly-ShapeNet. Our method attains a point-level AUROC of 91.2% and point-level AUPR of 38.7%, which improves over the strongest baseline,

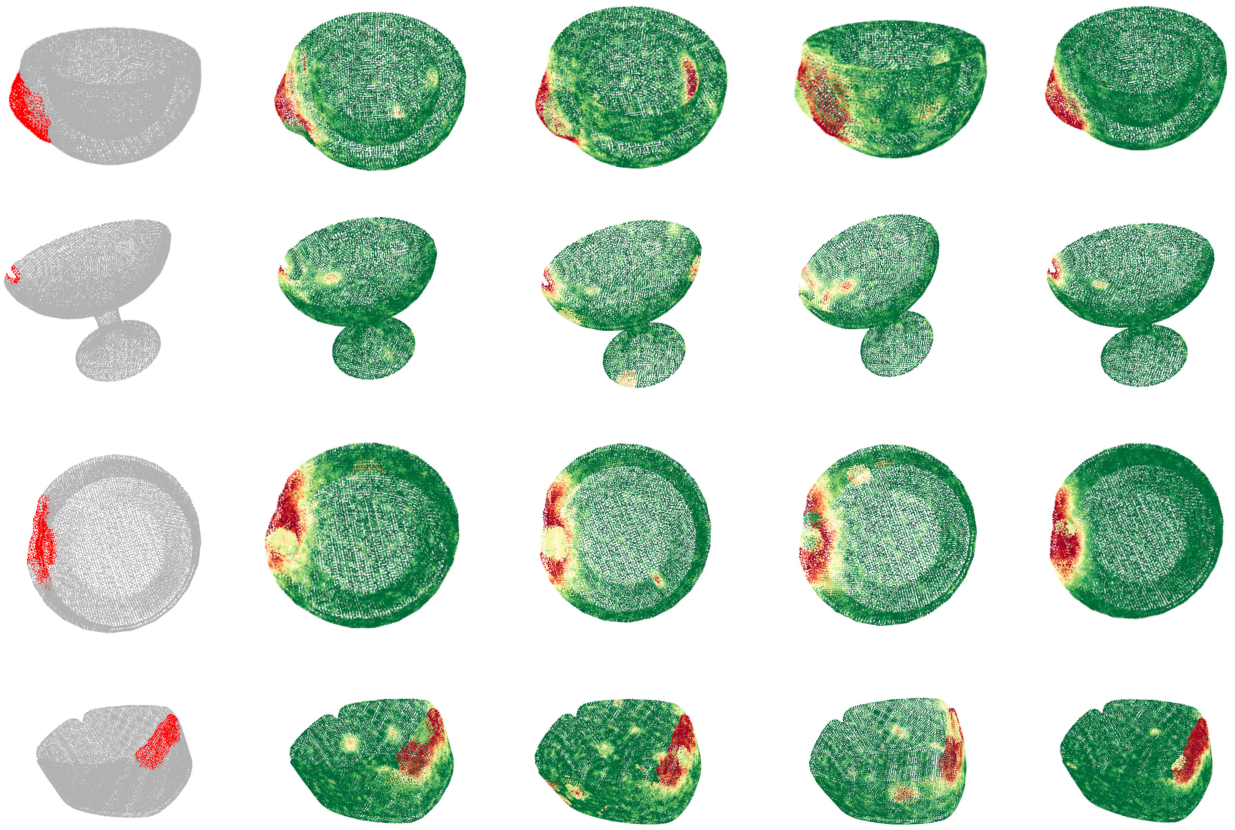

Figure 3: Qualitative results on the newly collected Anomaly-ShapeNet [36]. From left to right: input point clouds and ground truth annotations of anomalous points in red, and anomaly scores for each 3D point predicted by the proposed method.

Group3AD, by 6.6% and 13.3%, respectively. At the object level our AUROC is 86.8% and AUPR is 98.7%, exceeding Group3AD by 5.4% and 3.4%. These accuracy gains are achieved alongside a large runtime advantage: our default configuration runs at 17.31 FPS on the RTX-3090 workstation, which is substantially faster than the competing methods listed in Table 1.

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.

The observed improvements can be traced to three design choices. First, multi-scale token fusion with the Mamba adapter aggregates complementary cues across Transformer layers so that subtle local deviations are evaluated in their broader geometric context. Defects such as bulges and concavities manifest across receptive fields and benefit from the cross-layer evidence that Mamba supplies. Second, the Geometric Semantic-Aware Sorter (GSAS) produces a geometry-preserving ordering that enables state-space fusion to propagate information along coherent surface trajectories rather than arbitrary token sequences. This geometric consistency reduces spurious contextual mixing and sharpens localization. Third, the selective anomalous feature generator provides localized, feature-space

supervision that teaches the discriminator to attend to sparse, realistic deviations without corrupting global shape priors. Qualitatively, these components combine to produce compact heatmaps with low background noise for medium and large localized defects, as shown in Figure 3.

Failure modes are consistent with expectations for point-based pipelines. Very thin cracks that remove only a few points remain challenging and reduce AUPR because a small number of mis-scored points strongly affects precision. High-curvature ornamental details can sometimes be mistaken for defects, which suggests that future work could incorporate curvature-aware post-processing or augment the anomalous generator with structure-preserving perturbations. Overall, the results on Anomaly-ShapeNet demonstrate that the combination of GSAS, Mamba fusion, and sparse feature-space supervision yields both state-of-the-art detection and a deployment-friendly runtime.

3.5. Result on Real3D-AD

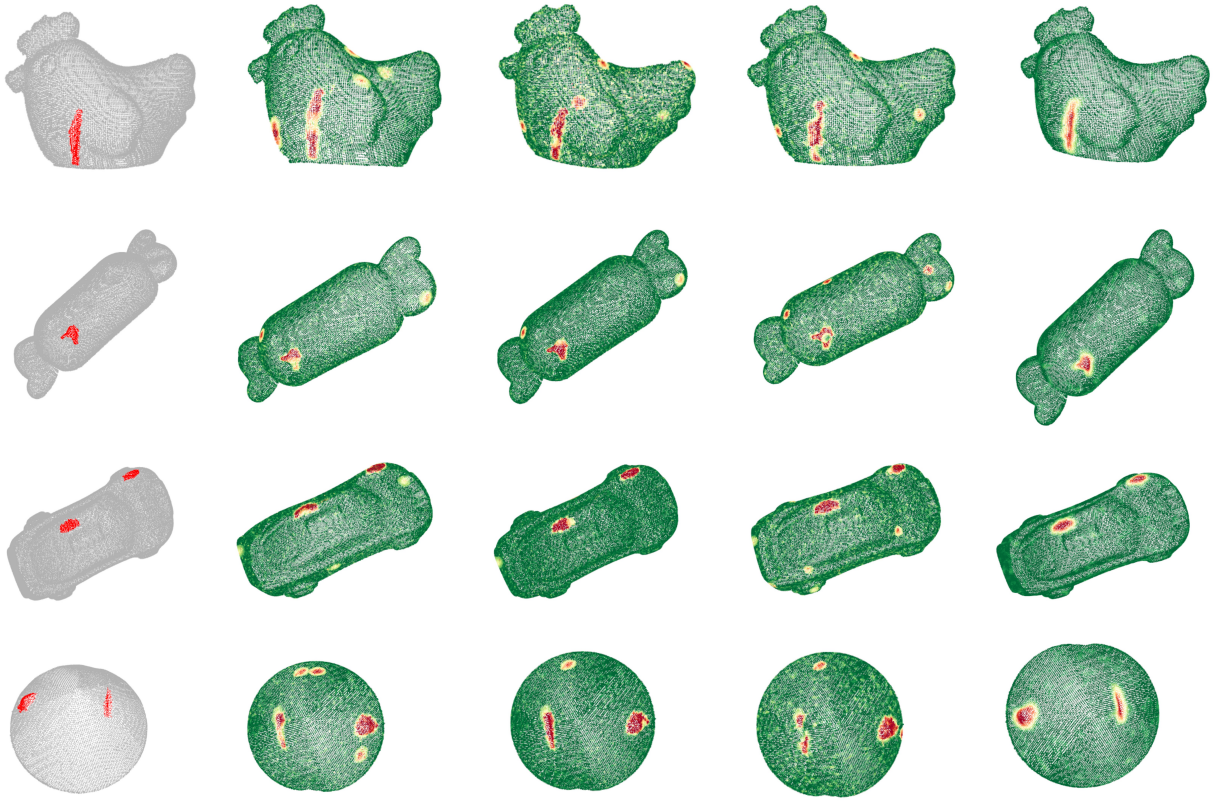

Figure 4: Qualitative results on Real3D-AD dataset. From left to right: input point clouds, Ground truth annotations of anomalous points in red, and anomaly scores for each 3D point predicted by the proposed method.

Table 2 and Figure 4 report performance on Real3D-AD, where scans are high density and defects arise under realistic sensor conditions. We compare with the same set of state-of-the-art methods used for Anomaly-ShapeNet. Our method achieves a point-level AUROC of 76.6% and point-level AUPR of 19.7%, improving over Group3AD by 2.8% and 5.8%, respectively. At the object level we obtain an AUROC of 78.4% and an AUPR of 77.7%, improving over Group3AD by 3.1% and 3.4%. As on the synthetic benchmark, our approach maintains a large throughput advantage with 17.31 FPS, which supports practical deployment in industrial inspection pipelines. Real3D-AD differs from synthetic benchmarks in three ways that highlight the strengths of our design. First, defects in real scans are often subtle and must be discriminated from sensor noise and varying point density. Mamba fusion aggregates multi-layer signals so that local irregularities are evaluated with respect to global part geometry, which reduces false positives caused by sensor artifacts. Second, GSAS ensures that the fusion operates on a surface-aware ordering, which is

Table 2
Average Results (%) on Real3D-AD dataset.

	Point Level (%)		Object Level (%)		Speed
	AUROC	AUPR	AUROC	AUPR	FPS
BTF(Raw)	57.3	2.4	60.5	61.3	2.05
BTF(FPFH)	73.2	6.6	63.7	61.6	1.01
M3DM(PointMAE)	63.9	4.9	55.4	57.5	0.31
M3DM(PointBERT)	63.9	5.4	54.0	58.3	0.52
PatchCore(FPFH)	57.9	7.3	59.6	59.3	0.10
PatchCore(FPFH+Raw)	68.2	12.5	68.4	66.9	0.10
PatchCore(PointMAE)	64.5	6.0	59.6	63.6	0.12
Reg3D-AD [34]	70.7	11.1	70.7	72.6	0.10
3D-ST [33]	70.7	11.1	64.8	72.6	1.52
Group3AD [35]	73.8	13.9	75.3	74.3	2.55
IMRNet [36]	72.8	16.8	72.8	62.8	5.62
R3D-AD [37]	59.4	4.3	73.6	63.5	7.15
Ours	76.6	19.7	78.4	77.7	17.31

important in dense, complex scans to prevent unrelated surface regions from contaminating contextual cues. Third, the selective anomalous feature generator produces spatially sparse supervisory signals that encourage sensitivity to small but consistent deviations, which is useful when prototype sets are limited and supervised defect examples are scarce.

Qualitatively, our method reliably highlights dents, missing material, and other geometric defects with high contrast while keeping false detections low in benign regions. Limitations include reduced sensitivity to appearance-only anomalies that do not affect geometry and occasional loss of localization precision in heavily occluded or extremely unevenly sampled areas. Practical mitigations include fusing appearance or reflectance channels when available and applying local density normalization during preprocessing. Results on Real3D-AD corroborate the conclusions from the synthetic benchmark. The combination of geometry-preserving ordering, linear-time cross-layer fusion, and sparse feature-space supervision improves both detection and localization robustness while offering a favorable runtime and memory profile for industrial applications.

3.6. Ablation Study

4. Conclusion

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.