POINT CLOUDS AS VISUAL TOKENS FOR PRETRAINED MULTIMODAL LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) for 3D scene understanding typically rely on multi-view images processed by pretrained visual encoders or point clouds processed by separate 3D networks. However, the former incurs substantial computation while sacrificing explicit geometric information, whereas the latter processes geometry directly but underutilizes the rich visual priors learned from large-scale image–text pretraining. To bridge this gap, we propose ReTok3D which enables direct point-cloud understanding through the pretrained visual pathway of an MLLM. ReTok3D converts volumetric point-cloud patches into a bounded set of visual tokens with minimal point-side processing, while incorporating 3D rotary position embeddings (3D-RoPE) and *spatially constrained spectral ordering* to preserve spatial structure and organize contextualized region features for language decoding. A two-stage training strategy further aligns point-derived features with RGB-D teacher representations and grounds visual attention to annotated referents. ReTok3D requires only 3.6–4.0M parameters for its 3D front end while achieving competitive performance on 3D question answering and dense captioning.

1 INTRODUCTION

Understanding a 3D scene through language requires identifying objects and relating them to the surrounding geometry. Point clouds measure geometry directly, whereas multimodal large language models (MLLMs) such as Qwen3-VL (Bai et al., 2025a) accept images and video through a pretrained visual pathway rather than a native point-cloud interface. Connecting point clouds to this pathway offers a way to combine explicit geometry with pretrained visual representations (Zheng et al., 2025; Wu et al., 2026a). However, their input structures differ significantly: images are 2D grid-based projections, whereas point clouds are unordered samples in metric space.

Existing 3D MLLMs typically follow two interface paradigms (Figure 1a,b). The first applies pretrained 2D encoders to captured or rendered multi-view images (Hong et al., 2023; Zhu et al., 2025; Zheng et al., 2025). These interfaces inherit strong visual representations, but their scene coverage depends on the selected views: overlapping observations repeat computation, surfaces absent from all views contribute no visual evidence, and processing cost grows with image resolution and view count. The second maps point clouds to language through dedicated 3D encoders (Chen et al., 2024a; Deng et al., 2025; Mei et al., 2025; Tang et al., 2026b) or lightweight tokenizers (Mei et al., 2026a; Tang et al., 2026a). These interfaces provide direct access to scene geometry, but construct point-derived features outside the pretrained visual pathway. They do not directly reuse an MLLM's visual encoder, leaving their joint image–text pretraining underexploited for point-cloud understanding.

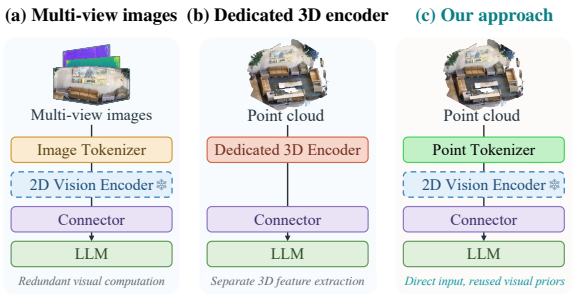


Figure 1: Three interfaces for 3D MLLMs. (a) Multi-view images use a pretrained visual pathway. (b) A dedicated 3D encoder maps points to language features. *(c) Our approach* maps points to visual tokens and reuses the pretrained visual encoder and connector.

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

We investigate a third interface paradigm: mapping point clouds to visual tokens and processing them through the pretrained 2D visual encoder and connector (Figure 1c). Point-token adaptation for frozen visual backbones demonstrates the feasibility of reusing pretrained visual features (Mei et al., 2026b). Extending this strategy to a complete MLLM also requires compatibility with the visual-to-language connector and the causal language decoder. This raises two questions. First, *how can a variable-sized point cloud fit a bounded visual-token layout without conflating layout indices with physical coordinates?* Second, *in what sequence should scene regions be presented to the language model?* Grouping determines each scene's region count before contextualized visual features are available. Ordering can subsequently exploit relationships revealed by the encoder without changing region membership or the token budget. Grouping and ordering are therefore complementary decisions made at different stages of the visual-to-language pathway.

Our method, ***ReTok3D*** (**Re**using pretrained models through **Tok**enization for **3D** understanding), answers the first question with a geometry-aware tokenizer. It groups cubic patches along a Hilbert curve, retains patches near each group's reference center, and pools their patch descriptors into a bounded set of regions. One learned linear layer maps each pooled descriptor to visual tokens. Centroid-derived positional signals locate these regions within the scene and provide relative positional information to visual attention. Point-side processing stays minimal, and the pretrained visual encoder and connector perform contextualization. It addresses the second question with *spatially constrained spectral ordering*: a spatial neighbor graph weighted by contextualized visual similarity yields a permutation of the region features, so the sequence seen by the decoder uses feature relationships unavailable to the initial geometric grouping, without changing region membership or the token budget. Reusing the visual pathway still requires aligning point-derived features with the representations expected by the pretrained model. Training therefore proceeds in two stages. *Cross-partition visual distillation* aligns region features with RGB-D teacher targets through shared point membership, so student regions and teacher superpoints may partition the scene differently. Grounded instruction tuning then combines next-token prediction with auxiliary *3D-grounded attention supervision*, which aligns the decoder's relative attention over regions with referent-fraction targets via the same point-to-region membership. Teacher features and referent labels are training-only; inference takes a point cloud and a text instruction, with a box prompt for dense captioning. With Qwen3-VL-8B and a 3.61M-parameter tokenizer, ReTok3D reaches 95.1 CIDEr on ScanQA, 60.3 exact match on SQA3D, and 88.0 CIDEr on ScanRefer captioning, the highest SQA3D and captioning scores among the compared systems. Our contributions are:

- We present a lightweight point-cloud interface that maps unordered 3D point clouds into a bounded set of visual tokens, enabling direct reuse of a pretrained MLLM's visual encoder and connector with centroid-derived 3D position signals distinct from the required token layout.
- To our knowledge, we are the first to use spectral ordering to sequence visual tokens for a language decoder, ordering contextualized 3D region features by joint geometric proximity and semantic feature similarity without altering region membership or feature dimensions.
- We combine cross-partition visual distillation with 3D-grounded attention supervision in a two-stage training strategy. These training-only objectives align point-derived features with RGB-D teacher targets and supervise relative visual attention without adding prediction heads or requiring teacher observations at inference.

2 Related work

Interfaces for 3D MLLMs. Multi-view interfaces represent scenes through rendered or captured images: 3D-LLM lifts multi-view features into a shared 3D representation (Hong et al., 2023), LLaVA-3D adds 3D positional embeddings to image patches and compresses tokens by location (Zhu et al., 2025), and Video-3D LLM treats observations as video with 3D positions and coverage-based view selection (Zheng et al., 2025); VLM-IE3D, SpatioLM, and Qwen-3D inject implicit or reconstructed geometry, depth and camera supervision, or 3D rotary positions into MLLMs (Li et al., 2026; Wu et al., 2026b; Lin et al., 2026). These designs make view selection and image-token compression part of the interface. Dedicated 3D encoders instead build scene representations with their own point-side networks: LL3DA connects scene features, visual prompts, and instructions through a query-based interface (Chen et al., 2024a); 3D-LLaVA pairs a sparse 3D encoder with an Omni Superpoint Transformer (Deng et al., 2025); PerLA aggregates local and global scene

representations (Mei et al., 2025); and NDTokenizer3D builds multi-scale Gaussian cells with a dedicated decoder (Tang et al., 2026b). Direct point-to-language interfaces such as Fase3D lower point-side cost with Fourier-based tokenization, curve serialization, and graph merging (Mei et al., 2026a). ReTok3D takes point clouds at inference but leaves contextualization and language mapping to the MLLM's pretrained visual encoder and connector, using RGB-D only for training targets.

Point tokens for pretrained visual backbones. Reusing a pretrained visual Transformer for points requires compatible token content and positions: EPCL feeds point-patch embeddings to a frozen CLIP Transformer (Huang et al., 2024), Any2Point assigns positional priors through virtual projection and adds adapters for local aggregation (Tang et al., 2024), and UTok3D learns a scale-normalized token interface for a frozen CLIP backbone with multi-view distillation and superpoint alignment (Mei et al., 2026b). Without reusing pretrained weights, Volt tokenizes volumetric patches with 3D rotary positions for a vanilla Transformer (Yilmaz et al., 2026); we adopt its cubic patches and bring 3D rotary positions into a pretrained encoder. These works target a visual encoder alone. ReTok3D extends pretrained-backbone reuse to a complete MLLM: point-derived tokens pass through the pretrained connector and are ordered for an autoregressive decoder, and Stage 1 adapts the superpoint alignment of UTok3D to the resulting region features.

Token ordering and grounding supervision. Curve serialization in PerLA and Fase3D orders points geometrically before or during feature construction (Mei et al., 2025; 2026a); spectral methods recover sequences from pairwise affinities (Recanati et al., 2018), and recursive spectral partitioning structures point tokens for autoregressive generation (Sun et al., 2026). To our knowledge, spectral ordering has not been used to order visual tokens for a language decoder. We apply it after visual contextualization, on a spatially constrained graph weighted by contextualized feature similarity, so the decoder's sequence reflects semantic relationships unavailable to geometric serialization. For grounding, Grounded 3D-LLM emits referent tokens aligned with scene phrases (Chen et al., 2024b), whereas Direct Visual Grounding supervises the language model's attention over image tokens with annotation-derived distributions (Esmaeilkhani & Latecki, 2026). We instead derive region-level attention targets from referents via point-to-region membership.

3 RETOK3D

ReTok3D adapts a pretrained MLLM to point clouds by reusing its visual feature extraction and language generation pathways. We describe the host through three components: a visual encoder $\mathcal{V}_{\phi_V}$, a visual-to-language connector $\mathcal{C}_{\phi_C}$, and an autoregressive language decoder $\mathcal{T}_{\phi_T}$, with pretrained parameters ϕ_V, ϕ_C, and ϕ_T, together with the adaptation modules specified in Section A.1. We suppress adapter parameters in this notation. The interface requires access to the visual encoder's input embeddings and positional mechanism, a connector that preserves region correspondence, and decoder attention to visual tokens for grounding supervision. Host-specific feature widths, token layouts, and adaptation modules are implementation choices (Section A.1).

We replace image-patch embedding with a geometry-aware 3D tokenizer $\mathcal{E}_{\theta_{3D}}$ with parameters θ_{3D} (Figure 2). It groups and filters point-cloud patches into a bounded set of regions and projects their descriptors into visual embeddings with centroid-based 3D positions. The host's visual encoder and connector then produce one contextualized feature per region. Because 3D regions have no intrinsic sequence order, spectral ordering Π_{spec} arranges these features into the sequence read by the language decoder, using both their geometry and feature similarity. Omitting the text instruction, which enters through the model's text tokenizer, and carrying region centroids and membership alongside the features, the composite model $\mathcal{F}_{3D}$ for a point cloud $\mathcal{P}$ is

$$\mathcal{F}_{3D}(\mathcal{P}) = \mathcal{T}_{\phi_T} \circ \Pi_{\text{spec}} \circ \mathcal{C}_{\phi_C} \circ \mathcal{V}_{\phi_V} \circ \mathcal{E}_{\theta_{3D}}(\mathcal{P}). \tag{1}$$

Training first distills a frozen RGB-D teacher, then applies grounded instruction tuning (Section 3.4).

3.1 GEOMETRY-AWARE VISUAL TOKENIZATION

Let $\mathcal{P} = \{(\mathbf{x}_i, \mathbf{f}_i)\}_{i=1}^{N}$ be a cloud of N points with metric coordinates $\mathbf{x}_i \in \mathbb{R}^3$ and c-dimensional attributes $\mathbf{f}_i \in \mathbb{R}^c$, such as color and normals. To avoid tying sequence length to scan density, the tokenizer pools fixed-width local descriptors into a bounded set of regions before projection.

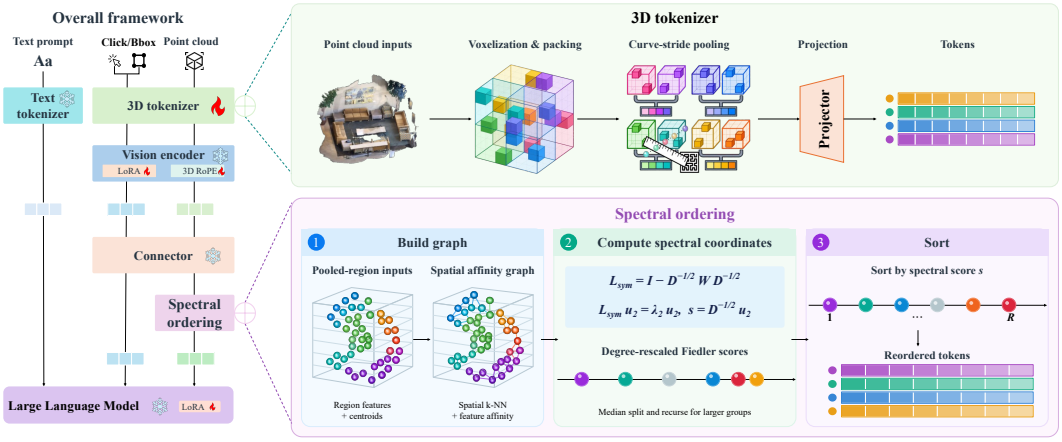


Figure 2: **Overview of ReTok3D.** Left: point-cloud-to-language pathway, where contextualized region features are spectrally reordered before language decoding. Top right: geometry-aware 3D tokenization through voxelization, cubic patches, curve-stride pooling, and projection. Bottom right: spatial-semantic graph construction and recursive spectral ordering based on region geometry and contextualized visual similarity. Click/Bbox denotes the location prompt in the text instruction. Flames and snowflakes indicate trainable and frozen components, respectively.

Voxels and cubic patches. Following Volt (Yilmaz et al., 2026), we average point attributes within voxels of side length δ and pack non-overlapping blocks with K voxels along each axis into cubic patches of side length $K\delta$. Each of the N_1 occupied patches, indexed by q, has a descriptor $\mathbf{u}_q \in \mathbb{R}^{d_u}$ of width $d_u = cK^3$ that concatenates its voxel attributes in fixed order, zero-filling empty voxels.

Curve-stride pooling. Since N_1 still varies across scenes, we pool consecutive patches along a space-filling curve into exactly R regions. Patches are stably sorted by the Hilbert indices of their patch coordinates. We choose the largest $R \le \min(N_1, R_{\max})$ satisfying the host's token-layout constraints, where $R_{\max}$ is the region budget. Any grid-fitting rule is specific to the chosen host (Section A.1). We index patches by their sorted rank $q \in \{0, \dots, N_1 - 1\}$. Rank q joins group $\mathcal{Q}_r$ with $r = 1 + \lfloor qR/N_1 \rfloor$, giving contiguous groups whose sizes differ by at most one. These groups partition all occupied patches, but consecutive patches can still be spatially distant. We therefore apply a distance mask within each group, retaining only patches within a radius τ_{jump} of the center $\mathbf{b}_r$ of its lower-middle patch. With patch centers $\mathbf{v}_q$, we define $\chi_{rq} = \mathbf{1}[\|\mathbf{v}_q - \mathbf{b}_r\|_2 \le \tau_{\text{jump}}]$, where $\mathbf{1}[\cdot]$ is one when its condition holds and zero otherwise, and pool the retained patches as $\bar{\mathbf{u}}_r = \big(\sum_{q\in\mathcal{Q}_r} \chi_{rq}\mathbf{u}_q\big)/\big(\sum_{q\in\mathcal{Q}_r} \chi_{rq}\big)$. The mean averages corresponding voxel slots of the retained patches with equal weight. The radius follows the scene's local spacing (Section A.2); the reference patch is always retained, so pooling yields exactly R nonempty regions, and setting all $\chi_{rq} = 1$ recovers coverage-preserving pooling. Retained points receive the region index $r(i) \in \{1, \dots, R\}$, while excluded, invalid, or padded points have $r(i) = 0$ and are excluded from region-centroid computation, student overlap counts, and grounding targets.

Projection and positions. We map each pooled descriptor through a scratch-initialized linear projection and arrange the outputs in the visual encoder's input format. Let $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_T]^\top \in \mathbb{R}^{T\times d_v}$ denote the resulting embeddings, where d_v is the encoder's input width and T is the visual-token count. Each embedding retains its source-region index $\rho(t) \in \{1, \dots, R\}$, keeping each token associated with its region independently of the input layout. The projection dimensions and packing are specified by the host interface (Section A.1). For region r, we compute the centroid $\mathbf{c}_r$ of its retained points and normalize it as $\tilde{\mathbf{c}}_r = (\mathbf{c}_r - \mathbf{o})/\max(\zeta, \epsilon)$ with $\mathbf{o}$ and ζ the midpoint and longest half-extent of the valid input bounding box and $\epsilon > 0$ a minimum scale for numerical stability. We then add the corresponding centroid embedding to each visual token:

$$\mathbf{z}_t = \mathbf{h}_t + g(\tilde{\mathbf{c}}_{\rho(t)}), \qquad t = 1, \dots, T, \tag{2}$$

where the position MLP $g : \mathbb{R}^3 \to \mathbb{R}^{d_v}$ encodes scene-relative location. Stacking these tokens gives $\mathbf{Z} \in \mathbb{R}^{T\times d_v}$. It uses one linear projection plus positional embeddings (Section A.1).

3.2 PRETRAINED VISUAL ENCODING AND AGGREGATION

The projected tokens $\mathbf{Z}$ match the visual encoder's input width, so the pretrained encoder extracts contextual features from 3D regions. Their packing follows the host's aggregation interface and preserves region membership; any required grid is a computational layout rather than metric coordinates, although it sets Qwen2.5-VL's window-attention neighborhoods and language-side position indices (Section A.1). Positions come from centroids: the additive embedding in Equation (2) gives each centroid's location relative to the scene bounding box, and axial 3D rotary position embeddings (RoPE) rotate visual queries and keys by scaled metric centroid coordinates, so their dot products depend on the displacement between region centroids. All tokens of a region share these coordinates, so RoPE encodes positions between regions, not their internal shape. The pretrained connector then aggregates the contextualized tokens into one feature $\mathbf{s}_r \in \mathbb{R}^{d_l}$ per region, where d_l is the decoder's embedding width. Each feature stays tied to its region's centroid and point membership, which distillation, ordering, and attention supervision rely on.

3.3 SPECTRAL ORDERING FOR LANGUAGE DECODING

The tokenizer's curve order depends only on geometry. Once the visual encoder has contextualized the regions, their feature similarity provides an additional cue for organizing the language-facing sequence. *Spatially constrained spectral ordering* operates on region features and centroids. It is a permutation of the R region features: applied after the connector, it leaves the token groups needed by the visual encoder intact and changes neither region membership nor feature values.

Affinity and bisection. Each metric centroid selects its $k = \min(k_{\max}, R-1)$ nearest neighbors, excluding itself; $k_{\max}$ caps the neighborhood size. Let $\mathcal{N}_k(r)$ be this neighbor set for region r. This restricts feature-based edges to spatial neighbors, which may be distant in sparse areas. With cosine dissimilarity $d_{rr'} = 1 - \hat{\mathbf{s}}_r^\top \hat{\mathbf{s}}_{r'}$ of L2-normalized features, the directed affinity for $R > 1$ is

$$A_{rr'} = \exp\left(-\frac{d_{rr'} - d_{\min}}{\tau}\right) \exp\left(-\frac{\|\mathbf{c}_r - \mathbf{c}_{r'}\|_2^2}{\sigma_r \sigma_{r'}}\right), \quad r' \in \mathcal{N}_k(r), \tag{3}$$

where σ_r is the k-th-neighbor distance and $d_{\min}$ and τ are the minimum and median absolute deviation of edge dissimilarities, both estimated per scene. The feature factor favors compatible contextualized representations; the spatial factor downweights long edges relative to local spacing. Edge weights are floored at $\epsilon_W > 0$, non-neighbors are zero, and $\mathbf{W} = (\mathbf{A} + \mathbf{A}^\top)/2$, where $\mathbf{A}, \mathbf{W} \in \mathbb{R}^{R \times R}$ are the directed and symmetric affinity matrices. To turn the graph into a sequence, we use its symmetric normalized Laplacian and Fiedler vector:

$$\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}, \qquad \mathbf{L}_{\text{sym}}\mathbf{u}_2 = \lambda_2 \mathbf{u}_2, \qquad s = \mathbf{D}^{-1/2}\mathbf{u}_2, \tag{4}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{D} = \operatorname{diag}(\sum_{r'} W_{rr'})$ is the degree matrix, $\mathbf{u}_2$ is the eigenvector of the second-smallest eigenvalue λ_2, and the degree-rescaled Fiedler scores $s \in \mathbb{R}^R$ form a one-dimensional coordinate that varies smoothly over strongly connected regions. Within strong-edge components, we sort regions by these scores, split at the median, and recurse in each half with recomputed scores, until pieces reach the leaf size. Concatenating the leaves gives π_{spec}, mapping each new sequence slot j to its original region index $\pi_{\text{spec}}(j)$ (Section A.3).

Applying the permutation. The permutation is computed from detached features and centroids and then applied jointly to the main and any auxiliary region-feature streams, grounding targets, and region-associated positional metadata supplied to the decoder. Every sequence slot therefore retains its region identity, and gradients flow through the gathered features. Text positions and causal indices stay fixed. The decoder's causal mask follows sequence order, so reordering changes which regions each region can attend to. Stage 2 and generation apply the ordering; Stage 1 does not.

3.4 TRAINING OBJECTIVES AND INFERENCE

Training first aligns point-derived features with the host's pretrained visual representation, then optimizes language responses with 3D-grounded attention supervision (Esmaeilkhani & Latecki, 2026). The choice of trainable host modules is specified separately in Section A.1.

Stage 1: cross-partition visual distillation. Visual alignment gives the scratch-initialized tokenizer a representation target in the host's visual feature space before response fine-tuning. For each host,

a frozen copy of its pretrained image pathway serves as the teacher, providing targets in the same connector output space as the student. It encodes RGB frames, lifts the features into 3D with registered depth, and averages them within fixed geometric *superpoints* (Landrieu & Simonovsky, 2018), giving one cached target $\mathbf{t}_m \in \mathbb{R}^{d_l}$ per superpoint m. Superpoints are supervision units, whereas student regions are constructed to meet the token budget, so the two partitions overlap many-to-many and a one-to-one feature comparison would compare different content. Following Mei et al. (2026b), we align each superpoint with retained student points through its overlap with regions:

$$\bar{\mathbf{s}}_m = \sum_r \alpha_{mr} \mathbf{s}_r, \qquad \alpha_{mr} = \frac{n_{mr}}{\sum_{r'} n_{mr'}}, \tag{5}$$

where n_{mr} counts retained points in superpoint m with $r(i) = r$ and α_{mr} is the fraction assigned to region r. $\bar{\mathbf{s}}_m$ is exactly the mean of the region features inherited by the superpoint's retained points. Let $\mathcal{M}$ contain the supported scene–superpoint pairs in the microbatch, each with retained student points and a valid teacher target; the index m includes scene identity. The visual objective is

$$\mathcal{L}_{\text{vis}} = \mathcal{D}(\{(\bar{\mathbf{s}}_m, \mathbf{t}_m, w_m) : m \in \mathcal{M}\}), \tag{6}$$

where w_m weights superpoints by teacher observation count and $\mathcal{D}$ combines feature regression, cosine alignment, and channel-statistic matching over the supported microbatch (Section A.4). The token budget can vary independently of the teacher partition. We minimize $\mathcal{L}_{\text{vis}}$ over the tokenizer and selected visual adaptation parameters while keeping the teacher fixed. Adaptation modules are listed in Section A.1; this stage keeps tokenizer order and ends at the connector.

Stage 2: grounded instruction tuning. We continue training the tokenizer and visual adaptation learned in Stage 1, and add language low-rank adaptation (LoRA) (Hu et al., 2021), using teacher-forced answers and descriptions. Annotated referents additionally supervise where the decoder attends. For an example b with R_b valid regions and at least one retained referent point, let β_{br} be the fraction of retained points in region r that lie on referents. With smoothing η, the target $\mathbf{p}_b$ has entries $p_{br} = (1-\eta)\beta_{br} / \sum_{r'} \beta_{br'} + \eta / R_b$, which favors regions with higher referent occupancy. Let $\mathbf{a}_b^{(\ell)}$ be the attention over regions at supervised layer ℓ, at the query that predicts the last supervised non-special token of the final assistant turn, normalized over visual keys separately for each head and then averaged over heads (Section A.5). Over L layers, the grounded-attention loss (GAL) is

$$\mathcal{L}_{\text{GAL}} = \frac{1}{BL} \sum_{b \in \mathcal{B}_+} \sum_{\ell=1}^{L} D_{\text{KL}}\big(\mathbf{p}_b \,\|\, \mathbf{a}_b^{(\ell)}\big), \tag{7}$$

where B is the microbatch size and $\mathcal{B}_+$ contains examples with a retained referent point. This loss adds no parameters and supervises relative attention among visual regions; it does not constrain the total attention mass assigned to visual versus text tokens. Stage 2 minimizes $\mathcal{L}_{\text{SFT}} + \lambda_{\text{GAL}} \mathcal{L}_{\text{GAL}}$, where $\mathcal{L}_{\text{SFT}}$ is the supervised fine-tuning (SFT) cross-entropy over assistant tokens. Other pretrained weights stay frozen but pass gradients to the tokenizer (Section A.1 and Section B).

Inference. At inference, the tokenizer maps points to visual embeddings, the reused visual pathway produces region features, and spectral ordering arranges them for autoregressive generation conditioned on the text instruction (Equation (1)). For dense captioning, a location prompt (Click/Bbox in Figure 2) specifies the target: its 3D box, normalized to the scene bounds, enters the instruction, from annotations in training and Mask3D proposals at evaluation.

4 Experiments

4.1 Experimental setup

Benchmarks and metrics. We evaluate question answering on ScanQA validation (Azuma et al., 2022) and SQA3D test (Ma et al., 2022), and dense captioning on ScanRefer validation (Chen et al., 2020). All three benchmarks use ScanNet indoor scenes (Dai et al., 2017). ScanQA tests scene-level understanding, while SQA3D additionally conditions each question on a textual situation. We report BLEU-4 (B-4), CIDEr (C), ROUGE-L (R), and METEOR (M) for ScanQA and ScanRefer, and exact match (EM@1) for SQA3D. ScanRefer evaluation uses Mask3D proposals (Schult et al., 2023) at an intersection-over-union (IoU) threshold of 0.5; protocols are detailed in Section C.

Table 1: 3D question answering on ScanQA validation and SQA3D test.

Method	Backbone	3D front end		ScanQA (val)				SQA3D (test)
		Params (M) ↓	FLOPs (G) ↓	B-4 ↑	C ↑	R ↑	M ↑	EM@1 ↑
Dedicated 3D encoders								
LL3DA (Chen et al., 2024a)	OPT-1.3B	118.9	40.2	13.5	76.8	37.3	15.9	–
PerLA (Mei et al., 2025)		119.8	163.4	14.5	78.1	39.6	17.4	–
3D-LLaVA (Deng et al., 2025)	Vicuna-1.5-7B	58.3	37.8	<u>17.1</u>	92.6	43.1	18.4	54.5
NDTokenizer3D (Tang et al., 2026b)		101.3	–	17.0	**98.6**	**44.9**	**19.4**	54.4
Direct point-to-language interfaces								
Fase3D (Mei et al., 2026a)	Qwen2.5-3B	10.5	**2.0**	**17.1**	90.1	42.6	18.2	53.9
Fase3D (Mei et al., 2026a)	Vicuna-1.5-7B	12.1	<u>2.1</u>	16.9	91.7	43.4	18.6	54.3
Pretrained visual-pathway reuse								
ReTok3D	Qwen2.5-VL-3B	<u>4.0</u>	16.4	16.1	93.2	44.1	18.6	<u>55.9</u>
ReTok3D	Qwen3-VL-8B	**3.6**	14.8	15.3	<u>95.1</u>	<u>44.7</u>	<u>18.9</u>	**60.3**

Model and training. We use Qwen2.5-VL-3B (Bai et al., 2025b) and Qwen3-VL-8B (Bai et al., 2025a) as the pretrained MLLM backbones. Both configurations take point coordinates, RGB, and normals as input and reuse their respective host's visual encoder and connector. RGB-D observations provide offline Stage 1 supervision; model inference uses points and text, with a proposal-box prompt for dense captioning. The tokenizer uses 0.02 m voxels and $5 \times 5 \times 5$-voxel patches, with a default budget of at most $R_{\max} = 2{,}048$ language-facing regions. Each region produces four visual-input tokens, so $T = 4R$; actual sequence lengths depend on occupied patches and grid fitting. Stage 1 distillation precedes grounded instruction tuning (Sections A.1 and 3.4 and Section B).

Baselines and reporting scope. We compare with dedicated-encoder methods LL3DA (Chen et al., 2024a), PerLA (Mei et al., 2025), 3D-LLaVA (Deng et al., 2025), and NDTokenizer3D (Tang et al., 2026b), and the lightweight Fase3D interface (Mei et al., 2026a). The tables identify their backbones: OPT-1.3B (Zhang et al., 2022), Vicuna-1.5-7B (Chiang et al., 2023), and Qwen2.5-3B (Yang et al., 2024). NDTokenizer3D benchmark scores are taken from Tang et al. (2026b); the remaining baseline scores follow Mei et al. (2026a). For Fase3D captioning we include its Qwen2.5-3B variant with an external segmenter, matching the proposal setting of the table. These systems differ in backbone, pretraining, adaptation, and training data, so the comparisons provide system-level context rather than isolate the tokenizer's contribution. Each ablation holds its backbone fixed (Section 4.4).

Parameter counts refer to the 3D front end; ours include the tokenizer projection, position MLP, and slot embeddings, excluding the pretrained host and its adaptation parameters. Our FLOPs are analytical matrix-multiply upper bounds for the projection and position MLP at $R_{\max} = 2{,}048$; actual counts depend on each scene's R (Section A.1). They exclude geometric preprocessing, spectral ordering, and the host. Published baseline costs may use different component boundaries and input sizes; these figures do not measure total-model efficiency or end-to-end speed. Dashes denote unavailable values. Benchmark scores are rounded to one decimal place. Bold and underlined values mark the best and second-best results before rounding, without implying statistical significance. Differences and ratios use unrounded values; arrows mark whether higher or lower is better.

4.2 Scene-level question answering

Table 1 reports scene-level question answering with both Qwen2.5-VL-3B and Qwen3-VL-8B. On ScanQA, the Qwen2.5-VL-3B configuration achieves 93.2 CIDEr, 44.1 ROUGE-L, and 18.6 METEOR, exceeding Fase3D with Qwen2.5-3B on these three metrics, although its BLEU-4 is lower (16.1 versus 17.1). With Qwen3-VL-8B, ReTok3D reaches 95.1 CIDEr, 44.7 ROUGE-L, and 18.9 METEOR, ranking second among the listed systems on all three metrics behind NDTokenizer3D. Compared with our Qwen2.5-VL-3B configuration, Qwen3-VL-8B scores higher on CIDEr, ROUGE-L, and METEOR but lower on BLEU-4 (15.3 versus 16.1), so neither configuration leads across all ScanQA metrics. On SQA3D, ReTok3D achieves 55.9 EM@1 with Qwen2.5-VL-3B and 60.3 with Qwen3-VL-8B, the second-highest and highest listed scores, respectively. Both exceed 3D-LLaVA

Table 2: Dense captioning on ScanRefer validation with Mask3D proposals at IoU 0.5.

Method	Backbone	3D front end		ScanRefer @0.5			
		Params (M) ↓	FLOPs (G) ↓	B-4 ↑	C ↑	R ↑	M ↑
Dedicated 3D encoders							
LL3DA (Chen et al., 2024a)	OPT-1.3B	118.9	80.4	36.8	65.2	55.1	26.0
PerLA (Mei et al., 2025)		119.8	326.8	38.0	69.4	56.8	29.1
3D-LLaVA (Deng et al., 2025)	Vicuna-1.5-7B	58.3	37.8	36.9	78.8	57.7	27.1
NDTokenizer3D (Tang et al., 2026b)		101.3	–	36.7	79.0	57.7	27.1
Direct point-to-language interfaces							
Fase3D (Mei et al., 2026a)	Qwen2.5-3B	10.5	**2.0**	41.3	78.1	57.6	27.9
Pretrained visual-pathway reuse							
ReTok3D	Qwen2.5-VL-3B	4.0	16.4	44.9	81.4	65.5	30.8
ReTok3D	Qwen3-VL-8B	**3.6**	14.8	**45.8**	**88.0**	**66.0**	**31.1**

(54.5), NDTokenizer3D (54.4), and the two Fase3D configurations (53.9 and 54.3). These results support using the point-to-visual interface with both pretrained hosts for questions conditioned on a textual situation, as well as scene-level questions in ScanQA. As discussed in Section 4.1, comparisons across complete systems do not isolate the effect of the interface from backbone and training differences. Both configurations use compact 3D front ends: Qwen2.5-VL-3B adds 4.0M parameters, while Qwen3-VL-8B adds 3.6M parameters. These are the two lowest front-end parameter counts in Table 1, compared with 58.3–119.8M for dedicated encoders and 10.5–12.1M for Fase3D. At the region cap, projection plus position MLP cost 16.4G (Qwen2.5-VL-3B) and 14.8G (Qwen3-VL-8B) FLOPs; smaller region counts reduce these costs linearly. Qwen3-VL-8B's smaller front end reflects its visual width; these costs exclude the reused pretrained modules.

4.3 Dense captioning

ScanRefer scores generated object descriptions after proposal matching at IoU 0.5, so results reflect both proposal association and description quality. As shown in Table 2, ReTok3D with Qwen2.5-VL-3B achieves 44.9 BLEU-4, 81.4 CIDEr, 65.5 ROUGE-L, and 30.8 METEOR, ranking second among the listed systems on all four metrics and exceeding all listed dedicated-encoder and Fase3D baselines. With Qwen3-VL-8B, ReTok3D reaches 45.8 BLEU-4, 88.0 CIDEr, 66.0 ROUGE-L, and 31.1 METEOR, the highest listed values on all four metrics. Its CIDEr margin is 9.0 points over NDTokenizer3D and 9.9 points over Fase3D with Qwen2.5-3B. Captioning reuses the same front ends as question answering. Both configurations outperform the listed Fase3D variant on all four metrics with fewer front-end parameters (3.6M and 4.0M versus 10.5M). Under the stated accounting boundaries, dedicated encoders use 16.1–33.1× as many parameters as our 8B front end; differences in FLOP accounting preclude runtime comparison (Section 4.1).

4.4 Ablation studies

We ablate attention supervision and spectral ordering (Table 3), the region budget (Table 4), and Stage 1 distillation (Table 5) with Qwen3-VL-8B; Qwen2.5-VL-3B results are in Appendix D.1.

Table 3: Module ablations on ScanQA and ScanRefer validation and SQA3D test with Qwen3-VL-8B. Bold marks the best value in each column.

Variant	ScanQA				ScanRefer @0.5				SQA3D
	B-4 ↑	C ↑	R ↑	M ↑	B-4 ↑	C ↑	R ↑	M ↑	EM@1 ↑
Baseline	14.6	91.5	43.7	18.4	44.4	82.1	65.3	30.5	56.1
+ Grounded attention	15.2	91.2	43.8	17.9	45.2	**88.1**	64.9	30.9	59.5
+ Spectral ordering	13.6	89.4	43.1	17.6	**45.8**	83.8	**66.1**	30.8	57.5
Full (ReTok3D)	**15.3**	**95.1**	**44.7**	**18.9**	**45.8**	88.0	66.0	**31.1**	**60.3**

Attention supervision and spectral ordering. In Table 3, the full variant improves all nine metrics over the baseline and is best or tied best on seven. ScanQA CIDEr rises from 91.5 to 95.1, ScanRefer

CIDEr from 82.1 to 88.0, and SQA3D EM@1 from 56.1 to 60.3. The individual modules have task-dependent effects: spectral ordering alone improves ScanRefer and SQA3D but lowers all four ScanQA metrics. Grounded attention alone gives the highest ScanRefer CIDEr (88.1), while spectral ordering gives the highest ScanRefer ROUGE-L (66.1). The combined model improves all three tasks over the baseline, although run-to-run variability is not quantified.

Table 4: Region-budget ablation on ScanQA and ScanRefer validation and SQA3D test with Qwen3-VL-8B. We vary the region cap $R_{\max}$; each scene uses $R \leq R_{\max}$ regions and $T = 4R$ tokens.

	ScanQA				ScanRefer @0.5				SQA3D
$R_{\max}$	B-4 ↑	C ↑	R ↑	M ↑	B-4 ↑	C ↑	R ↑	M ↑	EM@1 ↑
512	14.2	90.4	43.5	17.9	44.9	79.9	64.9	30.3	54.7
1,024	**15.9**	<u>91.7</u>	<u>44.0</u>	<u>18.2</u>	<u>45.5</u>	<u>83.1</u>	<u>65.4</u>	<u>30.6</u>	<u>55.6</u>
2,048	<u>15.4</u>	**93.4**	**44.5**	**18.4**	**45.8**	**88.0**	**66.0**	**31.1**	**60.3**

Region budget. Table 4 varies $R_{\max}$ from 512 to 1,024 and 2,048 language-facing regions. The actual count $R \leq R_{\max}$ varies across scenes, with $T = 4R$ visual-input tokens. Every ScanRefer and SQA3D metric increases with each doubling: captioning CIDEr rises from 79.9 to 83.1 and 88.0, and SQA3D EM@1 from 54.7 to 55.6 and 60.3. ScanQA CIDEr, ROUGE-L, and METEOR also increase, with CIDEr rising from 90.4 to 91.7 and 93.4, whereas ScanQA BLEU-4 peaks at 15.9 with a 1,024-region cap. On all three benchmarks, the second doubling gives the larger CIDEr or EM@1 gain. The default 2,048-region cap has the highest reported value on every metric except ScanQA BLEU-4. Larger budgets allow more, finer regions and longer visual and language-prefix sequences; the best accuracy–cost trade-off requires matched runtime measurements.

Table 5: Stage 1 ablation with Qwen3-VL-8B at 20k Stage 2 updates.

	ScanQA				ScanRefer @0.5				SQA3D
Training	B-4 ↑	C ↑	R ↑	M ↑	B-4 ↑	C ↑	R ↑	M ↑	EM@1 ↑
Without Stage 1	14.1	80.2	39.7	16.1	43.1	70.1	64.8	29.7	52.8
With Stage 1	**15.3**	**95.1**	**44.7**	**18.9**	**45.8**	**88.0**	**66.0**	**31.1**	**60.3**

Stage 1 visual distillation. Table 5 compares two-stage training against direct SFT on Qwen3-VL-8B. Direct SFT starts from a scratch-initialized tokenizer and the pretrained host, whereas two-stage training restores the tokenizer and visual adaptation learned by distillation. The comparison fixes Stage 2 data, adaptation modules, the grounding loss, spectral ordering, the cap of 2,048 regions, and 20k Stage 2 updates. Stage 1 initialization improves all nine metrics. CIDEr rises from 80.2 to 95.1 on ScanQA and from 70.1 to 88.0 on ScanRefer, BLEU-4 rises from 14.1 to 15.3 and from 43.1 to 45.8, and SQA3D EM@1 rises from 52.8 to 60.3. The variants match Stage 2 updates but not total compute: two-stage training adds 5,000 distillation updates in the reference schedule (Table 6), so the table shows the benefit of distillation-based initialization at a fixed Stage 2 schedule.

5 Conclusion

We show that a pretrained MLLM's visual encoder and connector can support 3D scene understanding through a compact point-cloud interface. ReTok3D maps irregular geometry to visual tokens with a linear projection and centroid-based positions, then orders contextualized regions using geometric proximity and semantic similarity. Cross-partition visual distillation and 3D-grounded attention supervision align this interface during training, without requiring teacher observations at inference. Across ScanQA, SQA3D, and ScanRefer, experiments with Qwen2.5-VL-3B and Qwen3-VL-8B demonstrate competitive question answering and dense captioning with a 3.6–4.0M-parameter tokenizer added to the pretrained host. Ablations support the benefit of visual distillation and the combined use of grounded attention and spectral ordering. These findings highlight the pretrained visual pathway as a reusable foundation for 3D MLLMs, with a lightweight interface connecting explicit geometry to learned visual representations.

AI USE STATEMENT

Generative AI assisted with drafting, English editing, figure formatting, organization, and consistency checks across the manuscript, code, and experiment records.

ETHICS STATEMENT

This work uses existing indoor-scene benchmarks. Scene-language models can inherit biases from their training data and generate inaccurate descriptions. Indoor reconstructions may also expose information about private spaces; applications should protect the privacy of occupants.

REPRODUCIBILITY STATEMENT

Section 3 defines the tokenizer, spectral ordering, and training objectives. Section A gives numerical and architectural details, and Sections B and C describe data preparation, optimization settings, and evaluation protocols. Sections 4 and D state the comparison scope.

REFERENCES

Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes. In *European conference on computer vision*, pp. 422–440. Springer, 2020.

Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19107–19117. IEEE, 2022.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. Scanrefer: 3d object localization in rgb-d scans using natural language. In *European conference on computer vision*, pp. 202–221. Springer, 2020.

Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding, reasoning, and planning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26418–26428. IEEE, 2024a.

Yilun Chen, Shuai Yang, Haifeng Huang, Tai Wang, Runsen Xu, Ruiyuan Lyu, Dahua Lin, and Jiangmiao Pang. Grounded 3d-llm with referent tokens. *arXiv preprint arXiv:2405.10370*, 2024b.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 2432–2443. IEEE, 2017.

Jiajun Deng, Tianyu He, Li Jiang, Tianyu Wang, Feras Dayoub, and Ian Reid. 3d-llava: Towards generalist 3d lmms with omni superpoint transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3772–3782. IEEE, 2025.

Parsa Esmaeilkhani and Longin Jan Latecki. Direct visual grounding by directing attention of visual tokens. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5787–5797. IEEE, 2026.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Xiaoshui Huang, Zhou Huang, Sheng Li, Wentao Qu, Tong He, Yuenan Hou, Yifan Zuo, and Wanli Ouyang. Frozen clip transformer is an efficient point cloud encoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 2382–2390, 2024.

Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4558–4567, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Landrieu_Large-Scale_Point_Cloud_CVPR_2018_paper.html.

Wenhao Li, Xueying Jiang, Quanhao Qian, Deli Zhao, Ran Xu, Shijian Lu, and Gongjie Zhang. 3d-aware vlms with implicit and explicit geometries. *arXiv preprint arXiv:2607.21595*, 2026.

Lucy Lin, Ayush Jain, Yifan Liu, and Katerina Fragkiadaki. Qwen-3d: A generalist 3d vision-language model for spatial understanding. *arXiv preprint arXiv:2608.02980*, 2026.

Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *arXiv preprint arXiv:2210.07474*, 2022.

Guofeng Mei, Wei Lin, Luigi Riz, Yujiao Wu, Fabio Poiesi, and Yiming Wang. Perla: Perceptive 3d language assistant. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14369–14379. IEEE, 2025.

Guofeng Mei, Wei Lin, Luigi Riz, Yujiao Wu, Yiming Wang, and Fabio Poiesi. Efficient encoder-free fourier-based 3d large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23785–23794, 2026a.

Guofeng Mei, Qinfeng Xiao, Bin Ren, Luigi Riz, Juan Liu, Xiaoshui Huang, Xu Zheng, Nicu Sebe, Ming-Hsuan Yang, and Fabio Poiesi. Parameter-efficient clip adaptation for 3d understanding via unified tokenization, 2026b. URL https://arxiv.org/abs/2505.18819.

Antoine Recanati, Thomas Kerdreux, and Alexandre d'Aspremont. Reconstructing latent orderings by spectral clustering. *arXiv preprint arXiv:1807.07122*, 2018.

Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8216–8223. IEEE, 2023.

Monan Sun, Bangzhen Liu, Huaidong Zhang, and Shengfeng He. Learning to tessellate: Point cloud generation via recursive spectral partitioning. *arXiv preprint arXiv:2608.02432*, 2026.

Yiwen Tang, Ray Zhang, Jiaming Liu, Zoey Guo, Bin Zhao, Zhigang Wang, Peng Gao, Hongsheng Li, Dong Wang, and Xuelong Li. Any2point: Empowering any-modality large models for efficient 3d understanding. In *European Conference on Computer Vision*, pp. 456–473. Springer, 2024.

Yiwen Tang, Ziyu Guo, Zhuhao Wang, Renrui Zhang, Qizhi Chen, Junli Liu, Delin Qu, Dong Wang, Bin Zhao, and Xuelong Li. Exploring the potential of encoder-free architectures in 3d lmms. In *International Conference on Learning Representations*, 2026a.

Yutao Tang, Cheng Zhao, Gaurav Mittal, Rohith Kukkala, Rama Chellappa, Cheng Peng, and Mei Chen. Scenes as tokens: Multi-scale normal distributions transform tokenizer for general 3d vision-language understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38584–38594, 2026b.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. *Advances in neural information processing systems*, 38: 13569–13597, 2026a.

Jing Wu, Jianhua Wu, Jiayi Guan, Jiahong Chen, Jinghui Lu, Hangjun Ye, Bingzhao Gao, and Long Chen. Spatiolm: Towards general physical spatial intelligence in vision-language models. *arXiv preprint arXiv:2608.01899*, 2026b.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Kadir Yilmaz, Adrian Kruse, Tristan Höfer, Daan de Geus, and Bastian Leibe. Volume transformer: Revisiting vanilla transformers for 3d scene understanding. *arXiv preprint arXiv:2604.19609*, 2026.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Duo Zheng, Shijia Huang, and Liwei Wang. Video-3d llm: Learning position-aware video representation for 3d scene understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8995–9006. IEEE, 2025.

Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. Llava-3d: A simple yet effective pathway to empowering lmms with 3d capabilities. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4295–4305. IEEE, 2025.

APPENDIX

A IMPLEMENTATION DETAILS

We detail host-layout constraints, numerical rules, and loss reductions for Section 3.

A.1 HOST-SPECIFIC INSTANTIATIONS

Our experiments instantiate the interface with Qwen2.5-VL-3B (Bai et al., 2025b) and Qwen3-VL-8B (Bai et al., 2025a). The reference dimensions and layout settings below describe Qwen3-VL-8B unless stated otherwise; the Qwen2.5-VL-3B parameter count is given separately. These settings instantiate the interface requirements in Section 3.

Projection and token packing. For the grouped-token interface used here, each pooled descriptor produces G visual embeddings through one linear projection: $\mathbf{G}_r = \mathrm{reshape}(\mathbf{W}_{\mathrm{tok}}\bar{\mathbf{u}}_r + \mathbf{b}_{\mathrm{tok}}, G, d_v)$, with $\mathbf{W}_{\mathrm{tok}} \in \mathbb{R}^{Gd_v \times d_u}$ and $\mathbf{b}_{\mathrm{tok}} \in \mathbb{R}^{Gd_v}$. Learned embeddings $\mathbf{e}_j \in \mathbb{R}^{d_v}$ distinguish the G positions in each group. In the notation of Equation (2), before layout packing, $t = (r-1)G + j$, $\rho(t) = r$, and $\mathbf{h}_t = \mathbf{G}_{rj} + \mathbf{e}_j$, where $\mathbf{G}_{rj}$ is the j-th row of $\mathbf{G}_r$. Thus the implemented token is $\mathbf{z}_t = \mathbf{G}_{rj} + \mathbf{e}_j + g(\tilde{\mathbf{c}}_r)$, and $T = RG$. Packing preserves these region associations. The learned embeddings act as position-specific biases; this parameterization retains the single linear projection of the pooled descriptor.

Centroid-based position signals. The MLP g in Equation (2) has widths $3 \to 128 \to d_v$ and uses the centroid normalization defined there, with minimum scale $\epsilon = 10^{-6}$ m. To instantiate the visual-attention positions in Section 3.2, we replace image-plane rotary positions with axial 3D RoPE. Every slot in region r shares the rotary coordinate $\mathbf{r}_r = (\mathbf{c}_r - \min_{r'} \mathbf{c}_{r'})/(K\delta)$, where the minimum is taken independently along each axis. Subtracting a shared origin preserves relative offsets, and division by the cubic patch side length $K\delta$ expresses them in patch-scale units. These coordinates set the rotations applied to visual queries and keys. Their positional contribution to attention depends on $\mathbf{r}_r - \mathbf{r}_{r'} = (\mathbf{c}_r - \mathbf{c}_{r'})/(K\delta)$. Regions with the same centroid receive identical rotary coordinates, regardless of differences in shape or extent. Axial RoPE uses base 10^4 and distributes each head's rotary frequency pairs as evenly as their count permits across the three axes. For Qwen3-VL we also omit its learned, interpolated 2D absolute position embeddings; Qwen2.5-VL has no pretrained absolute visual-position table. In both hosts, g supplies absolute 3D location and the learned slot embeddings distinguish positions within each region. The native grid keeps a region's G slots together for the connector to merge.

Grid fitting and token budget. For a region budget $R_{\max}$, grid fitting selects the largest $R \le \min(N_1, R_{\max})$ that factorizes as $R = hw$ with integers $h \le w \le 2h$, where h and w are the region-grid height and width; counts of one or two are also accepted. Fitting satisfies the host's layout constraints by reducing pooled groups rather than adding empty regions. Both hosts merge 2×2 visual-encoder inputs, so $G = 4$ inputs per region feature supplied to the language model. For Qwen2.5-VL the grid also defines window attention: 28 of its 32 visual blocks attend only within 4×4 tiles of the row-major grid of curve-ordered regions, and the remaining four attend over the whole scene. In the language decoder, region tokens initially receive the host's multimodal RoPE indices of this grid; spectral ordering permutes these region-associated indices with the features. Each region thus keeps grid indices that follow curve order rather than metric coordinates; reordering changes the causal order in which the decoder reads regions, not their rotary indices. The merge factor preserves each 3D region and its membership. The visual-input count is $T = GR$, bounded by $T_{\max} = GR_{\max}$, and R counts the region features supplied to the language model. The default budget is $R_{\max} = 2{,}048$ regions, not a fixed sequence length: R depends on the scene's occupied patches and grid fitting, and $T = 4R$ varies accordingly.

Tokenizer configuration. Voxels have side length $\delta = 0.02$ m and cubic patches have $K = 5$ voxels per axis. With $c = 6$ attributes (RGB and normals), a patch descriptor has $d_u = cK^3 = 750$ entries, with empty voxel slots zero-filled. The token projection is $\mathbf{W}_{\mathrm{tok}} \in \mathbb{R}^{Gd_v \times d_u}$ with $d_v = 1{,}152$, following the Qwen3-VL-8B configuration.[1] The rotary coordinate scale is $K\delta = 0.10$ m, the patch side length, and the minimum in the rotary-coordinate expression is taken per axis.

[1] https://huggingface.co/Qwen/Qwen3-VL-8B-Instruct/blob/main/config.json

Trainable parameters. The stage-wise adaptation in Section 3.4 includes all tokenizer parameters: projection, position MLP, and slot embeddings, together with visual query/key LoRA and pre-attention normalization; Stage 2 and direct SFT continue training all of them. Stage 2 language LoRA targets the query, key, value, output, gate, up, and down projections of the language decoder. The native mergers, language embeddings, output head, and all remaining pretrained parameters stay fixed. Direct SFT omits the Stage 1 initialization.

Parameter accounting. The tokenizer has 3,614,336 learned parameters (3.61M): 3,460,608 in the $750 \to 4 \times 1{,}152$ projection including its bias, 149,120 in the $3 \to 128 \to 1{,}152$ position MLP, and 4,608 in slot embeddings. The projection alone accounts for 3.46M. This analytical count excludes visual and language LoRA and the pretrained host. The projection and slot embeddings are proportional to d_v and the position MLP is affine in d_v, for a total of $3{,}137d_v + 512$ parameters. For the Qwen2.5-VL-3B backbone, whose vision tower has $d_v = 1{,}280$ (Bai et al., 2025b),[2] this gives 4,015,872 parameters (4.02M): 3,845,120 in the projection, 165,632 in the position MLP, and 5,120 in slot embeddings. The Qwen2.5-VL-3B front end is therefore larger than Qwen3-VL-8B's despite the smaller language model: tokenizer size tracks the visual hidden width d_v of the visual encoder, not the language-model size in the backbone's name.

FLOP accounting. For R regions, the projection and the two linear layers of the position MLP require $2R[d_u G d_v + 3 \times 128 + 128 d_v]$ matrix-multiply FLOPs, counting a multiplication and addition as two operations. At $R_{\max} = 2{,}048$, this gives upper bounds of 14.76G for Qwen3-VL-8B and 16.40G for Qwen2.5-VL-3B, rounded to 14.8G and 16.4G in the tables. Actual counts scale with each scene's R. This analytical accounting excludes biases, activations, embedding additions, geometric preprocessing, spectral ordering, and all host operations; it is not a runtime measurement.

A.2 Distance-to-center masking

The local-spacing rule adapts geometric candidate filtering from graph-based point grouping to the tokenizer's single Hilbert order. For ordered patch q, let $\mathcal{N}_{w_{\text{curve}}}(q)$ contain the other patches within w_{curve} curve ranks on either side. The radius is

$$d_q = \min_{q' \in \mathcal{N}_{w_{\text{curve}}}(q)} \|\mathbf{v}_q - \mathbf{v}_{q'}\|_2, \qquad \tau_{\text{jump}} = \gamma \max\Big(\text{med}_{\text{lo}}\{d_q\}_{q=0}^{N_1-1}, \epsilon_{\text{jump}}\Big). \tag{8}$$

Here d_q is the nearest-neighbor distance within the curve window, $\gamma > 0$ scales the radius, $\epsilon_{\text{jump}} > 0$ is the base spacing floor, and med_{lo} denotes the lower median, including for even counts. A single-patch scene retains its only patch. Reference settings are $w_{\text{curve}} = 8$, $\gamma = 3$, and $\epsilon_{\text{jump}} = 10^{-6}$ m. Distances use metric patch centers; the square root of the lower median of squared distances gives the same radius. The mask in Section 3.1 bounds pairwise distances between retained patch centers by $2\tau_{\text{jump}}$. Excluded patches have no fallback assignment. Scene-position normalization uses the full valid input bounding box, even though descriptors, centroids, student overlap counts, and grounding targets use retained points. Cached teacher targets are independent of this mask.

A.3 Spatially constrained spectral ordering

Graph. The graph in Equation (3) uses $k_{\max} = 8$. Neighbor distances are exact in FP64 and ties preserve region index. Region features are L2-normalized in FP32, with $\hat{\mathbf{s}}_r = \mathbf{s}_r / \max(\|\mathbf{s}_r\|_2, 10^{-12})$. The feature temperature τ is the lower-median absolute deviation of edge cosine dissimilarities about their lower median; if it is at most 10^{-6}, the mean absolute deviation about that median is used, and if that is also at most 10^{-6} the feature kernel is one. The spatial scale σ_r is the distance to the k-th neighbor; coincident centroids receive spatial factor one, and a positive distance with zero scale receives factor zero. The feature kernel and the full edge weight are each floored at $\epsilon_W = 10^{-12}$.

Recursion. Singletons retain their order without graph construction. For a scene with at least two regions, connected components at each recursion level are computed from the binary strong-edge adjacency of the current piece, with relative threshold $\rho_{\text{edge}} = 10^{-2}$:

$$E_{rr'} = \mathbf{1}[W_{rr'} > 0]\, \mathbf{1}[W_{rr'} \geq \rho_{\text{edge}} \mu_W], \qquad \mu_W = \text{med}_{\text{lo}}\{W_{rr'} : W_{rr'} > 0\}, \tag{9}$$

[2] https://huggingface.co/Qwen/Qwen2.5-VL-3B-Instruct/blob/main/config.json

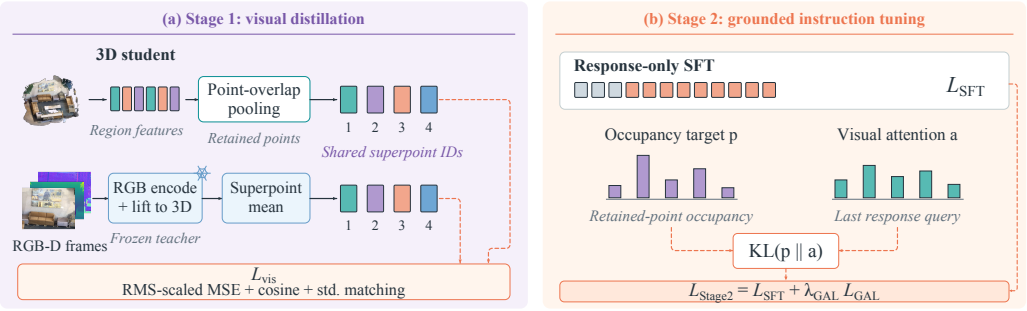


Figure 3: **Training objectives.** (a) In tokenizer order, point-overlap pooling maps student region features to teacher superpoints. Student pooling uses retained input points; cached RGB-D targets use a separate visibility mask. (b) Grounded instruction tuning combines response-only supervision with KL alignment of the decoder's relative visual attention $\mathbf{a}$ and the referent-fraction target $\mathbf{p}$ over regions; targets follow the visual features' region permutation. Attention is read at the query predicting the final assistant turn's last supervised non-special token.

where $E_{rr'} \in \{0, 1\}$ denotes strong-edge adjacency and μ_W is the lower-median positive weight, computed once from the scene graph. Each nontrivial component then uses all of its induced weighted edges, including weak ones, to form $\mathbf{L}_{\text{sym}}$ and the scores in Equation (4). Components occupy their parent's sequence interval in order of mean parent score, or of smallest original region index at the root. Child scores are oriented to have nonnegative covariance with the parent scores; at the root, the first score within 10^{-6} of the maximum magnitude is made positive. Scores are rounded to 10^{-6} bins and sorted stably. Let n be the number of regions in the current piece and $n_{\text{leaf}} = 16$ the leaf-size threshold. A piece with more than n_{leaf} regions splits after $\lfloor n/2 \rfloor$ sorted regions and recurses; pieces with at most n_{leaf} regions emit their local order. Singletons, pairs, and pieces whose nontrivial eigenvalue range is at most 2×10^{-7} fall back to tokenizer order. Pieces with at most 32 regions use dense FP64 eigendecomposition; larger pieces use sparse Chebyshev-filtered subspace iteration with a dense fallback when residuals exceed 10^{-8}. Graph storage is $O(Rk)$; exact neighbor search computes all $O(R^2)$ centroid distances, and the dense fallback is cubic in piece size.

A.4 Visual alignment loss

Figure 3 illustrates both training objectives.

Point membership and overlap pooling. Sampling and augmentation preserve each point's superpoint ID $m(i)$. Overlap counts in Equation (5) use the exact tokenizer assignment $r(i) > 0$, excluding padding, invalid labels, and masked patches. IDs are kept separate across scenes in a microbatch, and cache lookup uses persistent superpoint IDs. Correspondence is determined entirely by point membership, with no nearest-centroid approximation.

Teacher targets and reliability weights. Let $v_{if} \in \{0, 1\}$ indicate that point i has a valid feature $\mathbf{t}_{if}$ lifted from RGB-D frame f. The cache averages point–view observations within each superpoint:

$$\nu_m = \sum_{i:m(i)=m} \sum_f v_{if}, \qquad \mathbf{t}_m = \frac{\sum_{i:m(i)=m} \sum_f v_{if} \mathbf{t}_{if}}{\nu_m}, \quad \nu_m > 0. \tag{10}$$

Here ν_m counts valid point–view observations over accepted scene frames f; each visible point contributes once per frame. Teacher visibility defines the teacher mean independently of the student's retained-patch mask; only points with $r(i) > 0$ enter student overlap counts. A superpoint enters the loss only with $\nu_m > 0$ and $\sum_r n_{mr} > 0$. Let $\mathcal{M}$ contain these supported scene–superpoint pairs in the local microbatch. Stage 1 requires $\mathcal{M} \neq \varnothing$; all superpoint sums in this subsection range over $\mathcal{M}$, with scene identity included in m. The reliability weight in Equation (6) is $w_m = \nu_m / \sum_{m' \in \mathcal{M}} \nu_{m'}$; it is distinct from the geometric overlap coefficient α_{mr} used to pool student features.

Feature discrepancy. The loss is defined on the region features $\mathbf{s}_r$ and teacher targets $\mathbf{t}_m$ of Section 3.4, indexed as representation $j = 0$. If the host exposes J auxiliary region-feature representations, we index them by $j = 1, \ldots, J$, with region features $\mathbf{s}_r^{(j)}$ ($\mathbf{s}_r^{(0)} = \mathbf{s}_r$), and apply the same

overlap coefficients α_{mr} to each. For Qwen3-VL-8B these are its $J = 3$ DeepStack merger outputs (vision layers 8, 16, and 24); Qwen2.5-VL-3B has none, so $J = 0$ and λ_{aux} is inactive. Let D_j be the feature width, with $D_0 = d_l$, and let $\bar{\mathbf{s}}_m^{(j)}, \mathbf{t}_m^{(j)} \in \mathbb{R}^{D_j}$ be the aligned student and teacher features. Define the detached teacher root-mean-square (RMS) scale

$$\kappa_j = \max\left(10^{-6}, \text{stopgrad}\sqrt{\frac{1}{D_j}\sum_{m\in\mathcal{M}} w_m \|\mathbf{t}_m^{(j)}\|_2^2}\right). \tag{11}$$

Here stopgrad preserves the forward value and blocks its gradients, fixing the normalization for the student loss. Shared teacher scaling makes regression and statistic errors relative to target magnitude while preserving magnitude differences across regions, which independent region normalization would remove. For features $\mathbf{x}_m$ of a representation, let $\boldsymbol{\mu}_{\mathbf{x}} = \sum_m w_m \mathbf{x}_m$ and

$$\boldsymbol{\sigma}_{\mathbf{x}} = \sqrt{\sum_m w_m (\mathbf{x}_m - \boldsymbol{\mu}_{\mathbf{x}})^{\odot 2} + \epsilon_{\text{fp32}}\kappa_j^2}, \tag{12}$$

where $\boldsymbol{\mu}_{\mathbf{x}}$ and $\boldsymbol{\sigma}_{\mathbf{x}}$ are the weighted channel means and standard deviations, $\odot 2$ denotes element-wise squaring, the square root acts element-wise, and ϵ_{fp32} is the FP32 machine epsilon. Suppressing representation superscripts, the objective for one representation is

$$\mathcal{J}_j = \lambda_{\text{MSE}} \sum_m w_m \frac{\|\bar{\mathbf{s}}_m - \mathbf{t}_m\|_2^2}{D_j\kappa_j^2} + \lambda_{\cos}\sum_m w_m\big(1 - \cos(\bar{\mathbf{s}}_m, \mathbf{t}_m)\big) + \frac{\lambda_{\text{std}}}{D_j\kappa_j^2}\|\boldsymbol{\sigma}_{\bar{\mathbf{s}}} - \boldsymbol{\sigma}_{\mathbf{t}}\|_2^2, \tag{13}$$

where $\cos(\mathbf{x}, \mathbf{y})$ denotes cosine similarity, with vector norms clamped away from zero for numerical stability. This instantiates the discrepancy $\mathcal{D}$ in Equation (6) with feature regression, cosine alignment, and channel-statistic matching; the statistic term depends on the full supported microbatch and is counted once. With $J = 0$, $\mathcal{L}_{\text{vis}} = \mathcal{J}_0$; otherwise

$$\mathcal{L}_{\text{vis}} = \mathcal{J}_0 + \frac{\lambda_{\text{aux}}}{J}\sum_{j=1}^{J}\mathcal{J}_j. \tag{14}$$

The nonnegative coefficients weight feature regression, cosine alignment, channel-statistic matching, and the mean auxiliary-representation loss, respectively. The reference loss weights are $(\lambda_{\text{MSE}}, \lambda_{\cos}, \lambda_{\text{std}}, \lambda_{\text{aux}}) = (1, 0.1, 0.1, 1)$.

Alternative student-region alignment. The reference Stage 1 configuration transports student features to teacher superpoints; the alternative moves teacher features to student regions:

$$\omega_{rm} = \nu_m \frac{n_{mr}}{\sum_{r'} n_{mr'}}, \qquad \tilde{\nu}_r = \sum_m \omega_{rm}, \qquad \tilde{\mathbf{t}}_r^{(j)} = \frac{\sum_m \omega_{rm}\mathbf{t}_m^{(j)}}{\tilde{\nu}_r}, \tag{15}$$

where ω_{rm} is the observation-weighted overlap and $\tilde{\nu}_r$ is the total transferred observation weight of region r. Only regions with $\tilde{\nu}_r > 0$ are supported. Equation (13) then compares $\mathbf{s}_r^{(j)}$ with $\tilde{\mathbf{t}}_r^{(j)}$ using weights proportional to $\tilde{\nu}_r$, normalized over supported regions in the microbatch. The same transported targets and normalized region weights replace the superpoint targets and weights in the RMS and channel-statistic calculations. Only the loss-evaluation partition changes.

A.5 3D-GROUNDED ATTENTION COMPUTATION

For example b, R_b is the number of valid regions and η the smoothing weight of Section 3.4. The target uses the points retained by the tokenizer after sampling, augmentation, and grouping, with the same mask for the region point count and the referent count. For ScanQA the referent set contains the annotated objects associated with the question; for ScanRefer and Nr3D dense captioning it contains the target object. SQA3D examples have no referent set; they, like examples whose referent points are all dropped, contribute only to the response loss but remain in the batch size B. After spectral ordering, sequence slot j receives target $p_{b,\pi_{\text{spec}}(j)}$. Let q_b be the row that predicts the last supervised, non-special token before the final assistant turn terminator, and let $e_{br}^{(\ell,h)}$ be its query–key score for region r, layer ℓ, and head h. With H query heads, the relative visual attention is

$$a_{br}^{(\ell)} = \frac{1}{H}\sum_{h=1}^{H}\frac{\exp(e_{br}^{(\ell,h)})}{\sum_{r'=1}^{R_b}\exp(e_{br'}^{(\ell,h)})}. \tag{16}$$

All visual keys precede q_b. Scores follow each host's native query/key path: projections, query/key normalization where present (Qwen3-VL but not Qwen2.5-VL), head scaling, multimodal language RoPE, and grouped-query head expansion. Restricting softmax to visual keys permits computation from the selected query and key rows without materializing the full attention matrix. We supervise all decoder layers with $\lambda_{\mathrm{GAL}} = 1$ and $\eta = 0.05$. This single-query loss neither constrains every generated token nor guarantees faithful explanations or localization.

Response-loss reduction. For example b, let $\mathcal{P}_b$ be its point cloud, x_b its initial text prompt, and y_b its tokenized continuation, including subsequent user and assistant turns in multi-turn dialogue. Write y_{bt} for token t and $y_{b,<t}$ for its preceding continuation tokens. Let $\mathcal{A}_b$ contain the supervised assistant positions in y_b and p_Θ be the model's conditional token distribution, with Θ collecting its parameters. With $M = \sum_{b=1}^{B} |\mathcal{A}_b| > 0$ supervised tokens in the microbatch, the response loss is

$$\mathcal{L}_{\mathrm{SFT}} = -\frac{1}{M} \sum_{b=1}^{B} \sum_{t \in \mathcal{A}_b} \log p_\Theta(y_{bt} \mid \mathcal{P}_b, x_b, y_{b,<t}). \tag{17}$$

Prompt and padding positions are excluded. This is a token average across the microbatch, whereas Equation (7) averages over examples and layers, including examples with zero grounding loss. The KL is computed in at least FP32 from log attention, with $0 \log 0 = 0$ and padding excluded. Their combined loss is divided once by the actual gradient-accumulation length.

B Datasets and training details

B.1 Datasets

Stage 2 trains on the ScanQA, SQA3D, and ScanRefer training splits in Section 4.1 together with Nr3D object descriptions (Achlioptas et al., 2020); no 3D-LLM annotations are used.

B.2 Offline RGB-D teacher features

The teacher is the unadapted image pathway of the same base model (Qwen3-VL-8B or Qwen2.5-VL-3B): its image processor, visual encoder, and main merger produce cached Stage 1 targets without the language model. For Qwen3-VL, each of its three DeepStack mergers supplies an additional cached target stream. Registered depth and calibration link image tokens to scene points.

We visit temporally subsampled `scannet_frames_25k` frames in order with stride one, accepting at most 16 usable views per scene. A view requires at least 128 reconstructed points passing the visibility test. Let z_{if}^{cam} be point i's depth in frame f's camera coordinates, $(x_{if}^{\mathrm{img}}, y_{if}^{\mathrm{img}})$ its projected pixel, and $d_f(x, y)$ the observed depth map in meters. A point must have positive camera depth, project inside the depth image excluding a 10-pixel border, and satisfy $|z_{if}^{\mathrm{cam}} - d_f(x_{if}^{\mathrm{img}}, y_{if}^{\mathrm{img}})| \leq 0.25\, d_f(x_{if}^{\mathrm{img}}, y_{if}^{\mathrm{img}})$ at a pixel with positive observed depth. Registration compares color and depth intrinsics normalized by their image dimensions and requires agreement within 0.02, allowing different resolutions when color and depth observe the same rays; views failing this check are skipped. Each visible point takes the post-merger image token whose grid cell contains its normalized projected pixel; targets, stored in FP16 and upcast to FP32 for the loss, follow Equation (10). Superpoints without valid observations receive no supervision. Training and validation caches remain separate; variants with the same teacher, preprocessing, views, and superpoints share targets.

B.3 Point sampling and task construction

Point sampling keeps all points by default; the tokenizer's distance mask can still exclude patches (Section A.2). A positive sampling budget selects exactly that many points, without replacement when possible; scene-seeded validation sampling gives deterministic targets. Collators pad clouds and propagate validity masks, excluding padding from features and loss targets.

Joint training concatenates the datasets in Section B.1, so task frequency is proportional to dataset size. Both backbones use joint training without benchmark-specific fine-tuning; for each benchmark, all reported metrics come from the same checkpoint. Tasks other than SQA3D use random reflections

Table 6: Reference launcher defaults for the two training stages on eight workers.

Setting	Stage 1	Stage 2
Optimizer	AdamW	AdamW
Initial learning rate	5×10^{-4}	10^{-4}
Minimum learning rate	10^{-6}	10^{-6}
Schedule	Cosine decay	Cosine decay
Weight decay	0.05	0.1
Warmup fraction	0.05	0.03125
Gradient clipping norm	1.0	0.1
Local batch / accumulation	2 / 2	1 / 4
Effective batch	32	32
Duration	5,000 updates	8 epochs
Numeric precision	BF16	BF16
Distributed optimizer state	ZeRO-2	ZeRO-2
Visual LoRA rank / α	32 / 64	32 / 64
Language LoRA rank / α	—	32 / 64
Language LoRA dropout	—	0.05
Maximum language sequence	—	3,072

about the two vertical coordinate planes and rotations of up to $\pm 5^\circ$ about the vertical axis. SQA3D keeps the textual situation with the question and uses neither numeric pose inputs nor geometric augmentation; its training samples draw the situation text from the released paraphrases, and its validation and test annotations use ScanNet-validation geometry. Evaluation disables augmentation and repetition; ScanQA uses no object prompts.

B.4 Optimization

Table 6 specifies reference launcher defaults. Stage 1 trains the tokenizer and visual adaptation modules against cached teacher features. Stage 2 restores these parameters and adds language LoRA, using response prediction and the grounding loss in Equation (7), which applies to question-answering and captioning samples with annotated referents. The Stage 1 and Qwen2.5-VL-3B module ablations in Tables 5 and 7 use 20k Stage 2 updates. Effective batch equals worker count times local batch times gradient accumulation. LoRA scales each adapter's update by α divided by its rank.

C Evaluation protocols

Evaluation uses the official benchmark splits with augmentation and repetition disabled. Stage 1 validation measures feature alignment; Stage 2 validation loss measures response prediction with the auxiliary attention loss disabled. Generation metrics are computed separately from these losses. Deterministic decoding stops at <|im_end|> or the end token.

ScanQA. We report the validation split under the LL3DA scoring convention, decoding with four beams and at most 16 new tokens over all 4,675 validation questions, with early stopping enabled.

SQA3D. Decoding uses five beams, 1–64 new tokens, and no early stopping. Exact match (EM@1, test split in Table 1) applies the task's answer cleaning and uses the first reference.

Dense captioning. ScanRefer evaluation uses Mask3D proposals after greedy box non-maximum suppression (NMS) at IoU 0.5, without a score threshold. Each of the 2,068 ScanRefer validation objects is captioned from its best-overlapping proposal, whose normalized box precedes ScanRefer's description instruction in the prompt. We use four beams and at most 64 new tokens, and score with LL3DA's metrics at IoU ≥ 0.5 with 30-word truncation.

NDTokenizer3D front-end count. Because Tang et al. (2026b) report no front-end cost, we instantiated the public model definition on CPU with the released training configuration (Point Transformer V3 (PTv3) backbone with 12-dimensional normal-distribution-transform (NDT) inputs, three-layer multi-scale decoder with feature width 256, and the $256 \rightarrow 1{,}024$ alignment projection) and counted learnable parameters, replacing CUDA-only operators by parameter-faithful stubs. The scene-token path has 101,307,104 parameters: 97,544,528 in the PTv3 backbone, 3,495,824 in the

decoder including its classification and mask heads, 263,168 in the alignment projection, and 3,584 in the positional MLP and dataset-condition embedding. The backbone stores five dataset-specific normalization sets, of which one is active at inference; counting only that set gives 101.06M. The count excludes the LLaVA projector (20.98M), the prompt visual sampler (3.24M), the mask decoder (3.50M), and the `[SEG]` projection (4.46M), which serve interactive prompting and segmentation. Other baseline counts follow Mei et al. (2026a), whose projector accounting may differ.

D Analysis of component comparisons

D.1 Additional module ablations

Table 7 repeats the module ablation of Table 3 with Qwen2.5-VL-3B.

Table 7: Qwen2.5-VL-3B module ablations (ScanQA and ScanRefer validation, SQA3D test).

	ScanQA				ScanRefer @0.5				SQA3D
Variant	B-4 ↑	C ↑	R ↑	M ↑	B-4 ↑	C ↑	R ↑	M ↑	EM@1 ↑
Baseline	14.3	87.4	41.6	17.8	36.7	77.4	58.3	28.7	54.6
+ Grounded attention	15.3	91.7	43.5	<u>18.3</u>	40.9	<u>80.6</u>	61.8	<u>29.9</u>	55.1
+ Spectral ordering	<u>15.6</u>	<u>92.4</u>	**44.2**	18.0	<u>42.1</u>	80.2	<u>62.7</u>	29.6	<u>55.3</u>
Full (ReTok3D)	**16.1**	**93.2**	<u>44.1</u>	**18.6**	**44.9**	**81.4**	**65.5**	**30.8**	**55.9**

Variant definitions. The variants in Table 7 share Qwen2.5-VL-3B, Stage 1 initialization, distance masking, and the 2,048-region cap. The baseline uses Hilbert order and response supervision in Stage 2; + Grounded attention adds the KL loss of Equation (7) ($\lambda_{\mathrm{GAL}} = 1$, $\eta = 0.05$), + Spectral ordering reorders the regions, and Full, the 3B entry of Tables 1 and 2, combines both.

Results. The full variant improves all nine metrics over the baseline, raising CIDEr from 87.4 to 93.2 on ScanQA and from 77.4 to 81.4 on ScanRefer, and SQA3D EM@1 from 54.6 to 55.9. Each module alone also improves all nine metrics, raising ScanQA CIDEr to 91.7 with attention supervision and to 92.4 with spectral ordering; combining them is best on eight of nine metrics. Together with the Qwen3-VL-8B comparison in Table 3, these results support using both modules, although individual gains vary across hosts and tasks and run-to-run variability is not quantified.

Limitations and future work. Region pooling trades fine spatial detail for a compact sequence, and distance masking reduces retained scene coverage. Visual distillation depends on RGB-D view coverage, while spectral ordering adds 0.17–0.18 s per scene (Table 8). Compute-matched Stage 1 comparisons and repeat-run variability estimates would clarify the accuracy–cost trade-offs. The module study compares spectral ordering with Hilbert order only; Morton order, geometry-only or feature-only spectral graphs, and removal of the language-side grid indices remain untested. Attention localization is measured on separate diagnostic runs rather than the reported checkpoints (Table 9). Evaluation covers ScanNet-based tasks with two Qwen hosts; transfer to other scene distributions and geometry-only inputs remains to be assessed.

D.2 Comparison scope and mask coverage

Ablation protocol. Tables 3 to 5 and 7 organize module, Stage 1, and region-budget comparisons. Each comparison holds the backbone fixed and reports selected-checkpoint performance. The Qwen2.5-VL-3B module comparison and the Stage 1 comparison use 20k Stage 2 updates. The two-stage variant additionally incurs the cost of visual distillation, so equal Stage 2 duration does not imply equal total compute. The reported scores do not quantify variation across training runs.

Comparisons across backbones. The systems in Tables 1 and 2 differ in pretraining, model size, visual processing, and adaptation, so their scores give system-level context. Their differences reflect the complete systems and do not isolate tokenizer effects. Front-end counts omit pretrained computation and cannot measure total-system efficiency; Section D.3 measures end-to-end cost.

Mask coverage. With the reference mask at the default budget, the 312 ScanNet validation scenes keep 94.5% of their occupied patches on average (minimum 81.1%), and the 2,068 ScanRefer

validation referents keep 93.2% of their points ($r(i) > 0$; 5th percentile 81.1%); sampling drops none.

D.3 Inference cost and attention diagnostics

Table 8: End-to-end inference cost per ScanQA validation question on one A100 GPU, batch size one, 16 generated tokens. Tokens: visual tokens entering the language decoder. Visual: visual encoder and connector. Views: synthetic 640×480 images on the same host (cost reference).

			Prefill TFLOPs			Latency (s)			
Host	Input	Tokens	Visual	LM	Total	Visual	Spectral	Total	Mem. (GB)
Qwen2.5-VL-3B	Points (ours)	2,045	12.0	13.1	25.1	0.24	0.18	1.22	7.5
	8 views	3,128	16.7	21.0	37.6	0.98	–	1.81	7.7
	16 views	6,256	33.3	47.6	80.9	1.95	–	2.93	8.2
	32 views	12,512	66.7	118.4	185.0	3.90	–	5.20	9.3
Qwen3-VL-8B	Points (ours)	2,045	15.8	32.0	47.8	0.14	0.17	1.31	17.2

End-to-end cost. Table 8 measures complete inference in bfloat16 with SDPA attention, averaged over 100 evenly spaced ScanQA validation questions after five warm-up questions. Both hosts use the reference configuration (0.02 m voxels, $K = 5$, $R_{\max} = 2{,}048$, distance mask, and spectral ordering) with a scratch-initialized tokenizer and identity adapters. Learned values change only the eigensolver's iterations and the output length, so every run generates exactly 16 greedy tokens. Latencies are CUDA-synchronized, and prefill FLOPs are counted with PyTorch's FLOP counter at two FLOPs per multiply–add, including dense attention; the counted tokenizer costs, 16.4G and 14.8G, match Table 1. A point-cloud prefill costs 25.1 TFLOPs with Qwen2.5-VL-3B and 47.8 TFLOPs with Qwen3-VL-8B. The reused visual encoder and connector account for 48% and 33% of these FLOPs, and the tokenizer for less than 0.1%. On the Qwen2.5-VL-3B host, eight 640×480 views cost 37.6 TFLOPs and 1.81 s for the same output, compared with 25.1 TFLOPs and 1.22 s for the point cloud, and 32 views cost 185.0 TFLOPs and 5.20 s. Spectral ordering adds 0.18 and 0.17 s per scene, comparable to the visual encoder, and is the largest point-specific latency. Loading and geometric preprocessing take a further 0.14–0.15 s per scene on the CPU. Decoding takes 44 and 50 ms per token, and peak memory exceeds the weights by 0.4 and 0.7 GB.

Table 9: Decoder attention on held-out referents in Qwen2.5-VL-3B diagnostic runs after 2,000 updates. Mass: attention on regions containing referent points. Top-1: the most-attended region contains the referent. KL: Equation (7) on validation targets.

	ScanRefer (2,068 referents)			ScanQA (4,675 questions)		
Variant	Mass (%) ↑	Top-1 (%) ↑	KL ↓	Mass (%) ↑	Top-1 (%) ↑	KL ↓
Uniform attention	3.5	3.5	–	7.6	7.6	–
Baseline	3.3	2.0	5.10	7.0	4.6	4.30
+ Spectral ordering	3.4	2.7	5.44	7.5	7.2	4.53
+ Grounded attention	**24.8**	**72.6**	**2.09**	**12.9**	**19.5**	**2.39**

Attention on held-out referents. Table 9 tests whether attention supervision localizes referents in held-out scenes. We teacher-force each validation target and read the head-normalized decoder attention that Equation (7) supervises, from the query predicting the last answer token, averaged over decoder layers. Uniform attention gives a mass equal to the fraction of regions containing referent points, which is also the chance rate of the top-1 measure. The diagnostic uses separate Qwen2.5-VL-3B runs trained by direct SFT for 2,000 updates on a mixture that adds 3D-LLM data, with Morton serialization and no distance mask; they are not the checkpoints of Table 7. Without the loss, attention on referents stays at chance on both benchmarks. With it, the mass rises from 3.3% to 24.8% on ScanRefer, whose prompt gives the referent's box, and from 7.0% to 12.9% on ScanQA, whose questions give no location; the most-attended region contains the referent in 72.6% and 19.5% of examples. Spectral ordering alone leaves attention near chance. The loss thus transfers to held-out scenes, but attention need not explain the answers.

E QUALITATIVE RESULTS

Figure 4 presents selected successful predictions for scene-level question answering, situated question answering, and object captioning. The examples illustrate object identification, spatial relations, and answers conditioned on a textual situation.

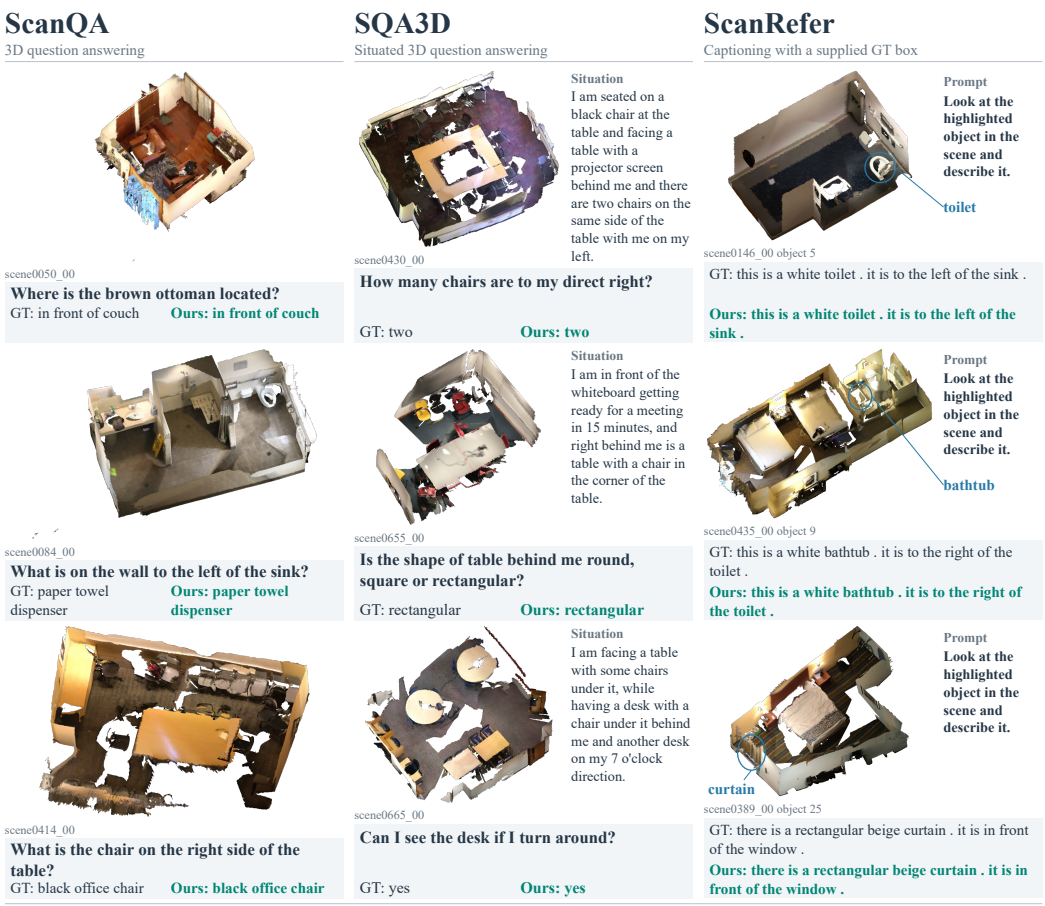


Figure 4: Qualitative examples from ScanQA (left), SQA3D (middle), and ScanRefer (right). Each example pairs a scene rendering with the question or instruction, a ground-truth reference (GT), and our prediction. SQA3D additionally provides the textual situation. The ScanRefer examples use supplied ground-truth boxes; the blue annotations identify the target objects for visualization. These examples illustrate caption generation given an object box, while Table 2 reports evaluation with Mask3D proposals. The model receives point clouds and text; renderings are illustrative.