GEMI3D: GEOMETRY-GROUNDED MODALITY-INDEPENDENT INDOOR 3D OBJECT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Indoor 3D object detection localizes and recognizes objects in 3D space from images, depth maps, point clouds, or their combinations, but existing detectors are typically designed for predefined input configurations. When the available inputs change, these detectors may require architectural changes or additional training. We therefore study modality-independent indoor 3D object detection, where a single detector handles any combination of image, depth, and point-cloud inputs. To this end, we propose GEMI3D, a *geometry-grounded* modality-independent framework for indoor 3D object detection, which maps heterogeneous inputs into a stable 3D geometric hub. **First**, Cross-Modal Hub Construction learns a stable spatial representation from point-cloud data and then extends the geometry-grounded hub with depth and image inputs mapped into the same 3D space, while feature anchoring preserves the point-cloud representation. **Second**, Modality-Conditioned Hub Adaptation uses modality dropout to expose the detector to different input combinations and fuses the available hub features for a modality-agnostic detection head. To comprehensively evaluate modality independence with inputs from distinct sources, we further construct TriScan3D, a frame-level benchmark derived from ScanNet++. With all modalities, GEMI3D achieves **68.6/52.3** mAP_{25}/mAP_{50} on SUN RGB-D and **70.7/51.4** on TriScan3D. Across all seven non-empty input combinations, it maintains robust detection performance without retraining. The code will be released upon acceptance.

1 INTRODUCTION

Indoor 3D object detection localizes and recognizes objects in 3D space, providing spatial information for indoor scene understanding. Existing detectors take RGB images, depth maps, point clouds, or their combinations as input (Qi et al., 2019; Song & Xiao, 2016; Qi et al., 2020; Rukhovich et al., 2022b). To exploit the complementary information carried by these inputs, multimodal perception has progressed from feature-level fusion, such as enriching points or voxels with image information (Sindagi et al., 2019; Vora et al., 2020), toward shared representation spaces and unified architectures that facilitate cross-modal alignment (Wang et al., 2022b; Liu et al., 2023). Nevertheless, detectors built around these modalities generally assume a predefined input configuration and remain tied to a fixed sensor setup, as illustrated in Fig. 1(a) and (b).

In practice, indoor perception systems span robotic platforms (Wang et al., 2024b; Lin et al., 2025), commodity mobile devices (Baruch et al., 2021), and AR platforms (Zhao et al., 2024), each with distinct sensor configurations. Such variability often requires different model designs or additional training to reuse a detector across devices or scenarios (Reza et al., 2024). Existing strategies, such as modality dropout (Neverova et al., 2016; Wang et al., 2024c), sensor-failure adaptation (Ge et al., 2023), and uniform encoders (Wang et al., 2024a), can improve robustness to missing modalities, but remain specialized training schemes or model structures for expected sensor settings.

For indoor 3D detection, achieving such modality independence is particularly challenging. Images mainly provide appearance and semantic cues, whereas depth maps and point clouds primarily encode geometry. These modality-specific cues make it challenging to maintain semantic–geometric alignment and consistent predictions when the available inputs change. We therefore study *modality-independent indoor 3D object detection*, which aims to use a single detector across different combinations of available image, depth, and point-cloud inputs without architectural changes or retraining.

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

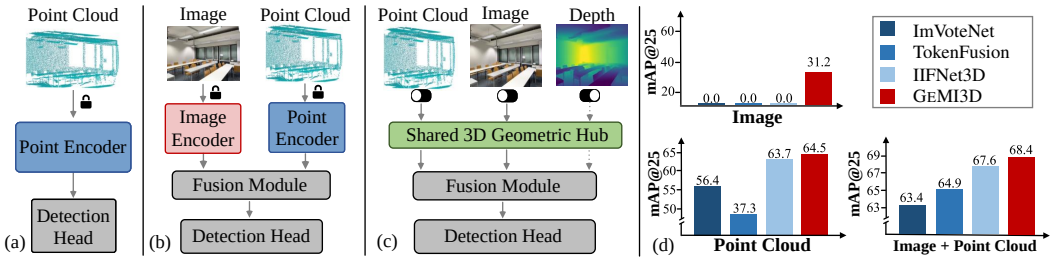

Figure 1: **Comparison of fixed-input and modality-independent indoor 3D detection** (§1). **(a, b)** Fixed-input detectors rely on predefined sensor configurations and modality-specific encoders; locks denote fixed input paths. **(c)** GeMI3D maps image, depth, and point-cloud inputs to a shared 3D geometric hub, enabling one detector to handle different input configurations; switches indicate the active paths. **(d)** mAP_{25} across different input configurations on SUN RGB-D (Song et al., 2015). GeMI3D remains effective under single-modality inputs, whereas fixed-input models degrade substantially when an expected modality is missing.

To address this problem, we propose GeMI3D, a geometry-grounded framework that maps image, depth, and point-cloud inputs into aligned 3D point sets. A shared encoder then processes the resulting points to form the *geometric hub* (Fig. 1(c)). Thus, a single detector can operate under different input combinations without architectural changes. GeMI3D is implemented through Cross-Modal Hub Construction (CMHC) and Modality-Conditioned Hub Adaptation (MCHA). **First**, CMHC trains the geometric hub on point clouds with detection supervision to establish a stable spatial representation. It then incorporates depth and image inputs by lifting them into 3D using observed or estimated depth (Yang et al., 2024a) and camera geometry (Wang et al., 2019; Qian et al., 2020). During cross-modal extension, feature anchoring preserves the point-cloud representation, while joint geometric and detection supervision maintains geometric fidelity and task relevance. **Second**, MCHA applies modality dropout during training, projects and fuses features from the active paths in the shared 3D space, and passes them to a modality-agnostic detection head. Together, CMHC and MCHA combine shared geometric representation learning with feature fusion over the available modalities, enabling modality-independent detection under different input combinations.

To evaluate modality independence with inputs from distinct sources, we construct TriScan3D, a frame-level benchmark derived from ScanNet++ (Yeshwanth et al., 2023) that pairs RGB-D observations with point clouds from independently reconstructed scene geometry. Experiments on SUN RGB-D (Song et al., 2015) and TriScan3D show strong performance with all modalities available: GeMI3D achieves **68.6/52.3** mAP_{25}/mAP_{50} on SUN RGB-D and **70.7/51.4** on TriScan3D. As shown in Fig. 1(d) and §4.3, the model operates across all seven input combinations without retraining and remains robust. These results highlight the effectiveness of shared geometric grounding and modality-conditioned feature fusion.

In a nutshell, GeMI3D makes three contributions. **First**, we identify sensor-configuration coupling as an important limitation in indoor 3D object detection, where changes in available image, depth, or point-cloud inputs often require architectural modifications or additional training and may degrade detection performance. **Second**, our key insight is to address this issue through input-level geometric grounding, establishing a shared 3D representation as a common basis for feature fusion and detection. GeMI3D realizes this insight through CMHC, which first trains the geometric hub on point clouds and then jointly extends it to depth and image inputs with feature anchoring. **Third**, we develop MCHA, which combines modality dropout with active-path feature projection and fusion, allowing a shared detection head to operate with different input combinations.

2 Related Work

Indoor 3D Object Detection. Indoor 3D detection is dominated by point-cloud methods, encoding geometry with point-based (Qi et al., 2017) or sparse-voxel (Graham & Van der Maaten, 2017) backbones, with attention-based variants developed subsequently (Zhao et al., 2021; Wu et al., 2022; 2024; Yang et al., 2023). Built on these representations, early detectors aggregate local evidence through voting and proposal generation (Qi et al., 2019; Zhang et al., 2020), while later methods explore query-based set prediction (Liu et al., 2021; Misra et al., 2021; Shen et al., 2024) and anchor-

free formulations (Rukhovich et al., 2022a; Zhao et al., 2026). Recent studies have increasingly focused on object grouping (Wang et al., 2022a; Zhu et al., 2024), sparse supervision (Zhu et al., 2025), and cross-dataset adaptation (Kolodiazhnyi et al., 2025; Zhao et al., 2025; Xie et al., 2026).

Beyond point clouds, images complement point features through image voting and cross-modal fusion (Qi et al., 2020; Wang et al., 2022c; Sun et al., 2026). They also serve as primary inputs: RGB-D detectors construct 3D volumes or frustum point sets (Song & Xiao, 2016; Qi et al., 2018), while image-based detectors lift monocular or multi-view features into 3D representations (Rukhovich et al., 2022b; Zhang et al., 2025). More recently, large-scale cuboid annotations and 2D supervision have advanced image-centric 3D detection at scale (Lazarow et al., 2025a; Jhang et al., 2025).

Despite their different designs, all the methods above assume a fixed input form. Methods that relax 3D annotation, pose, or sensor-geometry requirements remain specialized image pipelines (Hsu et al., 2025; Lazarow et al., 2025b; Cao et al., 2026), while selective modality dropout handles missing RGB or depth within a predefined RGB-D setting (Piekenbrinck et al., 2024). Flexible indoor detection across RGB, depth, point clouds, and their subsets remains underexplored. GEMI3D addresses this gap by mapping available modalities into a shared 3D geometric hub, enabling the same detector to handle different input subsets.

Missing Modalities and Modality Independence. Missing-modality learning studies multimodal robustness when one or more modalities are absent. Existing methods use modality dropout (Neverova et al., 2016), information transfer or generation (Vapnik & Vashist, 2009; Wei et al., 2023; Chen et al., 2024; Le et al., 2024), and failure-aware fusion or decoding (Ge et al., 2023; Cha et al., 2024) to mitigate performance degradation caused by missing modalities.

Recent work further supports varying or arbitrary modality combinations (Wang et al., 2024a; Zheng et al., 2024; Yun et al., 2024). Such flexibility is closely related to modality independence (Sun et al., 2023), which aims to reduce architectural and representational dependence on predefined modality types or combinations. In indoor 3D object detection, test-time modality switching among image, point-cloud, and joint inputs has been explored (Wang et al., 2024c). However, its separate image and point-cloud feature extractors leave the encoding architecture and representations tied to these predefined modalities and extending the model to additional modalities such as depth would require corresponding modality-specific encoding paths. GEMI3D instead maps available inputs into a common 3D form and processes them with a shared geometric hub and detector, reducing modality-specific dependence in the core model.

3 METHOD

3.1 FORMULATION AND OVERVIEW

Task Formulation. Existing indoor 3D object detectors are commonly designed for a fixed input form (Qi et al., 2019; Song & Xiao, 2016; Qi et al., 2020), and their performance can degrade when expected modalities are missing or the sensor configuration changes (Piekenbrinck et al., 2024). To move beyond this fixed-input setting, we formulate the task of modality-independent indoor 3D object detection, in which a single detector handles any non-empty subset of image, depth, and point-cloud inputs. Let $\boldsymbol{I} \in \mathbb{R}^{H\times W\times 3}$, $\boldsymbol{D} \in \mathbb{R}^{H\times W}$, and $\boldsymbol{P} \in \mathbb{R}^{N_P\times 3}$ denote an RGB image, a depth map, and the 3D coordinates of a point cloud, respectively, where H and W denote the image height and width, and N_P is the number of raw input points. Optional point-wise features are omitted for clarity. Let $\mathcal{A} \subseteq \{I, D, P\}$ denote a non-empty subset of available modalities, and $\boldsymbol{X}_m$ be the observation from $m \in \mathcal{A}$. Our task is to use a single detector f_Θ for every valid input subset:

$$\boldsymbol{B}_{\text{3D}} = f_\Theta(\{\boldsymbol{X}_m \mid m \in \mathcal{A}\}), \tag{1}$$

where $\boldsymbol{B}_{\text{3D}}$ contains object categories and 3D boxes.

Framework Overview. To achieve this objective, we build a shared 3D geometric hub that decouples sensor inputs from downstream detection. Given the camera intrinsic matrix $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$, each available observation is converted into a point representation and encoded through the same hub:

$$\boldsymbol{P}_m = \mathcal{T}_m(\boldsymbol{X}_m; \boldsymbol{K}) \in \mathbb{R}^{N_m\times 3}, \tag{2}$$

$$\boldsymbol{F}_m = \texttt{Encoder}(\boldsymbol{P}_m) \in \mathbb{R}^{N'_m\times C}, \tag{3}$$

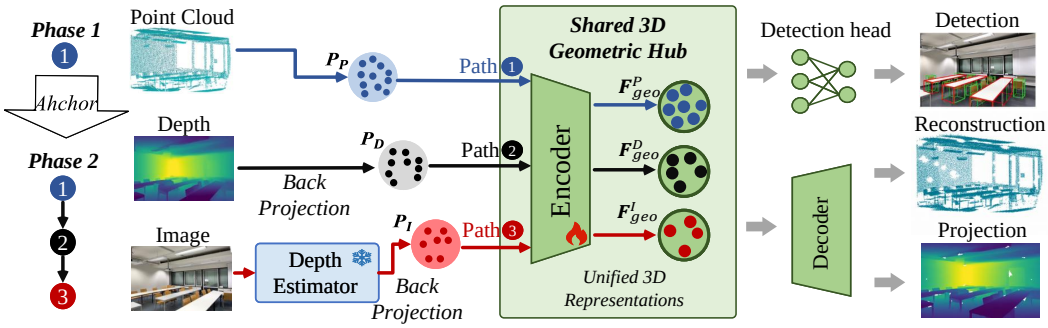

Figure 2: **Overview of Cross-Modal Hub Construction (CMHC)** (§3.2). Phase 1 trains the geometric hub on point clouds under detection supervision to establish a stable spatial representation. Phase 2 incorporates depth and image paths as back-projected 3D point sets, with image depth supplied by a frozen estimator. Feature anchoring preserves the point-cloud representation learned in Phase 1, while reconstruction, projection, and detection objectives maintain geometry and task-relevant structure.

where $\mathcal{T}_m$ denotes the modality-specific geometric conversion, N_m and N'_m are the numbers of input points and encoded 3D features, and C is the feature dimension. The resulting modality-wise hub features are then aligned and fused according to $\mathcal{A}$ before entering a shared 3D detection head.

The shared geometric hub is learned and adapted through two components: **i**) Cross-Modal Hub Construction (CMHC) establishes it through detection-supervised point-cloud training and extends the learned representation to depth and image inputs with feature anchoring (§3.2); and **ii**) Modality-Conditioned Hub Adaptation (MCHA) aligns and fuses the available hub features under modality dropout (§3.3), supporting a modality-agnostic 3D detection head (§3.4) across input combinations.

3.2 Cross-Modal Hub Construction

To obtain a unified representation across input modalities, CMHC constructs the geometric hub in two phases. **Phase 1** initializes the hub through detection-supervised training on point clouds, establishing a stable spatial representation. **Phase 2** extends the hub to depth and image inputs, with feature anchoring preserving the point-cloud representation learned in Phase 1.

In Phase 1, CMHC trains the geometric hub with point-cloud inputs under detection supervision to establish a stable spatial representation, as point clouds directly describe 3D geometry. Given a point cloud $\boldsymbol{P}$, the hub uses $\boldsymbol{P}_P = \mathcal{T}_P(\boldsymbol{P}) = \boldsymbol{P}$ and obtains $\boldsymbol{F}_P = \texttt{Encoder}(\boldsymbol{P}_P)$.

In Phase 2, depth and image inputs are incorporated into the geometric hub. A depth map $\boldsymbol{D}$ is back-projected into 3D using camera intrinsics $\boldsymbol{K}$ (Wang et al., 2019):

$$\boldsymbol{P}_D = \mathcal{T}_D(\boldsymbol{D}; \boldsymbol{K}) = \texttt{BackProj}(\boldsymbol{D}; \boldsymbol{K}) \in \mathbb{R}^{N_D \times 3}, \tag{4}$$

where $\texttt{BackProj}$ lifts depth pixels to 3D coordinates and N_D is the number of resulting points.

For the image path, depth $\boldsymbol{D}_I$ is first estimated from $\boldsymbol{I}$ by a depth estimator $\texttt{DepthEst}$ (Ranftl et al., 2021; Yang et al., 2024a;b). It is then back-projected into a point representation:

$$\boldsymbol{D}_I = \texttt{DepthEst}(\boldsymbol{I}) \in \mathbb{R}^{H \times W}, \tag{5}$$

$$\boldsymbol{P}_I = \mathcal{T}_I(\boldsymbol{I}; \boldsymbol{K}) = \texttt{BackProj}(\boldsymbol{D}_I; \boldsymbol{K}) \in \mathbb{R}^{N_I \times 3}, \tag{6}$$

where N_I denotes the number of image-derived points. The resulting point sets $\boldsymbol{P}_m$ are processed by the shared encoder to obtain $\boldsymbol{F}_m = \texttt{Encoder}(\boldsymbol{P}_m)$ for $m \in \{P, D, I\}$. Feature anchoring preserves the hub's initial point-cloud representation by enforcing object-level feature consistency with the frozen Phase 1 hub, while reconstruction and projection objectives maintain geometric fidelity and detection supervision retains task-relevant structure.

3.3 Modality-Conditioned Hub Adaptation

MCHA adapts the geometric hub to changing input availability by learning to integrate active-path features into a unified representation for the modality-agnostic detection head. During training,

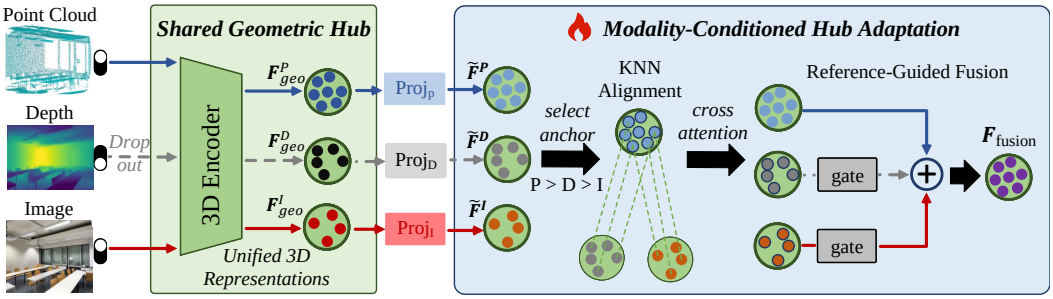

Figure 3: **Overview of Modality-Conditioned Hub Adaptation (MCHA)** (§3.3). Modality dropout exposes the model to different active input paths during training. Features from the active paths are extracted by the geometric hub, projected into a shared feature space, aligned to an available reference selected with priority $P \succ D \succ I$, and combined through reference-guided fusion.

modality dropout (Neverova et al., 2016) varies the active modality set $\mathcal{A}$. The shared encoder processes the active paths with lightweight feature calibration: residual input adapters and modality-specific batch normalization accommodate their different feature statistics. Modality-specific projectors then map the calibrated features into a shared fusion space:

$$\tilde{\boldsymbol{F}}_m = \texttt{Proj}_m(\boldsymbol{F}_m), \qquad m \in \mathcal{A}, \tag{7}$$

where $\tilde{\boldsymbol{F}}_m \in \mathbb{R}^{N'_m \times C}$, and $\texttt{Proj}_m$ preserves the sparse coordinates of the hub features. A consistency objective in §3.5 further encourages agreement between co-located projected features.

Although all paths use the same 3D coordinate system, their sparse features may occupy different spatial locations. For spatially coherent fusion, MCHA selects the highest-priority active path under $P \succ D \succ I$ as the reference r. This order reflects how directly each path provides 3D geometry: point clouds enter directly, depth is back-projected, and images additionally require depth estimation. Let $\mathcal{C}_m$ denote the sparse 3D coordinates associated with $\tilde{\boldsymbol{F}}_m$. MCHA then establishes local correspondences between each auxiliary path and the reference:

$$\mathcal{N}_{m \to r} = \texttt{KNN}_k(\mathcal{C}_r, \mathcal{C}_m), \qquad m \in \mathcal{A} \setminus \{r\}, \tag{8}$$

where $\mathcal{N}_{m \to r}$ associates each reference location with its k nearest locations on path m.

Gated cross-attention fuses locally matched auxiliary features $\tilde{\boldsymbol{F}}_m$ with the reference features $\tilde{\boldsymbol{F}}_r$:

$$\boldsymbol{F}_{\text{fusion}} = \tilde{\boldsymbol{F}}_r + \sum_{m \in \mathcal{A} \setminus \{r\}} g_m \, \texttt{CrossAttn}\Big(\tilde{\boldsymbol{F}}_r, \tilde{\boldsymbol{F}}_m; \mathcal{N}_{m \to r}\Big), \tag{9}$$

where $\tilde{\boldsymbol{F}}_r$ form the queries, $\tilde{\boldsymbol{F}}_m$ provide the keys and values, and $\mathcal{N}_{m \to r}$ restricts attention to the KNN matches. The learnable scalar g_m controls the contribution of auxiliary path m.

When only one path is active, it serves as the reference and no cross-modal feature is injected. Overall, varying the available inputs changes only the active paths and reference selection, while the model architecture and parameters remain unchanged.

3.4 Modality-Agnostic Detection Head

For any active modality set $\mathcal{A}$, MCHA produces $\boldsymbol{F}_{\text{fusion}}$ in a common feature space and passes it to a shared modality-agnostic detection head $\texttt{Detector3D}$ (Rukhovich et al., 2022a; 2023):

$$\boldsymbol{B}_{\text{3D}} = \texttt{Detector3D}(\boldsymbol{F}_{\text{fusion}}). \tag{10}$$

During training, the resulting predictions are supervised by the detection objective in §3.5.

3.5 Training Objectives

CMHC Objectives. For $m \in \{P, D, I\}$, we define the path-wise detection loss as $\mathcal{L}_{\text{det},m} = \mathcal{L}_{\text{cls},m} + \mathcal{L}_{\text{box},m} + \lambda_{\text{qual}} \mathcal{L}_{\text{qual},m}$. Here, $\mathcal{L}_{\text{cls},m}$ and $\mathcal{L}_{\text{box},m}$ denote the classification and box regression losses, while $\mathcal{L}_{\text{qual},m}$ uses matched-box 3D IoU as the soft target for sigmoid binary cross-entropy (Wu et al., 2020). Phase 1 trains the hub on point clouds by minimizing $\mathcal{L}^{(1)}_{\text{CMHC}} = \mathcal{L}_{\text{det},P}$.

In Phase 2, a training-only decoder reconstructs $\hat{\boldsymbol{P}}_m$ from $\boldsymbol{F}_m$ using the bidirectional squared Chamfer loss (Fan et al., 2017): $\mathcal{L}_{\text{rec},m} = \text{CD}(\hat{\boldsymbol{P}}_m, \boldsymbol{P}_m)$. For $m \in \{D, I\}$, the projection operator $\Pi_{\boldsymbol{K}}$ maps $\hat{\boldsymbol{P}}_m$ to depth and soft coverage maps: $(\hat{\boldsymbol{D}}_m, c_m) = \Pi_{\boldsymbol{K}}(\hat{\boldsymbol{P}}_m)$. The projection loss is

$$\mathcal{L}_{\text{proj},m} = \frac{\sum_{u\in\mathcal{V}_m} c_m(u)|\hat{\boldsymbol{D}}_m(u) - \boldsymbol{D}_m(u)|}{\sum_{u\in\mathcal{V}_m} c_m(u)} + \frac{\lambda_{\text{cov}}}{|\mathcal{V}_m|}\sum_{u\in\mathcal{V}_m}(1 - c_m(u)), \tag{11}$$

where $\boldsymbol{D}_m \in \{\boldsymbol{D}, \boldsymbol{D}_I\}$ is the target depth, $\mathcal{V}_m = \{u \mid \boldsymbol{D}_m(u) > 0\}$ is its valid-pixel set, and $c_m(u) \in [0, 1]$ denotes the soft coverage of pixel u by the projected points. The two terms penalize depth error and insufficient projection coverage, respectively.

At each level $\ell \in \{1, 2, 3\}$, point-cloud features $\boldsymbol{F}_P^\ell$ are anchored to frozen Phase 1 features $\boldsymbol{F}_{P,0}^\ell$:

$$\mathcal{L}_{\text{anc}} = \frac{1}{\sum_{\ell=1}^{3}|\mathcal{B}_\ell|}\sum_{\ell=1}^{3}\sum_{b\in\mathcal{B}_\ell}\Big[1 - \cos\big(\text{Pool}_b(\boldsymbol{F}_P^\ell), \text{Pool}_b(\boldsymbol{F}_{P,0}^\ell)\big)\Big], \tag{12}$$

where $\mathcal{B}_\ell$ contains ground-truth boxes with sufficient co-located features, and Pool_b denotes mean pooling over the corresponding features in box b.

The Phase 2 objective combines reconstruction, projection, detection, and feature anchoring:

$$\begin{aligned}\mathcal{L}_{\text{CMHC}}^{(2)} = \frac{\lambda_{\text{rec}}}{3}\sum_{m\in\{P,D,I\}}\mathcal{L}_{\text{rec},m} + \frac{\lambda_{\text{proj}}}{2}\sum_{m\in\{D,I\}}\mathcal{L}_{\text{proj},m} \\ + \lambda_{\text{det}}\sum_{m\in\{P,D,I\}} w_m\mathcal{L}_{\text{det},m} + \lambda_{\text{anc}}\mathcal{L}_{\text{anc}},\end{aligned} \tag{13}$$

where w_m is the detection weight for path m, and each λ weights its corresponding loss.

MCHA Objectives. For an active modality set $\mathcal{A}$, MCHA applies the same detection loss and regularizes $\{\tilde{\boldsymbol{F}}_m \mid m \in \mathcal{A}\}$ with $\mathcal{L}_{\text{cons}}(\mathcal{A})$. Let L_h be the number of selected hub levels and $\tilde{\boldsymbol{F}}_m^\ell$ the level-ℓ component of $\tilde{\boldsymbol{F}}_m$. Let $\mathcal{S}_\mathcal{A}$ be the set of unordered active-path pairs, and Ω_{mn}^ℓ denote the co-located positions of paths m and n at level ℓ. The consistency loss is

$$\mathcal{L}_{\text{cons}}(\mathcal{A}) = \frac{1}{\max(1, |\mathcal{S}_\mathcal{A}|L_h)}\sum_{\{m,n\}\in\mathcal{S}_\mathcal{A}}\sum_{\ell=1}^{L_h}\frac{\sum_{x\in\Omega_{mn}^\ell}\|\tilde{\boldsymbol{F}}_m^\ell(x) - \tilde{\boldsymbol{F}}_n^\ell(x)\|_1}{C_\ell\max(1, |\Omega_{mn}^\ell|)}, \tag{14}$$

where C_ℓ is the feature dimension at level ℓ. The consistency loss is zero when fewer than two paths are active. The overall objective is $\mathcal{L}_{\text{MCHA}}(\mathcal{A}) = \mathcal{L}_{\text{det}}(\mathcal{A}) + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}}(\mathcal{A})$, which fine-tunes the hub and jointly optimizes the feature-calibration, projection, and fusion modules.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our method on two indoor datasets: SUN RGB-D and TriScan3D.

- **SUN RGB-D** (Song et al., 2015) was captured using 4 different RGB-D sensors across diverse indoor scenes. Following the standard SUN RGB-D train and val split (Song et al., 2015), we use 5,285 training samples and 5,050 validation samples. The dataset provides oriented 3D bounding-box annotations spanning 37 object categories. Following the standard evaluation protocol (Qi et al., 2019), we train and evaluate the model on the 10 most frequent object categories.
- **TriScan3D** is our 10-category frame-level benchmark constructed from ScanNet++ (Yeshwanth et al., 2023), which provides high-fidelity laser-scanned geometry, DSLR images, and commodity-level iPhone RGB-D streams. We use a scene-disjoint 80/20 split to prevent cross-view leakage while preserving the target-class distribution, yielding 14,862 training and 3,668 validation samples. Each sample comprises an RGB image, a depth map, and a frame-level point cloud derived from the reconstructed scene mesh, enabling evaluation with RGB-D observations and geometry from an independent 3D source. Additional construction details are provided in the appendix.

Table 1: **Quantitative comparison with fixed-input indoor 3D detectors** on SUN RGB-D (Song et al., 2015) and TriScan3D (§4.2). †CuTR results follow its SUN RGB-D protocol (Lazarow et al., 2025a), with at most 100 detections per image. ‡TokenFusion's hard feature replacement is sensitive to the source-specific surface and sampling differences between frame RGB and mesh-derived point clouds in TriScan3D.

Modality	Detector	Params	FPS	SUN RGB-D		TriScan3D	
				mAP_{25}	mAP_{50}	mAP_{25}	mAP_{50}
P	VoteNet[ICCV2019]	1.0M	22.2	57.7	32.9	57.5	29.1
I+P	ImVoteNet[CVPR2020]	43.4M	16.6	63.4	38.0	61.5	30.6
I	ImVoxelNet[WACV2022]	104.5M	32.9	41.0	13.5	31.7	5.5
I+P	TokenFusion[CVPR2022]	22.4M	10.5	64.9	48.3	57.2‡	31.9‡
P	TR3D[ICIP2023]	14.7M	22.4	67.1	50.4	68.0	46.2
P	SPGroup3D[AAAI2024]	16.8M	14.4	65.4	47.1	68.4	47.1
I+D	CuTR[CVPR2025]	99.2M	15.5	59.4†	34.0†	–	–
I+P	IIFNet3D[PR2026]	72.3M	8.2	67.6	47.5	67.2	48.4
P	SR3D[AAAI2026]	–	–	68.1	50.9	–	–
I+D+P	GEMI3D (Ours)	68.8M	6.8	**68.6**	**52.3**	**70.7**	**51.4**
P		68.8M	12.2	64.5	49.1	68.3	49.8

Input Settings. We denote RGB images, depth maps, and point clouds as I, D, and P, and consider all seven non-empty combinations: I, D, P, I+D, I+P, D+P, and I+D+P. Without retraining, GEMI3D is evaluated on all seven combinations by disabling unavailable paths, whereas fixed-input baselines are evaluated on their native modality sets and subsets with missing inputs zeroed.

Evaluation Metrics. We evaluate indoor 3D object detection using mean Average Precision (mAP). Predictions are ranked by confidence and matched to ground-truth boxes of the same class using a specified 3D IoU threshold. AP is the area under the resulting precision–recall curve, and mAP averages AP across classes. Following the standard indoor 3D detection protocol (Qi et al., 2019), we report mAP at IoU thresholds of 0.25 and 0.5, denoted as mAP_{25} and mAP_{50}, respectively.

Implementation Details. We implement GEMI3D in MMDetection3D (Contributors, 2020) with MinkowskiEngine (Choy et al., 2019), Point Transformer V3 (Wu et al., 2024) as the shared encoder, and a TR3D-style detection head (Rukhovich et al., 2023). Image pixels are lifted into 3D predicted by frozen Depth Anything V2 model (Yang et al., 2024b) and the corresponding camera intrinsics, then transformed into the scene coordinate system using the camera poses. Complete implementation and training details are in the appendix and the code will be released upon acceptance.

4.2 Quantitative Results under Fixed-Input Settings

Table 1 compares GEMI3D with representative fixed-input detectors under their native configurations. With all three modalities, GEMI3D achieves **68.6/52.3** mAP_{25}/mAP_{50} on SUN RGB-D and **70.7/51.4** on TriScan3D. These results exceed SR3D by **0.5/1.4** on SUN RGB-D, and SPGroup3D and IIFNet3D by **2.3/4.3** and **3.5/3.0** on TriScan3D. Under P-only input, GEMI3D obtains 64.5/49.1 on SUN RGB-D and 68.3/49.8 on TriScan3D, remaining competitive with dedicated point-cloud detectors. These results indicate that the shared geometric hub effectively integrates complementary modalities while retaining competitive detection performance under P-only input.

4.3 Quantitative Results under Different Input Combinations

We next evaluate detector reliability across changing input modalities, without retraining any model. Table 2(a) first tests fixed-input multimodal detectors with their complete I+P input and either modality alone. Trained with I+P, ImVoteNet, TokenFusion, and IIFNet3D obtain 63.4, 64.9, and 67.6 mAP_{25} on SUN RGB-D. Their scores decrease to 56.4, 37.3, and 63.7 with P-only input, whereas I-only input causes an extreme drop to 0.0, 0.0, and 0.0. The same trend is observed on TriScan3D. These results show that fixed-input detectors are strongly coupled to their predefined inputs and struggle particularly when the expected point-cloud geometry is unavailable.

Table 2(b) evaluates GEMI3D under all seven non-empty input combinations. GEMI3D achieves 68.6 and 70.7 mAP_{25} on SUN RGB-D and TriScan3D under the full I+D+P input. Under I-only, D-only, and P-only inputs, the corresponding scores are 31.2, 64.1, and 64.5 on SUN RGB-D and 19.9, 42.2, and 68.3 on TriScan3D. Results across all seven combinations indicate that the proposed

Table 2: **Quantitative results under different input combinations** on SUN RGB-D (Song et al., 2015) and TriScan3D using mAP_{25} (§4.3). (a) Fixed-input detectors trained with I+P are evaluated with complete inputs or either modality alone without retraining. (b) GEMI3D uses one model for all non-empty input combinations.

Method	SUN RGB-D			TriScan3D		
	I+P	P	I	I+P	P	I
ImVoteNet [CVPR2020]	63.4	56.4	0.0	61.5	56.8	0.0
TokenFusion [CVPR2022]	64.9	37.3	0.0	57.2	49.6	0.0
IIFNet3D [PR2026]	67.6	63.7	0.0	67.2	66.6	0.0

(a) Fixed-input detectors under missing modalities

Dataset	I	D	P	I+D	I+P	D+P	I+D+P
SUN RGB-D	31.2	64.1	64.5	68.0	68.4	64.8	68.6
TriScan3D	19.9	42.2	68.3	48.6	69.9	69.4	70.7

(b) GEMI3D under different input combinations

Table 3: **Component ablations of CMHC and MCHA** on SUN RGB-D (Song et al., 2015) using mAP_{25} (§4.4). I+D+P denotes the complete input, and Avg. is the mean over all seven non-empty input combinations.

Variant	I+D+P	P	D	I	Avg.
w/o Phase 1 training	66.4	59.3	59.7	29.6	58.3
w/o reconstruction&projection losses	67.2	64.1	64.4	29.7	60.7
w/o detection supervision	67.1	**64.7**	**64.7**	29.3	60.9
w/o feature anchoring	67.0	64.0	64.6	31.0	60.9
Full CMHC	**68.6**	64.5	64.1	**31.2**	**61.4**

(a) Cross-Modal Hub Construction (CMHC)

Variant	I+D+P	P	D	I	Avg.
w/o feature projector	67.9	64.8	64.3	31.3	61.3
Voxel-wise mean fusion	67.0	64.7	**64.6**	**32.2**	61.0
w/o consistency supervision	67.6	**64.9**	64.1	30.9	61.1
w/o hub fine-tuning	67.1	63.7	62.3	30.7	60.2
Full MCHA	**68.6**	64.5	64.1	31.2	**61.4**

(b) Modality-Conditioned Hub Adaptation (MCHA)

geometric hub supports both single-modality and partial-modality detection. Complete mAP and mAR results at both IoU thresholds are provided in Appendix Table 5.

4.4 DIAGNOSTIC EXPERIMENT

Tables 3 and 4 study the components of CMHC (§3.2) and MCHA (§3.3), as well as design choices involving modality dropout, feature anchoring, and feature calibration.

CMHC Components. Table 3(a) evaluates a Phase 2-only setting and three Phase 2 ablations targeting **i)** the reconstruction and projection losses, **ii)** detection supervision, and **iii)** feature anchoring. Compared with Full CMHC, complete-input mAP_{25} drops by 2.2, 1.4, 1.5, and 1.6. The seven-combination average drops by 3.1, 0.7, 0.5, and 0.5, supporting the contribution of Phase 1 point-cloud initialization and the individual Phase 2 designs.

MCHA Components. Table 3(b) then examines the roles of feature projection, reference-guided fusion, consistency supervision, and hub fine-tuning in MCHA; voxel-wise averaging serves as the fusion alternative. Relative to Full MCHA, the four variants reduce complete-input mAP_{25} by 0.7, 1.6, 1.0, and 1.5, while the seven-combination average mAP_{25} also decreases under all four settings, quantifying the effect of each MCHA design.

Additional Design Choices. Finally, Table 4 studies (a) dropout, (b) anchoring weight, and (c) feature calibration. The adopted dropout strategies raise average mAP_{25} over no dropout by 2.7/13.9 on SUN RGB-D/TriScan3D. On SUN RGB-D, weighted sampling exceeds uniform sampling by 1.1 in complete-input mAP_{25}, while uniform sampling is marginally better for single-modality inputs. On TriScan3D, uniform sampling slightly lowers complete-input mAP_{25} but improves all other results relative to weighted sampling, most notably D-only by 2.5, supporting balanced sampling for het-

Table 4: **Ablation studies** using mAP_{25} (§4.4). Modality dropout is evaluated on SUN RGB-D (Song et al., 2015) and TriScan3D, while the remaining ablations use SUN RGB-D. Avg. denotes the mean over all seven non-empty input combinations. The adopted settings are marked in red.

Dropout strategy	I	D	P	I+D+P	Avg.
Dataset: SUN RGB-D					
None (always I+D+P)	23.1	60.7	63.0	68.0	58.7
Uniform over seven	**31.7**	**64.4**	**65.1**	67.5	61.3
Weighted over seven	31.2	64.1	64.5	**68.6**	**61.4**
Dataset: TriScan3D					
None (always I+D+P)	0.0	15.8	66.5	**71.2**	41.7
Uniform over seven	**19.9**	**42.2**	**68.3**	70.7	**55.6**
Weighted over seven	19.4	39.7	68.1	71.1	55.0

(a) Modality dropout

λ_{anc}	I+D+P	Avg.
0	67.0	60.9
0.1	**68.6**	**61.4**
0.5	67.9	61.2
1.0	67.5	61.1

(b) Feature anchoring

Variant	Adapter	BN	I+D+P	Avg.
No calibration	–	–	67.7	60.2
Adapter only	✓	–	67.9	60.2
BN only	–	✓	68.0	61.1
Full calibration	✓	✓	**68.6**	**61.4**

(c) Feature calibration

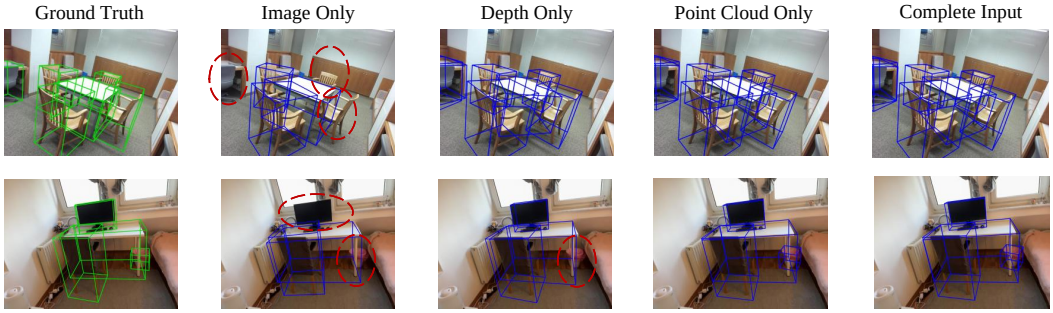

Figure 4: **Qualitative results** on SUN RGB-D (Song et al., 2015) (top) and TriScan3D (bottom) (§4.5). Differences are highlighted with red dashed circles ◌.

erogeneous sources. $\lambda_{anc} = 0.1$ gives the best complete-input mAP_{25} of 68.6, while full calibration improves complete-input mAP_{25} by 0.7/0.6 over adapter-only/BN-only.

4.5 Qualitative Results

Fig. 4 visualizes predictions on SUN RGB-D and TriScan3D with I-only, D-only, P-only, and I+D+P inputs. I-only detects most salient objects but misses some, whereas D-only recovers more but localizes some tables imprecisely. P-only and I+D+P yield more complete and accurate predictions through direct geometry and complementary multimodal cues. Together, these examples show that GeMI3D adapts to available inputs. Additional failure cases are provided in Fig. 5.

5 Conclusion

Indoor 3D detectors are often tied to fixed predefined configurations of image, depth, and point-cloud inputs, limiting reuse as inputs change. We propose GeMI3D, which maps heterogeneous inputs into a shared 3D geometric hub, enabling one detector to handle varying inputs without architectural changes or retraining. CMHC builds and extends the hub through feature anchoring and geometric and detection supervision, while MCHA combines modality dropout with feature projection and fusion for a shared detection head. Results on SUN RGB-D and TriScan3D show competitive performance with complete inputs and robustness to changing inputs. These results support shared geometric representation learning as a promising basis for indoor 3D detection under changing input configurations. Future work will explore adaptive reference selection and further improve robustness to degraded sensor inputs in real-world settings.

AI USE STATEMENT

In this work, generative AI tools were used for manuscript language editing and for general discussion of experimental settings. They were not used to implement the proposed method, generate data, conduct experiments, or interpret experimental results. All AI-assisted text and suggestions were independently reviewed and verified by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Our framework is implemented in PyTorch, and the experiments are conducted on four NVIDIA L40S GPUs. Complete implementation, training, evaluation, and TriScan3D construction details are provided in Appendices A–C. The code, configurations, selected frame identifiers, converted annotations, and preprocessing scripts will be released upon acceptance.

REFERENCES

Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *NeurIPS Datasets and Benchmarks Track*, 2021.

Yang Cao, Feize Wu, Dave Zhenyu Chen, Yingji Zhong, Lanqing Hong, and Dan Xu. Vggt-det: Mining vggt internal priors for sensor-geometry-free multi-view indoor 3d object detection. In *CVPR*, pp. 4708–4717, 2026.

Juhan Cha, Minseok Joo, Jihwan Park, Sanghyeok Lee, Injae Kim, and Hyunwoo J Kim. Robust multimodal 3d object detection via modality-agnostic decoding and proximity-based modality ensemble. *arXiv preprint arXiv:2407.19156*, 2024.

Mengxi Chen, Fei Zhang, Zihua Zhao, Jiangchao Yao, Ya Zhang, and Yanfeng Wang. Probabilistic conformal distillation for enhancing missing modality robustness. In *NeurIPS*, 2024.

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pp. 3075–3084, 2019.

MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020.

Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, pp. 605–613, 2017.

Chongjian Ge, Junsong Chen, Enze Xie, Zhongdao Wang, Lanqing Hong, Huchuan Lu, Zhenguo Li, and Ping Luo. Metabev: Solving sensor failures for 3d detection and map segmentation. In *ICCV*, pp. 8721–8731, 2023.

Benjamin Graham and Laurens Van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Peng-Hao Hsu, Ke Zhang, Fu-En Wang, Tao Tu, Ming-Feng Li, Yu-Lun Liu, Albert Y. C. Chen, Min Sun, and Cheng-Hao Kuo. Openm3d: Open vocabulary multi-view indoor 3d object detection without human annotations. In *ICCV*, pp. 8688–8698, 2025.

Jin-Cheng Jhang, Tao Tu, Fu-En Wang, Ke Zhang, Min Sun, and Cheng-Hao Kuo. V-mind: Building versatile monocular indoor 3d detector with diverse 2d annotations. In *WACV*, pp. 9577–9586, 2025.

Maksim Kolodiazhnyi, Anna Vorontsova, Matvey Skripkin, Danila Rukhovich, and Anton Konushin. Unidet3d: Multi-dataset indoor 3d object detection. In *AAAI*, pp. 4365–4373, 2025.

Justin Lazarow, David Griffiths, Gefen Kohavi, Francisco Crespo, and Afshin Dehghan. Cubify anything: Scaling indoor 3d object detection. In *CVPR*, pp. 22225–22233, 2025a.

Justin Lazarow, Kai Kang, and Afshin Dehghan. Rooms from motion: Un-posed indoor 3d object detection as localization and mapping. In *NeurIPS*, 2025b.

Duy-Tho Le, Hengcan Shi, Jianfei Cai, and Hamid Rezatofighi. Diffusion model for robust multi-sensor fusion in 3d object detection and bev segmentation. In *ECCV*, pp. 232–249, 2024.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pp. 936–944, 2017a.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pp. 2980–2988, 2017b.

Xuewu Lin, Tianwei Lin, Lichao Huang, Hongyu Xie, and Zhizhong Su. Bip3d: Bridging 2d images and 3d perception for embodied intelligence. In *CVPR*, pp. 9007–9016, 2025.

Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *ICCV*, pp. 2949–2958, 2021.

Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *ICRA*, pp. 2774–2781, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *ICCV*, pp. 2906–2917, 2021.

Natalia Neverova, Christian Wolf, Graham Taylor, and Florian Nebout. Moddrop: adaptive multi-modal gesture recognition. *TPAMI*, 2016.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Konyul Park, Yecheol Kim, Daehun Kim, and Jun Won Choi. Resilient sensor fusion under adverse sensor failures via multi-modal expert fusion. In *CVPR*, pp. 6720–6729, 2025.

Jens Piekenbrinck, Alexander Hermans, Narunas Vaskevicius, Timm Linder, and Bastian Leibe. RGB-D Cube R-CNN: 3D Object Detection with Selective Modality Dropout. In *CVPRW*, pp. 1997–2006, 2024.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pp. 652–660, 2017.

Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, pp. 918–927, 2018.

Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, pp. 9277–9286, 2019.

Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *CVPR*, pp. 4404–4413, 2020.

Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge Belongie, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3d object detection. In *CVPR*, pp. 5881–5890, 2020.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pp. 12179–12188, 2021.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

Md Kaykobad Reza, Ashley Prater-Bennette, and M Salman Asif. Robust multimodal learning with missing modalities via parameter-efficient adaptation. *TPAMI*, 2024.

Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Fcaf3d: Fully convolutional anchor-free 3d object detection. In *ECCV*, pp. 477–493, 2022a.

Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In *WACV*, pp. 2397–2406, 2022b.

Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Tr3d: Towards real-time indoor 3d object detection. In *ICIP*, pp. 281–285, 2023.

Yichao Shen, Zigang Geng, Yuhui Yuan, Yutong Lin, Ze Liu, Chunyu Wang, Han Hu, Nanning Zheng, and Baining Guo. V-detr: Detr with vertex relative position encoding for 3d object detection. In *ICLR*, pp. 3368–3386, 2024.

Vishwanath A Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection. In *ICRA*, pp. 7276–7282, 2019.

Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*, pp. 808–816, 2016.

Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015.

Jun Sun, Shoukang Han, Yu-Ping Ruan, Xiaoning Zhang, Shu-Kai Zheng, Yulong Liu, Yuxin Huang, and Taihao Li. Layer-wise fusion with modality independence modeling for multi-modal emotion recognition. In *ACL*, pp. 658–670, 2023.

Zhihao Sun, Zixin Fan, Bin Fan, and Hongmin Liu. Iifnet3d: Instance-to-instance fusion with dual attention for indoor rgb-d 3d object detection. *Pattern Recognition*, 2026.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 2009.

Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, pp. 4604–4612, 2020.

Haiyang Wang, Lihe Ding, Shaocong Dong, Shaoshuai Shi, Aoxue Li, Jianan Li, Zhenguo Li, and Liwei Wang. Cagroup3d: Class-aware grouping for 3d object detection on point clouds. In *NeurIPS*, 2022a.

Shiming Wang, Holger Caesar, Liangliang Nan, and Julian FP Kooij. Unibev: Multi-modal 3d object detection with uniform bev encoders for robustness against missing sensor modalities. In *IV*, pp. 2776–2783, 2024a.

Tai Wang, Xiaohan Mao, Chenming Zhu, Runsen Xu, Ruiyuan Lyu, Peisen Li, Xiao Chen, Wenwei Zhang, Kai Chen, Tianfan Xue, et al. Embodiedscan: A holistic multi-modal 3d perception suite towards embodied ai. In *CVPR*, pp. 19757–19767, 2024b.

Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, pp. 8445–8453, 2019.

Yikai Wang, Xinghao Chen, Lele Cao, Wenbing Huang, Fuchun Sun, and Yunhe Wang. Multimodal token fusion for vision transformers. In *CVPR*, pp. 12186–12195, 2022b.

Yikai Wang, TengQi Ye, Lele Cao, Wenbing Huang, Fuchun Sun, Fengxiang He, and Dacheng Tao. Bridged transformer for vision and point cloud 3d object detection. In *CVPR*, pp. 12114–12123, 2022c.

Zhenyu Wang, Yali Li, Taichi Liu, Hengshuang Zhao, and Shengjin Wang. Ov-uni3detr: Towards unified open-vocabulary 3d object detection via cycle-modality propagation. In *ECCV*, pp. 73–89, 2024c.

Shicai Wei, Yang Luo, and Chunbo Luo. One-stage modality distillation for incomplete multimodal learning. *arXiv preprint arXiv:2309.08204*, 2023.

Shengkai Wu, Xiaoping Li, and Xinggang Wang. Iou-aware single-stage object detector for accurate localization. *Image and Vision Computing*, 2020.

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *NeurIPS*, 2022.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *CVPR*, pp. 4840–4851, 2024.

Tao Xie, Tao An, Feng Liu, Wensheng Jin, Zhengyu Li, Lijun Zhao, and Ruifeng Li. H^2A^2: Homogeneity-aware and heterogeneity-aware feature perception for unified indoor 3d object detection. In *CVPR*, pp. 40717–40726, 2026.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, pp. 10371–10381, 2024a.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. In *NeurIPS*, 2024b.

Yu-Qi Yang, Yu-Xiao Guo, Jian-Yu Xiong, Yang Liu, Hao Pan, Peng-Shuai Wang, Xin Tong, and Baining Guo. Swin3D: A pretrained transformer backbone for 3d indoor scene understanding. *arXiv preprint arXiv:2304.06906*, 2023.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, pp. 12–22, 2023.

Sukwon Yun, Inyoung Choi, Jie Peng, Yangfan Wu, Jingxuan Bao, Qiyiwen Zhang, Jiayi Xin, Qi Long, and Tianlong Chen. Flex-moe: Modeling arbitrary modality combination via the flexible mixture-of-experts. In *NeurIPS*, 2024.

Runmin Zhang, Zhu Yu, Si-Yuan Cao, Lingyu Zhu, Guangyi Zhang, Xiaokai Bai, and Hui-Liang Shen. Boosting multi-view indoor 3d object detection via adaptive 3d volume construction. In *ICCV*, pp. 5980–5989, 2025.

Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *ECCV*, pp. 311–329, 2020.

Chenyu Zhao, Xianwei Zheng, Zimin Xia, Linwei Yue, and Nan Xue. Real-time 3d object detection with inference-aligned learning. In *AAAI*, pp. 13099–13107, 2026.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, pp. 16259–16268, 2021.

Yiqin Zhao, Ashkan Ganj, and Tian Guo. Towards in-context environment sensing for mobile augmented reality. In *ACM MobiCom*, pp. 2091–2097, 2024.

Zijing Zhao, Zhu Xu, Qingchao Chen, Yuxin Peng, and Yang Liu. Investigating domain gaps for indoor 3d object detection. In *ACMMM*, pp. 13198–13205, 2025.

Xu Zheng, Yuanhuiyi Lyu, and Lin Wang. Learning modality-agnostic representation for semantic segmentation from any modalities. In *ECCV*, pp. 146–165, 2024.

Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *AAAI*, pp. 12993–13000, 2020.

Yun Zhu, Le Hui, Yaqi Shen, and Jin Xie. Spgroup3d: Superpoint grouping network for indoor 3d object detection. In *AAAI*, pp. 7811–7819, 2024.

Yun Zhu, Le Hui, Hang Yang, Jianjun Qian, Jin Xie, and Jian Yang. Learning class prototypes for unified sparse supervised 3d object detection. In *CVPR*, pp. 9911–9920, 2025.

APPENDIX

For completeness, we provide additional details on reproducibility, additional experiments, the construction of TriScan3D, and the limitations and potential social impact of this work. The appendix is organized as follows:

- §A documents the assets and licenses, computing infrastructure, randomness controls, and key hyperparameters used in our experiments.
- §B provides additional ablations and qualitative failure cases omitted from the main paper.
- §C.1 motivates and positions TriScan3D; §C.2–§C.8 describe its construction pipeline; and §C.9 discusses its scope and limitations.
- §D and §E discuss the limitations and potential social impact of this work.

A REPRODUCIBILITY

A.1 ASSETS AND LICENSES

Our experiments use SUN RGB-D (Song et al., 2015) and TriScan3D, constructed from ScanNet++ (Yeshwanth et al., 2023) in §C. The raw RGB-D images, depth maps, camera parameters, reconstructed geometry, and original annotations should be obtained from the corresponding dataset releases and used under their original terms. For TriScan3D, we will release the selected frame identifiers, converted annotations, class mappings, and preprocessing scripts upon acceptance, while the raw ScanNet++ assets remain available through the official ScanNet++ release.

The implementation is based on MMDetection3D (Contributors, 2020) and uses MinkowskiEngine (Choy et al., 2019), Point Transformer V3 (Wu et al., 2024), a TR3D-style indoor detector (Rukhovich et al., 2023), and Depth Anything V2 (Yang et al., 2024b). Third-party code and pretrained weights are used under their respective licenses.

A.2 COMPUTING INFRASTRUCTURE

Experiments use four NVIDIA L40S GPUs, each with 48 GB of memory. Point-cloud initialization uses one GPU, whereas all subsequent training stages use four GPUs. We implement the network in PyTorch 2.1 with CUDA 12.1, using MMEngine $\geq$0.10, MMCV $\geq$2.1.0, MMDetection $\geq$3.2.0, MMDetection3D $\geq$1.4.0, and MinkowskiEngine 0.5.4.

Efficiency Evaluation Protocol. The parameter counts and inference throughput in Table 1 are measured under a unified protocol on one NVIDIA GeForce RTX 4090. We use batch size 1 and FP32 inference without test-time augmentation, warm up the model for 50 iterations, and then average throughput over 2,000 samples. Point-based detectors retain their configured sampling budgets and image-based detectors retain their official test-time preprocessing. Parameter counts include all modules required at deployment, including frozen components. Accordingly, GEMI3D contains 68.81M deployed parameters, of which 14.51M are trainable; the deployed count includes the frozen Depth Anything V2 estimator and frozen ResNet-50 FPN. For GEMI3D, we benchmark both complete I+D+P inference and P-only inference using the same checkpoint and model parameters.

A.3 IMPLEMENTATION AND TRAINING DETAILS

Implementation. The shared 3D encoder uses Point Transformer V3 with MinkowskiEngine sparse operations, and the detection head follows TR3D. Within the shared encoder, each modality uses a residual input adapter with 16 hidden channels, and each batch-normalization layer maintains modality-specific affine parameters and running statistics; all other encoder weights are shared. The image path uses a frozen Depth Anything V2 metric-depth estimator with a ViT-S encoder, an input size of 518, a maximum depth of 20.0 m, and piecewise depth correction; valid depths are restricted to $[0.1, 8.0]$ m. The final DINOv2 (Oquab et al., 2023) patch map from the Depth Anything V2 encoder is bilinearly sampled at valid image pixels and attached to the corresponding lifted RGB points. Level-specific projections then add these semantic features residually to the hub representation. We separately train a Faster R-CNN (Ren et al., 2015) with a ResNet-50 FPN (He

et al., 2016; Lin et al., 2017a), then freeze and use only its backbone and FPN neck to extract multi-scale RGB features. When the image path is active, features sampled at the projected 3D locations of the highest-priority available path under $P \succ D \succ I$ are transformed by sparse 1×1 convolutions and added to hub levels 1–3 with a residual scale of 0.1 before cross-modal fusion. MCHA uses local k-nearest-neighbor fusion with $k = 16$.

Algorithm 1 summarizes the complete CMHC and MCHA training pipeline.

Algorithm 1 Pseudo-code of the GEMI3D Training Pipeline (§A.3).

```
# X_P/X_D/X_I: point-cloud/depth/image observations
# H/G: geometric hub/shared detector; T_m: geometric conversion
# CMHC Phase 1: initialize the hub from point clouds (§3.2)
for batch in point_cloud_loader:
  P_P = T_P(batch.X_P)
  F_P = H(P_P, path=P)
  update(H, G, loss=det_loss(G(F_P), batch.targets))
H_anchor = freeze(copy(H))

# CMHC Phase 2: cross-modal hub extension (§3.2; objectives: §3.5)
for batch in cross_modal_loader:
  L_cmhc, features = 0, {}
  for m in [P, D, I]:
    P_m = T_m(batch[m], batch.camera)
    features[m] = H(P_m, path=m)
    P_hat_m = Decoder(features[m])
    L_cmhc += lambda_rec / 3 * chamfer(P_hat_m, P_m)
    L_cmhc += lambda_det * w[m] * det_loss(
        G(features[m]), batch.targets)
    if m in [D, I]:
      D_hat_m, coverage_m = project(P_hat_m, batch.intrinsics)
      L_cmhc += lambda_proj / 2 * projection_loss(
          D_hat_m, coverage_m, batch.depth[m])
  F_P_anchor = H_anchor(T_P(batch.X_P))
  L_cmhc += lambda_anc * anchor_loss(features[P], F_P_anchor)
  update(H, adapters, BN, Decoder, G, loss=L_cmhc)

# MCHA: adapt fusion to active modalities (§3.3; objectives: §3.5)
freeze(G.neck, G.head)
for batch in multimodal_loader:
  A = sample_nonempty_subset(probabilities)
  projected = {}
  for m in A:
    P_m = T_m(batch[m], batch.camera)
    F_m = H(P_m, path=m)       # path adapter + path-specific BN
    projected[m] = Projector[m](F_m)
  r = select_reference(A, priority=[P, D, I])
  F_fusion = projected[r]
  for m in A - {r}:
    matches = knn(projected[r].coords, projected[m].coords, k=16)
    F_fusion += gate[m] * cross_attention(
        projected[r], projected[m], matches)
  boxes = G(F_fusion)          # shared detection head (§3.4)
  L_mcha = det_loss(boxes, batch.targets)
  L_mcha += lambda_cons * consistency_loss(projected, A)
  update(H, adapters, BN, Projector, fusion,
         loss=L_mcha, hub_lr=0.1, base_lr=1.0)
return H, adapters, BN, Projector, fusion, G
```

Training Schedule. All conducted experiments use random seed 4, a voxel size of 0.03 m, and AdamW (Loshchilov & Hutter, 2017) with weight decay 10^{-4}, 500-iteration linear warmup, cosine learning-rate decay, and gradient clipping at norm 10. CMHC Phase 1 point-cloud training runs for 20 epochs with batch size 8. CMHC Phase 2 cross-modal extension runs for 20 epochs with batch size 1 per GPU and learning rate 3×10^{-4}. MCHA runs for 20 epochs with batch size 6 per GPU and learning rate 9×10^{-4}. Both CMHC Phase 2 and MCHA sample $100{,}000$ points per scene. On SUN RGB-D, the training set is repeated five times per epoch for CMHC Phase 1 and MCHA, whereas CMHC Phase 2 uses one pass per epoch. The Faster R-CNN is trained for 12 epochs with batch size 8 per GPU; its ResNet-50 backbone and FPN neck are then frozen for MCHA.

Objectives. The soft projection loss in Eq. 11 is evaluated at one-quarter of the input resolution. For reconstruction, the Chamfer loss uses at most $8{,}192$ points from each predicted and target set, while the decoder output is capped at $65{,}536$ points. In Eq. 13, $(w_P, w_D, w_I) = (1.0, 0.25, 0.25)$ and $\lambda_{\text{anc}} = 0.1$. The reconstruction, projection, and detection weights follow a piecewise-linear loss-weight schedule: they are initialized to 1.0, 0.02, and 0.01, held fixed through epoch 5, linearly

adjusted to 0.5, 0.01, and 0.03 by epoch 12, and held fixed thereafter. Detection uses focal loss (Lin et al., 2017b) for classification and rotated 3D DIoU loss (Zheng et al., 2020) for box regression. The consistency loss in Eq. 14 is applied to the last two feature levels with weight 0.1, sampling at most 8,192 co-located positions per modality pair and level.

Input Sampling and Fine-Tuning. During MCHA training, SUN RGB-D uses sampling probabilities of 0.55, 0.07, 0.12, 0.06, 0.03, 0.04, and 0.13 for I+D+P, D+P, I+P, I+D, P, D, and I, respectively, whereas TriScan3D samples the seven non-empty combinations uniformly. The hub remains trainable, with its backbone optimized at $0.1\times$ the base learning rate, while the detection neck and head remain frozen. The input adapters, modality-specific BN parameters, projectors, fusion module, and semantic-refinement modules are optimized at the base rate. At test time, each input combination is selected through its active-path mask while the model weights and detection head remain unchanged.

B Additional Experimental Results and Ablations

B.1 Complete Results under All Input Combinations

Table 5: **Complete Evaluation Under All Input Combinations** (§B.1). We report mAP and mAR at 3D IoU thresholds of 0.25 and 0.50 for all seven non-empty input combinations on SUN RGB-D (Song et al., 2015) and TriScan3D, without retraining between combinations.

Input	mAP_{25}	mAP_{50}	mAR_{25}	mAR_{50}
I	31.2	9.2	87.8	38.6
D	64.1	48.7	96.8	75.8
P	64.5	49.1	96.9	75.2
I+D	68.0	51.1	97.3	75.7
I+P	68.4	52.0	97.2	76.2
D+P	64.8	49.3	96.9	75.8
I+D+P	68.6	52.3	97.4	76.8

(a) SUN RGB-D

Input	mAP_{25}	mAP_{50}	mAR_{25}	mAR_{50}
I	19.9	3.3	70.4	21.8
D	42.2	21.5	85.0	46.1
P	68.3	49.8	93.3	71.3
I+D	48.6	24.4	87.6	49.0
I+P	69.9	50.8	93.6	71.8
D+P	69.4	50.7	93.5	71.6
I+D+P	70.7	51.4	93.5	72.0

(b) TriScan3D

Table 5 extends the evaluation in §4.3 to both IoU thresholds and recall metrics. Across the P-containing settings (P, I+P, D+P, and I+D+P), all four metrics remain close on both datasets, with mAR_{25} and mAR_{50} showing particularly small variation as the auxiliary modalities change. When P is unavailable, I-only and D-only inputs yield lower AP but retain comparatively high recall, indicating that the detector still recovers many objects despite reduced detection precision and localization quality. These results show that the robustness observed with mAP_{25} also extends to stricter localization and recall metrics.

B.2 Additional Ablations

Table 6: **Additional Objective Ablations** on SUN RGB-D (Song et al., 2015) using mAP_{25} (§B.2). I+D+P denotes the complete input, and Avg. is the mean over all seven non-empty input combinations. Adopted settings are marked in red.

Variant	I+D+P	Avg.
w/o reconstruction ($\lambda_{\text{rec}} = 0$)	67.2	60.8
w/o projection ($\lambda_{\text{proj}} = 0$)	68.2	61.0
w/o loss-weight scheduling	67.8	60.7
Full CMHC	**68.6**	**61.4**

(a) Detailed CMHC objectives

λ_{cons}	I+D+P	Avg.
0	67.6	61.1
0.1	**68.6**	**61.4**
0.5	68.5	61.1
1.0	67.9	60.6

(b) Consistency weight

Table 6(a) further examines the CMHC objectives. Removing reconstruction, projection, and loss-weight scheduling reduces complete-input/average mAP_{25} by 1.4/0.6, 0.4/0.4, and 0.8/0.7, respectively. These results show that all three components contribute. In Table 6(b), $\lambda_{\text{cons}} = 0.1$ achieves the best complete-input/average mAP_{25} of 68.6/61.4. Removing consistency supervision or assigning it a larger weight reduces performance, indicating that moderate cross-path consistency improves alignment.

Table 7 isolates the additional image semantic features while retaining Depth Anything V2 for geometric lifting. Adding R50-FPN features improves I-only and complete-input mAP_{25} by 1.4 and 0.7, respectively. Further incorporating DINOv2 point features yields additional gains of 3.5 and 1.7. These results show that multi-scale and point-level image semantics complement the geometry-grounded image path without changing the shared detection architecture.

Table 7: **Ablation of Image Semantic Refinement** on SUN RGB-D using mAP_{25} (§B.2). Depth Anything V2 lifting is retained in all variants.

R50-FPN	DINOv2	I	I+D+P
–	–	26.4	66.2
✓	–	27.7	66.9
✓	✓	**31.2**	**68.6**

B.3 Additional Qualitative Results and Failure Cases

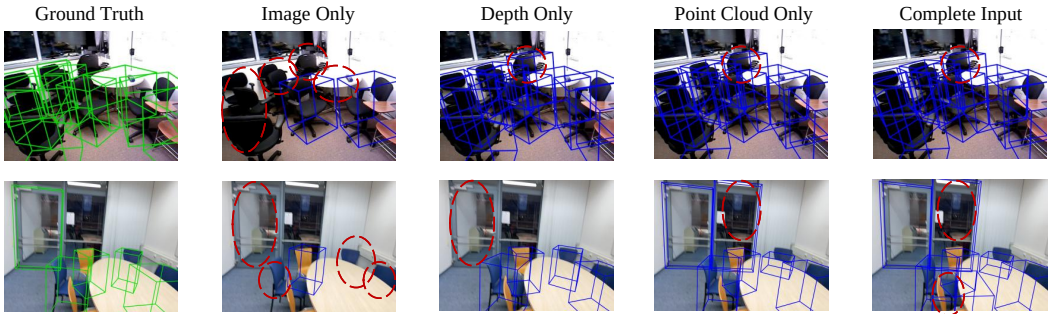

Figure 5: **Additional Qualitative Results and Failure Cases** on SUN RGB-D (Song et al., 2015) (top) and TriScan3D (bottom) (§B.3). Differences are highlighted with red dashed circles ◌.

Fig. 5 illustrates representative limitations under different input configurations. I-only predictions remain relatively weak on both datasets and miss several objects. D-only generally improves detection coverage but can still omit instances, such as the door in the bottom TriScan3D scene. With depth, point-cloud, or complete inputs, GEMI3D can also detect plausible objects missing from the ground-truth annotations, including heavily occluded chairs in both scenes. Such predictions are counted as false positives despite being visually reasonable. Finally, in the bottom scene, the floor-to-ceiling glass panel beside the door is misclassified as a door, revealing a remaining semantic ambiguity.

C TriScan3D Benchmark Construction

C.1 Motivation and Benchmark Positioning

Modality-independent indoor 3D object detection remains underexplored. Standard indoor detection benchmarks and evaluation protocols are primarily organized around a fixed input configuration, such as point clouds, RGB-D observations, or a predefined combination of images and geometry. Existing missing-modality studies generally evaluate robustness by removing one of the expected inputs from a fixed sensor pool, but they do not provide a unified protocol for testing whether the same detector can operate across arbitrary RGB, depth, and point-cloud subsets. Consequently, existing benchmarks do not directly measure the central capability considered in this work: reusing one detector when the available input modalities change.

ScanNet++ provides a suitable foundation for constructing such a protocol because it jointly offers posed commodity RGB-D sequences, high-fidelity reconstructed scene geometry, camera calibration, and scene-level semantic instance annotations (Yeshwanth et al., 2023). These assets provide complementary sources of geometric information: frame-level depth observed by the handheld RGB-D sensor and visible geometry derived from the independently reconstructed scene. However, the original ScanNet++ benchmarks focus on novel-view synthesis and semantic or instance-level 3D scene understanding rather than frame-level 3D bounding-box detection. Its raw annotations therefore cannot be directly used to evaluate modality-independent detectors.

A recent detection-oriented adaptation of ScanNet++ derives per-frame 3D boxes from its instance annotations and evaluates conventional RGB, RGB-D, and point-based detectors (Lazarow et al., 2025a). That setting is designed primarily to assess detection accuracy under each method's native, fixed input configuration and employs class-agnostic evaluation. In contrast, our objective is to isolate modality independence. We therefore construct category-aware frame-level samples in which RGB images, depth maps, and point clouds are spatially aligned but retained as distinct modality paths. In particular, the point-cloud path is generated by raycasting the reconstructed scene mesh rather than trivially back-projecting the current depth map. This separation prevents the depth and point-cloud modalities from becoming duplicate representations of the same frame-level observation. Each sample is associated with a common set of oriented 3D boxes, allowing the same detector to be evaluated consistently under all seven non-empty subsets of $\{I, D, P\}$.

Our benchmark provides a task-specific evaluation protocol for modality-independent indoor 3D detection. We will release the selected frame identifiers, processed annotations, class mappings, data-generation scripts, and evaluation configurations. By making different input subsets comparable under shared samples, labels, splits, and metrics, we hope this benchmark lowers the barrier to studying modality-independent 3D detectors and encourages broader attention to sensor-configuration generalization in indoor 3D perception.

C.2 Construction Overview

Starting from the raw ScanNet++ v2 release, we convert the iPhone RGB-D sequences and scene-level annotations into frame-level samples containing an RGB image, a depth map, a reconstruction-derived visible point cloud, camera parameters, and oriented 3D bounding boxes. The construction pipeline consists of coverage-aware frame selection (§C.3), frame-level annotation construction (§C.4), quality filtering and alignment verification (§C.5), scene-disjoint splitting and final statistics (§C.6), class and alias filtering (§C.7), and point-cloud generation (§C.8). The final benchmark applies core-class filtering, manually verified aliases, alignment filtering, and mesh-raycast point-cloud generation. Algorithm 2 summarizes the complete pipeline.

C.3 Frame Selection

For each scene, we first generate candidate frames with valid camera calibration and discard frames with nearly empty depth maps using a loose valid-depth sanity check. We remove near-duplicate viewpoints using a minimum camera translation of 0.10 m or a minimum camera rotation of $15°$. For each remaining candidate, we estimate its geometry coverage by back-projecting the depth frame into 3D, voxelizing the points at 0.05 m, and subsampling them with a point stride of 8.

Table 8: **Key Parameters for TriScan3D Frame Selection** (§C.3).

Parameter	Value
Minimum camera translation	0.10 m
Minimum camera rotation	$15°$
Valid-depth sanity threshold	0.01
Coverage voxel size	0.05 m
Coverage point stride	8
Target average frames per scene	32
Budget weighting	$\sqrt{N_{\text{cand}}}$
Per-scene budget range	$[8, 60]$
Coverage retention threshold	0.98

We then allocate a frame budget K for each scene. Specifically, we target an average of 32 selected frames per scene within each split/shard, weight the per-scene budget by the square root of the number of candidate frames, and clip the budget to $K \in [8, 60]$. Given K, we select frames in two stages. First, we greedily add frames that maximize the marginal number of newly covered scene voxels until the selected subset reaches at least 98% of the geometry coverage achieved by a uniform K-frame selection. If the budget is not yet exhausted, we fill the remaining slots using a soft object-quality criterion computed from boxes that pass the ground-truth quality filters listed in Table 9.

For a valid box o observed in a candidate frame f, we compute a class-agnostic object-quality score

$$q_{f,o} = q_{\text{img}} + q_{\text{area}} + q_{\text{pts}} + q_{\text{sup}} + q_{\text{ext}} + b_{\text{ctr}}.$$

Here, q_{img} is the projected in-image ratio, $q_{\text{area}} = \min(\sqrt{a/0.02}, 1)$ uses the projected area fraction a, $q_{\text{pts}} = \min(\log(1+n)/\log(2001), 1)$ uses the number of supporting points n, $q_{\text{sup}} = \min(s/0.15, 1)$ uses the support XY area ratio s, and $q_{\text{ext}} = \min(e/0.50, 1)$ uses the maximum

Algorithm 2 Pseudo-code of TriScan3D Benchmark Construction (§C.2).

```
# scenes: ScanNet++ v2 RGB-D sequences, poses, meshes, and 3D instances
# aliases/classes: verified label map and the ten retained object classes

B = {"train": [], "val": []}
# Scene-disjoint split (§C.6)
splits = scene_disjoint_split(scenes, n_train=723, n_val=182)

for split, scene_set in splits.items():
  for scene in scene_set:
    # Filter candidate viewpoints (§C.3)
    F = valid_frames(scene.rgbd, min_valid_depth=0.01)
    F = suppress_near_duplicates(F, min_trans=0.10, min_rot=15)

    # Build visible geometry and frame-level boxes (§C.4)
    voxels = {}
    valid_boxes = {}
    for f in F:
      points = backproject(f.depth, f.intrinsics)
      voxels[f] = voxelize(points[::8], voxel_size=0.05)
      boxes = transform_boxes(scene.instances, f.pose)
      valid_boxes[f] = quality_filter(
          boxes, f, min_points=20, min_area=400,
          min_in_image=0.35, max_area_frac=0.65,
          min_xy_support=0.03, min_extent=0.08)

    # Select complementary viewpoints by coverage (§C.3)
    K = scene_budget(len(F), target_average=32, limits=(8, 60))
    reference = uniform_selection(F, K)
    selected = greedy_coverage(
        F, voxels, K, target=0.98 * coverage(reference))
    selected = fill_by_object_gain(
        selected, F, valid_boxes, K, tie_breaker=voxels)

    for f in selected:
      # Verify RGB/depth/pose alignment (§C.5)
      mesh_depth = raycast_depth(scene.mesh, f.camera)
      if depth_consistency(mesh_depth, f.depth) < 0.30:
        continue

      # Merge aliases and retain benchmark classes (§C.7)
      boxes = remap_and_filter(
          valid_boxes[f], aliases=aliases, classes=classes)
      if len(boxes) == 0:
        continue

      # Generate the independent point-cloud path (§C.8)
      points = raycast_visible_mesh(
          scene.mesh, f.camera, n_points=100000,
          color_source="reconstruction")
      points = downsample(points, n_points=49152)

      B[split].append(FrameSample(
          rgb=f.rgb, depth=f.depth, points=points,
          camera=f.camera, boxes=boxes))

return B
```

support extent ratio along the box x and y axes. We set $b_{\text{ctr}} = -0.15$ when the box has centroid-depth inconsistency and 0 otherwise. Given the currently selected frame set S, the object-quality gain of a candidate frame is

$$G_{\text{obj}}(f \mid S) = \sum_{o \in \mathcal{O}(f)} \max\big(0, q_{f,o} - \max_{f' \in S} q_{f',o}\big),$$

where $\mathcal{O}(f)$ denotes the valid boxes in frame f, with the inner maximum defined as zero when $S = \emptyset$. This gain favors frames that provide a better observation of an object than any already selected frame. Ties are broken by geometry coverage gain.

Table 8 lists the key frame-selection parameters. The motion thresholds suppress near-duplicate views, while the voxel size and point stride define geometry coverage. Scene budgets scale with the square root of the candidate count, are clipped to $[8, 60]$, and target 32 frames per scene on average. Selection stops after retaining 98% of the attainable coverage.

C.4 Frame-Level Ground Truth

The 3D boxes are derived from ScanNet++ scene-level object annotations. For each selected frame, scene-level boxes are transformed into the current camera frame using the camera pose and are stored in a SUN RGB-D-style upright coordinate system. We additionally project the 3D boxes to the RGB/depth image planes to check visibility and projection quality. Each sample is exported in an MMDetection3D-style information file with the frame identifier, scene identifier, point-cloud path, image path, camera extrinsics, camera intrinsics, axis-alignment matrix, and object instances.

C.5 Quality Filtering

We apply the ground-truth quality filters summarized in Table 9 when constructing frame-level annotations. After projecting each scene-level box into a selected frame, we discard boxes with insufficient depth/point support, an extremely small 2D projection, too little in-image projection, abnormal projected area, weak depth support in the projected region, or insufficient support along the box extent. We do not use centroid-depth inconsistency as a hard filter, because an object's center may be invisible even when the object itself is clearly observable. These filters are designed to remove clearly unreliable frame-level boxes rather than retain only fully visible objects.

During visual inspection, we also found a small number of frames with obvious RGB/depth/pose/box misalignment. We therefore add an alignment filter based on mesh-raycast and depth-consistency checks. Specifically, we raycast the reconstructed scene mesh into the current camera view and compare the rendered mesh depth with the observed depth map. Frames whose consistent valid-depth pixel ratio is lower than 0.3 are removed.

Table 9: **Key Parameters for TriScan3D Ground-Truth Quality Filtering** (§C.5).

Parameter	Value
Minimum in-box points per instance	20
Minimum projected box area	400 px
Minimum projected in-image ratio	0.35
Maximum projected area fraction	0.65
Minimum support XY area ratio	0.03
Minimum support extent ratio	0.08

C.6 Scene-Disjoint Split

The benchmark is split at the scene level rather than at the frame level. This prevents different views of the same room from appearing in both training and validation sets. We start from 905 valid scenes and assign 723 scenes for training and 182 scenes for validation. This fixed scene split is preserved throughout all subsequent frame-level processing. The resulting benchmark statistics are shown in Table 10.

Table 10: **Final TriScan3D Statistics** after the full filtering pipeline and nonempty-frame pruning (§C.6).

Split	Frames	GT Boxes	GT/Frame
Train	14,862	56,327	3.79
Val	3,668	14,525	3.96
All	18,530	70,852	3.82

C.7 Class Set and Alias Merging

We evaluate ten frequent and reliable indoor categories: chair, table, monitor, cabinet, trash can, door, bookshelf, bed, sink, and sofa. The class set is designed to keep the benchmark sufficiently trainable while avoiding extremely sparse or ambiguous categories. Table 11 reports the final class distribution. Class aliases are merged as listed in Table 12. We do not use substring-based automatic merging; for example, *table lamp* is not merged into table and *cabinet door* is not merged into cabinet.

Table 11: **Class Distribution of TriScan3D** (§C.7).

Class	Train GT	Val GT	All GT
Chair	21,085	5,583	26,668
Table	10,730	2,813	13,543
Monitor	6,449	1,643	8,092
Cabinet	6,888	1,705	8,593
Trash can	2,097	486	2,583
Door	5,074	1,262	6,336
Bookshelf	986	237	1,223
Bed	722	233	955
Sink	1,327	319	1,646
Sofa	969	244	1,213

Table 12: **Manually Verified Class-Alias Whitelist** used for TriScan3D core-class filtering (§C.7).

Target class	Raw labels mapped to the target class
Chair	chair, office chair
Table	table, desk, coffee table, dining table, side table, end table, bedside table, office table, computer desk, office desk, classroom table, round table, conference table, study table
Monitor	monitor, computer monitor
Cabinet	cabinet, kitchen cabinet, storage cabinet, file cabinet, bedside cabinet, bathroom cabinet, open cabinet, lab cabinet, tool cabinet, TV cabinet, electrical cabinet, electric cabinet, computer cabinet, toilet cabinet, office cabinet, bath cabinet, mirror cabinet, first aid cabinet, wall cabinet, laboratory cabinet, small cabinet, rolling cabinet, filing cabinet, display cabinet
Trash can	trash can, trash bin, dustbin, outdoor trash can, garbage bin, dust bin, trashcan, paper bin, trash container
Door	door, sliding door, shower door, French door, double door, glass door
Bookshelf	bookshelf, book shelf, low bookshelf, small bookshelf
Bed	bed
Sink	sink, kitchen sink, bathroom sink, wash basin, washbasin
Sofa	sofa, couch, L-shaped sofa, floor couch, single couch

C.8 Point-Cloud Sources

The depth and point-cloud paths use distinct geometric sources. The depth path internally back-projects the current iPhone depth map using its camera parameters. In contrast, the point-cloud input is generated independently by raycasting the reconstructed ScanNet++ scene mesh from the current view. We retain visible surfaces, generate approximately 100k points, and downsample them to 49,152 points. Their color is inherited from the reconstruction rather than the current RGB frame. This design avoids making the depth and point-cloud paths duplicate representations of the same frame-level observation.

C.9 TriScan3D Limitations

TriScan3D is constructed from ScanNet++ scene-level annotations rather than being an official ScanNet++ detection task. Its quality depends on camera-pose accuracy, visibility estimation, and the transformation from scene-level boxes to frame-level boxes. The alignment filter removes a small number of obviously misaligned samples, but residual depth noise, occlusion, and partial visibility can still affect training and evaluation. Finally, mesh-raycast point clouds and frame-level depth maps differ in geometry source, sampling density, and noise characteristics, introducing a source gap between the two modalities.

D Limitations

MCHA currently selects the reference path using a fixed $P \succ D \succ I$ priority, which does not account for sample-specific input quality when a higher-priority path is degraded. Quality-aware routing, which assigns object queries to expert decoders according to modality feature quality (Park et al., 2025), suggests a learnable alternative to this fixed priority. In addition, modality dropout simulates complete path absence but does not explicitly cover partial corruption within an available modality. Existing sensor-corruption protocols, including point loss, view noise, and occlusion (Ge et al., 2023), offer a direct way to augment training with partially degraded inputs. Future work will explore these directions to improve robustness under unreliable sensor inputs.

E Social Impact

By allowing one checkpoint to operate across sensor configurations, GeMI3D may reduce repeated model training and simplify the deployment of indoor 3D detection on robotic and assistive systems. At the same time, RGB-D and point-cloud observations can reveal the layout, contents, and activities of private indoor spaces. Detection errors or uneven performance across environments, sensors, and object categories may also propagate to downstream navigation, manipulation, or monitoring. Modality independence should therefore not be interpreted as guaranteed reliability under sensor failure. Deployment in safety- or privacy-sensitive settings should use consent-aware data

collection, appropriate access controls, domain-specific validation, monitoring for degraded inputs and uncertain predictions, and human oversight where consequences are substantial.