OpenBox: Annotate Any Bounding Boxes in 3D

In-Jae Lee[1] **Mungyeom Kim**[1] **Kwonyoung Ryu**[2] **Pierre Musacchio**[1] **Jaesik Park**[1]

[1]Seoul National University [2]POSTECH

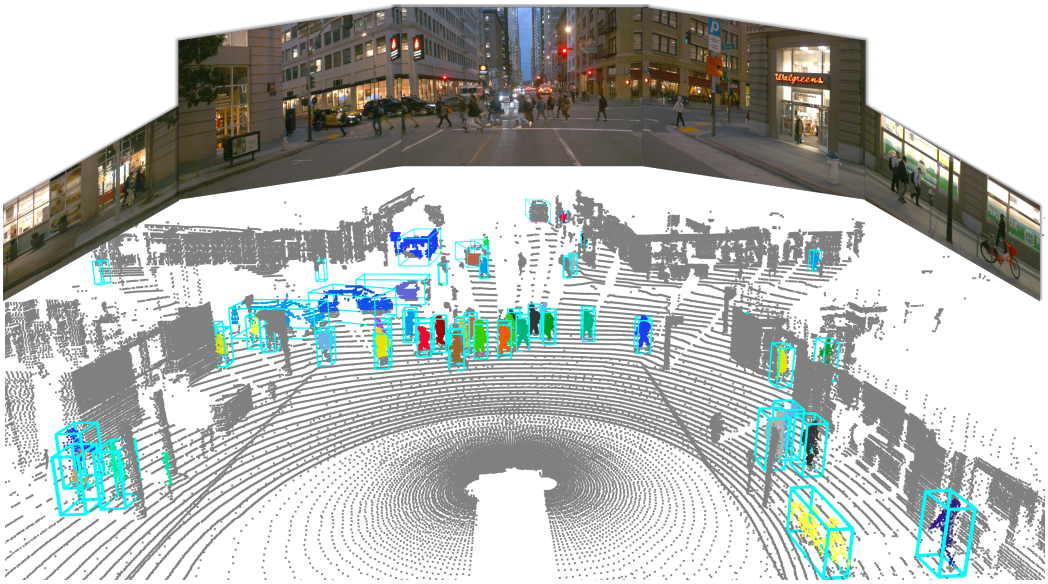


Figure 1: We introduce **OpenBox**, which utilizes a 2D vision foundation model to annotate 3D bounding boxes automatically. It annotates instances of vehicles, pedestrians, and cyclists. We demonstrate it with Waymo Open Dataset [33]. Best viewed in color and zoomed in.

Abstract

Unsupervised and open-vocabulary 3D object detection has recently gained attention, particularly in autonomous driving, where reducing annotation costs and recognizing unseen objects are critical for both safety and scalability. However, most existing approaches uniformly annotate 3D bounding boxes, ignore objects' physical states, and require multiple self-training iterations for annotation refinement, resulting in suboptimal quality and substantial computational overhead. To address these challenges, we propose **OpenBox**, a two-stage automatic annotation pipeline that leverages a 2D vision foundation model. In the first stage, OpenBox associates instance-level cues from 2D images processed by a vision foundation model with the corresponding 3D point clouds via context-aware refinement. In the second stage, it categorizes instances by rigidity and motion state, then generates adaptive bounding boxes with class-specific size statistics. As a result, OpenBox produces high-quality 3D bounding box annotations without requiring self-training. Experiments on the Waymo Open Dataset (WOD), the Lyft Level 5 Perception dataset, and the nuScenes dataset demonstrate improved accuracy and efficiency over baselines.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1 Introduction

3D object detection has become increasingly important across a wide range of applications, including autonomous driving [16, 21, 23, 42], robotics [29, 41], and virtual reality [15, 37]. In autonomous driving, it provides essential inputs for motion prediction that, in turn, inform path planning and vehicle control. As a result, the accuracy of 3D object detection is directly tied to the overall safety and reliability of the system. While recent advances in deep learning have significantly improved detection performance, most existing frameworks [16, 21, 23, 42] remain limited to a fixed set of object categories and are heavily reliant on large-scale, human-annotated datasets. This closed-set assumption becomes particularly problematic in open-world autonomous driving scenarios. In such settings, the system must be able to detect a wide range of object types, including rare or previously unseen instances.

Integrating open-vocabulary detection enables models to recognize arbitrary categories based on semantic descriptions, thereby overcoming the limitations of a fixed label space. In the 2D image domain, open-vocabulary perception has been accelerated by the availability of large-scale image-text paired datasets and the emergence of vision foundation models. These models demonstrate strong generalization capabilities across tasks such as classification [30, 34], detection [22, 24, 48], and segmentation [14, 19, 31].

Despite the advances above, creating large-scale annotated 3D datasets remains a major bottleneck. Unlike 2D images, LiDAR point clouds provide precise geometric structure but lack rich semantic context, making them difficult to align with text-based supervision and challenging to annotate manually. To address these limitations, several unsupervised methods [40, 43, 45, 46] have been proposed. These typically follow a pipeline in which ground points are removed from raw LiDAR scans, spatial clustering is applied to extract object instances, and scene flow [1, 25, 26] or a persistence point score (PP Score) [40, 43, 46] is used to identify motion states. The resulting 3D bounding boxes are then refined through multiple rounds of self-training [43, 45, 46] or sampling strategies [46]. However, these methods generally do not consider physical properties of instances for box annotation, which leads to low-quality boxes and remains computationally expensive due to their iterative refinement. More recently, several works [18, 26, 46] incorporate image semantics to assist automatic annotation. Nevertheless, [46] fuses modality-specific 3D bounding boxes at the output level without geometric alignment, and [18] does not fully leverage visual cues to improve 3D annotation quality.

This paper proposes a two-stage pipeline, **OpenBox**, that automatically annotates 3D bounding box for arbitrary classes. Our approach leverages high-quality instance-level information from 2D vision foundation models (e.g., Grounding DINO [22], SAM2 [31]) as supervisory signals, thereby reducing the cost and time of manual annotation. In the first stage (Cross-modal Instance Alignment), 2D instance-level information is unprojected onto the 3D point cloud. To address noisy or incomplete instance point clouds caused by the vision foundation model, we apply a context-aware refinement step to enhance the quality of instance-level points. Subsequently, the refined instances are categorized into three physical types: static rigid, dynamic rigid, and deformable. Based on category-specific object size statistics, we generate 3D bounding boxes for each category. Specifically, we construct a mesh for static rigid objects using the Signed Distance Function (SDF) [36] and filter out noise points through majority voting. We then further refine the bounding box via 3D-2D IoU alignment and visibility. We conduct experiments on the WOD [33], Lyft [12], and nuScenes [2] datasets. Qualitative results on real-world data show that our method produces high-quality and robust 3D annotations, as illustrated in Fig. 1.

Our contributions are summarized as follows:

- We propose OpenBox, a novel automatic annotation pipeline that requires only synchronized ego poses, images, and LiDAR point clouds, without self-training.
- To improve point cloud quality, we introduce a two refinement process: context-aware refinement and surface-aware noise filtering based on the SDF. We also generate bounding boxes adaptively based on the physical types of instances.
- Training with OpenBox-generated annotations achieves **70.49**% AP_{3D} for the vehicle class of the WOD [33] at 0.5 IoU. On the Lyft dataset [12], OpenBox outperforms state-of-the-art approach by **+19.94**% AP_{3D} when directly compared to human annotation boxes.

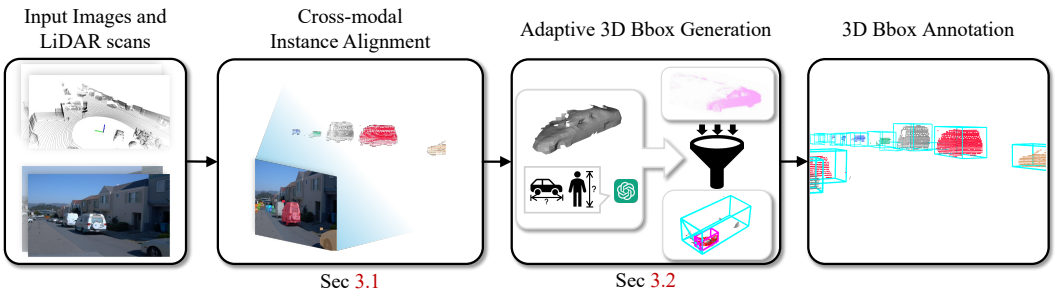


Figure 2: **Pipeline Overview of OpenBox.** With time-synchronized, unlabeled images and LiDAR scans, cross-modal instance alignment (Sec 3.1) associates 2D instance cues with corresponding point clouds. Adaptive 3D bounding box generation (Sec 3.2) independently chooses the most suitable fitting strategy for each instance, yielding high-quality 3D bounding boxes.

2 Related Work

2.1 Open-vocabulary 3D Object Detection

The rapid progress of 2D vision foundation models [11, 20, 22, 30, 34] has spurred active research in open-vocabulary 3D object detection. Most existing methods [8, 26, 44] focus on accurately aligning 2D visual cues with 3D spatial information. UP-VL [26] enhances the MI-UP [25] auto-labeling pipeline by incorporating OpenSeg [11] to generate semantically aligned amodal 3D bounding boxes for open-vocabulary transfer. Additionally, it introduces a loss function that facilitates 2D-3D mapping, allowing the model to learn point-level features guided by distillation loss. Find and Propagate [8] generates frustum-shaped 3D proposals using 2D open-vocabulary detectors [20, 24], followed by multi-view alignment and density-based filtering to improve the detection of distant objects. OpenSight [44] lifts 2D bounding boxes obtained from Grounding DINO [22] into 3D space to enable generic object perception followed by semantic interpretation. Whereas prior work redesign 3D object detectors for the open-vocabulary setting, we focus on annotating the dataset to allow open-vocabulary 3D detection.

2.2 Unsupervised 3D Object Detection

LiDAR-based. Most unsupervised 3D object detection methods [1, 25, 40, 43, 45] solely rely on LiDAR point clouds to perform automatic annotation. Common pipelines first estimate motion states using PP score [40, 43] or scene flow [1, 25], and then perform ground removal followed by point cloud clustering [3, 7]. Except for [40], which utilizes class-wise size statistics, these limitations primarily stem from the lack of semantic information inherent in LiDAR, especially when compared to RGB images. As a result, they generally train and evaluate models in a *class-agnostic* manner. Moreover, CPD [40] incurs additional computational overhead by jointly using dense prototypes (CProto) and downsampled point clouds, resulting in significantly longer training time.

Multi-modal based. Several approaches [18, 26, 46] use 2D vision foundation models to incorporate image information. LiSe [46] integrates 3D bounding boxes obtained from the LiDAR branch (via [43]) and the image branch (via [14, 22, 38]) in a distance-aware manner. UNION [18] leverages appearance from 2D images to cluster and distinguish between mobile and immobile objects.

Unlike these approaches, which depend on iterative self-training [43, 46] and do not consider physical properties of instance [18, 26, 40, 43, 46], our method is designed to produce physical-state-specific annotations and to alleviate the need for iterative refinement.

3 Method

This section explains the role and design choices of each module in our proposed automatic 3D bounding box annotator, **OpenBox**. As shown in Fig. 2, our system consists of two main stages: (1) Cross-modal Instance Alignment and (2) Adaptive 3D Bounding Box Generation.

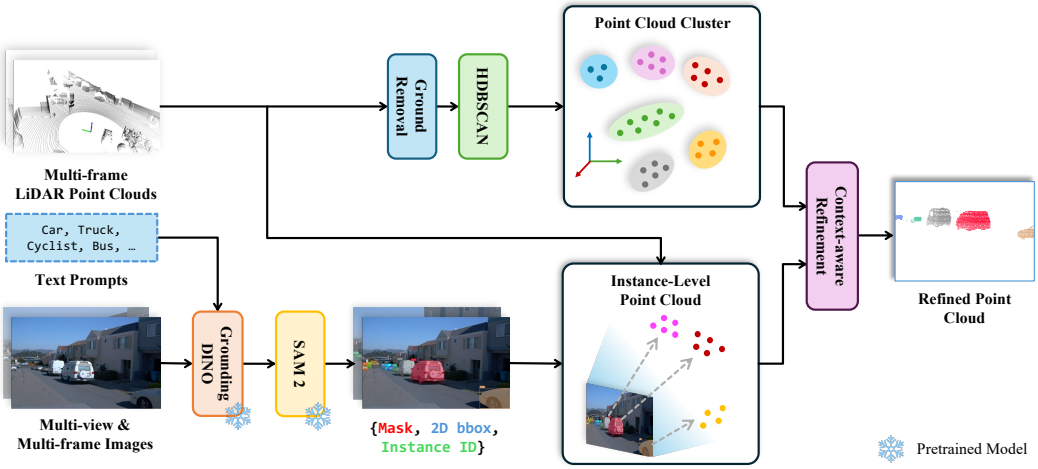


Figure 3: **Cross-modal Instance Alignment.** To obtain a refined point cloud, the pipeline generates two complementary point cloud clusters. The LiDAR (upper) branch removes ground points and applies HDBSCAN [3] to produce coarse 3D clusters. The image-LiDAR (lower) branch uses Grounding DINO [22] followed by SAM2 [31] to generate 2D instance masks. This information is unprojected to a point cloud. Context-aware refinement fuses the two proposals, discarding noisy points and incorporating adjacent points from the point cloud cluster, yielding a refined per-object point cloud.

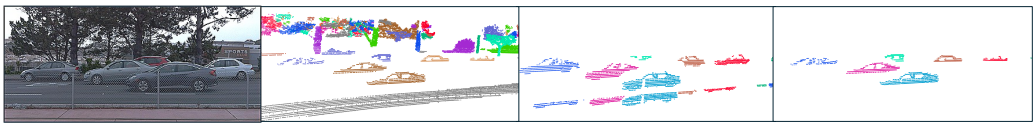
3.1 Cross-modal Instance Alignment

Instance-level Feature Extraction. As shown in Fig. 3, to leverage the strong capabilities of vision foundation models [22, 31] trained on large-scale datasets, we establish a mapping between a 2D image $I_j^{(t)} \in \mathbb{R}^{3\times H\times W}$ which include N instance IDs $\mathcal{T}_j^{(t)}$, c class labels $\mathcal{C}_j^{(t)}$, N segmentation masks $\mathcal{M}_j^{(t)}$, and 3D to 2D mapping function $\mathbf{\Pi}_j$ and a corresponding 3D point cloud $\mathcal{P}^{(t)} \in \mathbb{R}^{M\times 3}$ captured at time t from the j-th camera. These information are obtained using a 2D detection network Ψ [22] and a segmentation network Φ [31] as follows:

$$\mathcal{B}_j^{(t)}, \mathcal{C}_j^{(t)} = \Psi(I_j^{(t)}), \quad \mathcal{M}_j^{(t)}, \mathcal{T}_j^{(t)} = \Phi(\mathcal{B}_j^{(t)}, txt), \quad \mathcal{V}_j^{(t)} = \mathbf{\Pi}_j(\mathcal{P}^{(t)}, I_j^{(t)}), \tag{1}$$

where H, W, $\mathcal{B}_j^{(t)} \in \mathbb{R}^{N\times 4}$, $\mathcal{V}_j^{(t)} \in \mathbb{R}^{N\times 2}$ and txt denote the height, width of the image $I_j^{(t)}$, 2D bounding boxes, pixel coordinates of $\mathcal{P}^{(t)}$ projected to $I_j^{(t)}$ and text prompts respectively. By associating the image pixel with a point cloud which is projected on to mask, we can obtain the instance-level point clouds $\mathcal{F}_j^{(t)} = \{\mathcal{F}_{ij}^{(t)} \in \mathbb{R}^{M'\times 6}\}_i$. Here, $\mathcal{F}_{ij}^{(t)}$ is the i-th instance-level point cloud, which contains 3D coordinate (x, y, z), semantic class, instance presence, and instance ID. However, the boundaries of the masks obtained from [31] are imprecise, due to calibration errors, so directly unprojecting them into 3D can result in noisy point clouds. To mitigate this issue, we adopt the adaptive erosion proposed in [13], which erodes masks based on object size to eliminate boundary noise while preserving instance structure. For convenience, we omit the subscripts t and j from this point onward.

Context-aware Refinement. As shown in Fig. 4-(c), LiDAR points are often projected onto background objects (*e.g.* guardrail and wall) that occlude the actual foreground instance, resulting in inaccurate unprojection. These noisy points, located outside the true object region, tend to yield 3D bounding boxes that are improperly scaled. To address this issue, we refine the unprojected instance-level point clouds $\mathcal{F}$. We perform majority voting within clustered regions $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_{N'}\}$, where each cluster $\mathcal{R}_k \in \mathbb{R}^{m_k\times 3}$ is obtained from the ground-removed raw LiDAR point cloud $\mathcal{P}$ using HDBSCAN [3], following ground segmentation based on [17]. For each segment $\mathcal{R}_k$, we compare it with all instance-level point clouds $\mathcal{F}_i$ and compute bidirectional proximity-based inclusion ratios. Specifically, we determine the proportion of points in $\mathcal{R}_k$ that overlap with any point in $\mathcal{F}_i$, and vice versa. If mutual overlap between the two clusters is sufficient, the cluster $\mathcal{R}_k$ is assigned the instance ID i and retained; otherwise, it is discarded. This process can be formulated as follows:



(a) Reference Image (I) (b) Point Cloud Cluster ($\mathcal{R}$) (c) Instance-level Point Cloud ($\mathcal{F}$) (d) Refined Point Cloud ($\mathcal{F}_{\text{ref}}$)

Figure 4: **Context-aware Refinement.** (a) Reference image. (b) Point cloud clusters $\mathcal{R}$ after using HDBSCAN [3] on ground-removed LiDAR point cloud. (c) Noisy instance-level point clouds $\mathcal{F}$. (d) Result of the Context-aware refinement $\mathcal{F}_{\text{ref}}$.

$$\frac{|\{p \in \mathcal{R}_k \mid \text{dist}(p, \mathcal{F}_i) < \delta\}|}{|\mathcal{R}_k|} > \alpha, \quad \frac{|\{p \in \mathcal{R}_k \mid \text{dist}(p, \mathcal{F}_i) < \delta\}|}{|\mathcal{F}_i|} > \beta \quad \Rightarrow \quad \mathcal{R}_k \leftarrow i, \tag{2}$$

where $|\cdot|$ denotes the cardinality of a set, and $\text{dist}(p, \mathcal{F}_i) < \delta$ holds if and only if there exists $f \in \mathcal{F}_i$ such that $\|p - f\|_2 < \delta$.

3.2 Adaptive 3D Bounding Box Generation

Most prior methods [13, 43, 46] generate boxes without considering the physical properties of individual objects. This often leads to inaccurate localization and reduces the consistency of the data used to train 3D object detection networks. To address this issue, we propose an adaptive box-generation strategy that accounts for the physical types of each instance.

Static & Dynamic Points Decomposition. The refined instance-level LiDAR point clouds $\mathcal{F}_{\text{ref}}$ obtained in the previous step remain sparse. Yet, by aggregating consecutive point cloud frames to a global coordinate system, the instance-level point cloud can be significantly densified. Incorporating point clouds from dynamic objects is challenging, as it may introduce motion artifacts [40]. Thus, we use the PP score [43] to estimate the ephemerality of each point in the refined point cloud.

Initial Bounding Box Generation. Empirically, we found that categorizing each instance based on its physical properties leads to better performance. In particular, we divide instances $\mathcal{F}_{\text{ref}}$ into three types: rigid and static $\mathcal{F}^S_{\text{ref}}$, rigid and dynamic $\mathcal{F}^D_{\text{ref}}$, and deformable $\mathcal{F}^{deform}_{\text{ref}}$. For each type, we generate a corresponding 3D bounding box. We use ChatGPT [27] to determine the object type based on the given semantic class to distinguish between rigid and deformable objects. Then, using this classification in conjunction with motion cues estimated via the PP score [43], we generate an appropriate 3D bounding box for each instance. We initially generate a bounding box for all three object types using an approach [47] that maximizes the closeness of points to edges. However, due to the sparsity of the point cloud and occlusion, the resulting bounding box may underestimate the actual object size. To address this, we use ChatGPT [27] to retrieve the typical size of the object class in terms of length, width, and height. If any of the initial bounding box dimensions are smaller than 80% of the typical size, we adjust the box size as described in the following sections.

Handling Static & Rigid Instances. Although we densify the aggregated static instance point cloud $\mathcal{F}^S_{\text{ref}}$, it still contains noise due to limitations of the context-aware refinement. To suppress it, we apply a surface-aware filtering method based on proximity voting over mesh vertices. Specifically, we reconstruct a mesh surface $\mathbf{S}$ from the point cloud using SDF [36]. For each vertex $v \in \mathbf{S}$, we identify nearby foreground and background points using the proximity function $\mathcal{P}_C(\cdot, v)$ defined in Eq.3, where τ denotes the distance between the point and the vertex from the mesh. We retain vertices where foreground associations dominate, forming the refined surface $\mathbf{S}_{\text{ref}}$ as defined in Eq. 4. The final refined static point cloud $\mathcal{F}^{S,(2)}_{\text{ref}}$ is then constructed by collecting all foreground points near the filtered surface vertices.

$$\mathcal{P}_C(\mathcal{F}, v) = \{p \in \mathcal{F} \mid \|p - v\|_2 < \tau\}, \tag{3}$$

$$\mathbf{S}_{\text{ref}} = \left\{v \in \mathbf{S} \;\middle|\; |\mathcal{P}_C(\mathcal{F}^S_{\text{ref}}, v)| > |\mathcal{P}_C(\mathcal{F}^C_{\text{ref}}, v)|\right\}, \quad \mathcal{F}^{S,(2)}_{\text{ref}} = \bigcup_{v \in \mathbf{S}_{\text{ref}}} \mathcal{P}_C(\mathcal{F}^S_{\text{ref}}, v). \tag{4}$$

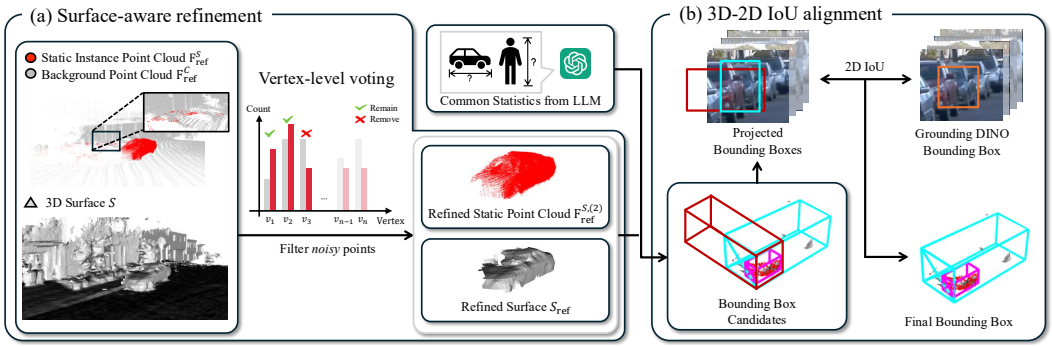


Figure 5: **Handling Static & Rigid Instances.** (a) We filter noisy points in the aggregated static point cloud via vertex-level voting on the reconstructed surface, producing $\mathcal{F}_{\text{ref}}^{S,(2)}$ and $\mathbf{S}_{\text{ref}}$. (b) We then adjust the bounding box using surface normals and statistical priors, and select the final box based on 2D IoU with projected boxes and Grounding DINO [22] boxes.

To refine the initial bounding box, we extract the corresponding instance-level surface mesh $\mathbf{S}_{ins}$ from $\mathbf{S}$. As shown in Fig. 5, if the box is too small, we determine the resizing direction using surface normal vectors, rather than searching over 8 directions as in [13]. We rotate the surface into the ego coordinate system. Then, we compute the dot product between surface normals and the four orthonormal reference vectors to determine which sides of the object are represented. If all four sides are covered, no resizing is needed. Otherwise, we generate two resized box candidates based on statistical priors. This is because the longer side of the initial box cannot be reliably assumed to represent the object's actual length. (Please see Sec. C.3 for more details.) To select the optimal box, we match each 3D candidate with 2D bounding boxes (from Sec. 3.1) using instance ID, project them onto multiple views and time series images, and compute their 2D IoUs. The box with the higher IoU is selected as the final result.

Handling Dynamic & Rigid Instances. In these cases, we rely on point clouds from a single moment in time, which makes it harder to accurately estimate the object's position, orientation, and size. To deal with this problem, OpenBox uses the fact that the orientation of a dynamic object is approximately aligned with the direction of the position difference in adjacent frames. OpenBox first estimates the object's orientation by the direction of the object trajectory associated with 2D tracking IDs. We then rotate the initial bounding box to align it with the estimated orientation angle. After aligning the orientation, we refine the box size. For each face along the X and Y axes, we compute the dot product between the outward surface normal and the LiDAR ray direction at the face center. As shown in Fig. 6, OpenBox extends the box only when the dot product between the ray and face normal is negative. We determine the final box size using standard object-size statistics.

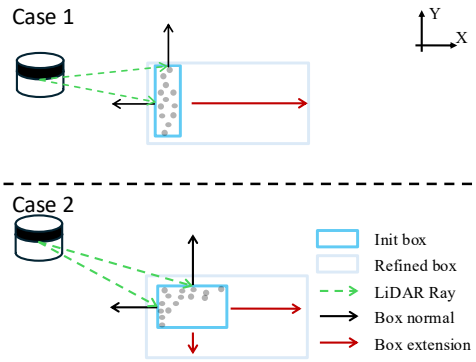


Figure 6: **Visibility-based box extension.** Case 1 has one negative value from the dot product between the ray and the normal, yielding a one-sided extension, whereas Case 2 has two negative values, leading to a two-sided extension.

Handling Deformable Instances. Deformable instances such as pedestrians, animals, or cyclists exhibit articulated or non-rigid motion, causing spatial inconsistencies across frames that often lead to ghosting or distorted geometry when aggregated [4]. Due to their limited surface structure, geometry-based refinement is ineffective. To maintain reliability, we generate bounding boxes from a single frame by tightly fitting the visible region using the closeness-to-edge algorithm [47], which provides robust representations without relying on rigid geometric assumptions.

Table 1: **3D object-detection results on the WOD [33] validation set.** * indicates trained and evaluated in the camera-frustum region, while others use full 360° coverage. † and ‡ denote models trained with CST and CBR from CPD [40], using the training settings given in the next sentence. For †, we flip the **OpenBox** annotations and point clouds to obtain 360° coverage; for ‡, we fill the region outside the camera frustum with CPD annotations. All values denote AP_{3D} at each IoU threshold. **Bold** means best performance, underlined means second-best. Only L1 results are shown here; we provide the full L2 results in the Table 7.

Method	Modality	Vehicle $IoU_{0.5}$ / $IoU_{0.7}$	Pedestrian $IoU_{0.3}$ / $IoU_{0.5}$	Cyclist $IoU_{0.3}$ / $IoU_{0.5}$
CPD* [40]	LiDAR	30.30 / 20.90	14.28 / 11.22	3.47 / **3.08**
OpenBox* (Ours)	LiDAR + Camera	**70.49 / 32.41**	**57.95 / 17.11**	**20.81** / 2.15
DBSCAN [7]	LiDAR	2.32 / 0.29	0.51 / 0.00	0.28 / 0.03
MODEST [43]	LiDAR	18.51 / 6.46	11.83 / 0.17	1.47 / 1.14
OYSTER [45]	LiDAR	30.48 / 14.66	4.33 / 0.18	1.27 / 0.33
CPD [40]	LiDAR	57.79 / 37.40	21.91 / 16.31	5.83 / 5.06
OpenBox† (Ours)	LiDAR + Camera	**66.89** / 39.14	**55.71 / 37.82**	**21.00 / 7.08**
OpenBox‡ (Ours)	LiDAR + Camera	59.09 / **40.68**	39.09 / 28.16	8.27 / 6.23
Human Annotation	-	93.31 / 75.70	87.25 / 77.93	58.84 / 54.88

Table 2: **3D object-detection results on the Lyft [12] validation set.** Following [46], we evaluate in class-agnostic manner at IoU = 0.25, and each value represents AP_{BEV} / AP_{3D}. **Bold** means best performance, underlined means second-best.

Method	Modality	0–30m	30–50m	50–80m	0–80m
MODEST-PP ($T = 0$) [43]	LiDAR	46.4 / 45.4	16.5 / 10.8	0.9 / 0.4	21.8 / 18.0
LiSe ($T = 0$) [46]	LiDAR + Camera	54.5 / 54.0	24.2 / 22.8	1.4 / 1.2	29.2 / 27.5
OpenBox (Ours)	LiDAR + Camera	**62.4 / 62.3**	**56.6 / 50.6**	**19.9 / 19.5**	**49.6 / 43.3**
Human Annotation	-	82.8 / 82.6	70.8 / 70.3	50.2 / 49.6	69.5 / 69.1

4 Experiments

4.1 Experimental Setup

Dataset and Implementation Details. We conduct experiments on Waymo Open Dataset (WOD) [33], Lyft Level 5 Perception Dataset (Lyft) [12] and nuScenes [2]. For 3D object detection networks, we train Voxel R-CNN [6] for WOD [33], PointRCNN [32] for Lyft [12] and CenterPoint [42] for nuScenes [2] with annotation following the baselines [18, 40, 43, 46]. We refer the reader to WOD [33], Lyft [12] and nuScenes [2] for detailed descriptions of the evaluation metrics. Our code is based on OpenPCDet [35] and MMDetection3D [5]. Additional details for training, hyperparameter and network are in the Appendix B.

Baselines. In the WOD [33] benchmark, the state-of-the-art method CPD [40] evaluates the reliability of 3D bounding boxes using the CSS score and constrains network training by jointly learning dense CProtos within those boxes. For the Lyft [12] dataset, LiSe [46] fuses 3D bounding boxes from an image branch [38] and a LiDAR branch [43] based on distance. Finally, for nuScenes [2], UNION distinguishes mobile objects by leveraging visual appearance features extracted with DINOv2 [28]. Since our method in WOD [33] performs annotation only on point clouds that fall within the camera frustum field of view (FOV), we conduct experiments under two different settings.

Experiments Scenarios. We conduct experiments under two scenarios to comprehensively evaluate the quality of our automatic annotations. *Scenario 1* trains a 3D object detector on automatically annotated data and evaluates it on a human-annotated validation dataset. *Scenario 2* directly compares the automatic annotations with the human annotations on the training set.

Table 3: **Annotation performance on Lyft [12] training dataset.** We evaluate our automatically annotated dataset with a human-annotated dataset. Following [46], we evaluate in class-agnostic manner at IoU = 0.25, and each value represents AP_{3D}. **Bold** means best performance.

Method	0–30m	30–50m	50–80m	0–80m
LiSe [46]	17.47	6.87	1.35	6.31
OpenBox (Ours)	**56.62**	**28.10**	**6.47**	**26.25**

Table 4: **3D object-detection results on the nuScenes [2] validation set.** Following [18], we evaluate for 3 classes, and each value represents AP_{3D}. **Bold** means best performance.

Method	Modality	Car	Pedestrian	Cyclist
UNION [18]	LiDAR + Camera	30.1	41.6	0.0
OpenBox (Ours)	LiDAR + Camera	**40.9**	**62.7**	**5.2**

4.2 Main results

Comparison on WOD. Table 1 presents the `LEVEL_1` AP_{3D} results of experiments conducted on the WOD [33] under *Scenario 1*. For a fair comparison with the state-of-the-art method CPD [40], we conduct the experiments for two FOV (Field of View). Under the camera frustum view setting, our method consistently outperforms CPD [40] for vehicle and pedestrian classes, even though CPD incorporates additional training techniques (e.g., CST and CBR) beyond its annotation pipeline. The inferior performance of the cyclist class can be attributed to our use of the prompt "bicycle" in Grounding DINO [22], which often yields undersized bounding boxes compared to those enclosing the entire cyclist. Furthermore, we evaluate two extended settings: (1) applying CPD's [40] training schemes (CST and CBR) to our boxes, and (2) combining our boxes with CPD's [40] and then training with CST and CBR. Both approaches lead to performance improvements across all classes, with particularly notable gains for pedestrian and cyclist categories. We attribute this to a fundamental design difference: CPD [40] annotates only stationary objects, resulting in low recall. Furthermore, it relies on class-agnostic tracking and classifies based on box-size statistics. In contrast, our method identifies the object class using a 2D vision foundation model [22], and generates adaptive bounding boxes that reflect each class's physical properties, resulting in more accurate annotations.

Comparison on Lyft. Table 2 shows the results of class-agnostic 3D object detection on the Lyft [12] dataset under *Scenario 1*, using an IoU threshold of 0.25. To ensure a fair comparison, we evaluate against baseline methods [43, 46] that do not assume multiple traversals and do not apply self-training strategies. Our method demonstrates improved performance in both AP_{BEV} and AP_{3D} across all distance ranges compared to baselines. In particular, for long-range scenarios (50–80m), our method outperforms Lise [46] by +18.5% in AP_{BEV} and +18.4% in AP_{3D}. Furthermore, as shown in Table 3, we evaluate the performance of automatic annotations in the *Scenario 2* environment. Our method consistently outperforms LiSe [46] across all ranges. This performance gap arises because LiSe [46] integrates 3D boxes from the image branch, generated using the method of [38], and from the LiDAR branch, based on [43]. However, neither of these components explicitly considers the physical properties or semantics of the object classes, which limits their precision.

Comparison on nuScenes. As shown in Table 4, we observe performance improvements across all classes under *Scenario 1*. One key reason for the significant gains is that, unlike OpenBox, UNION [18] omits the refinement process for point clouds and 3D bounding boxes. In particular, it does not explicitly consider the camera-lidar calibration error when projecting LiDAR point clouds on DINOv2 [28] feature maps which leads to noise at the boundary of the objects. Additionally, UNION [18] neither resizes nor relocalizes the initial 3D bounding boxes, leading the model to predict suboptimal bounding boxes.

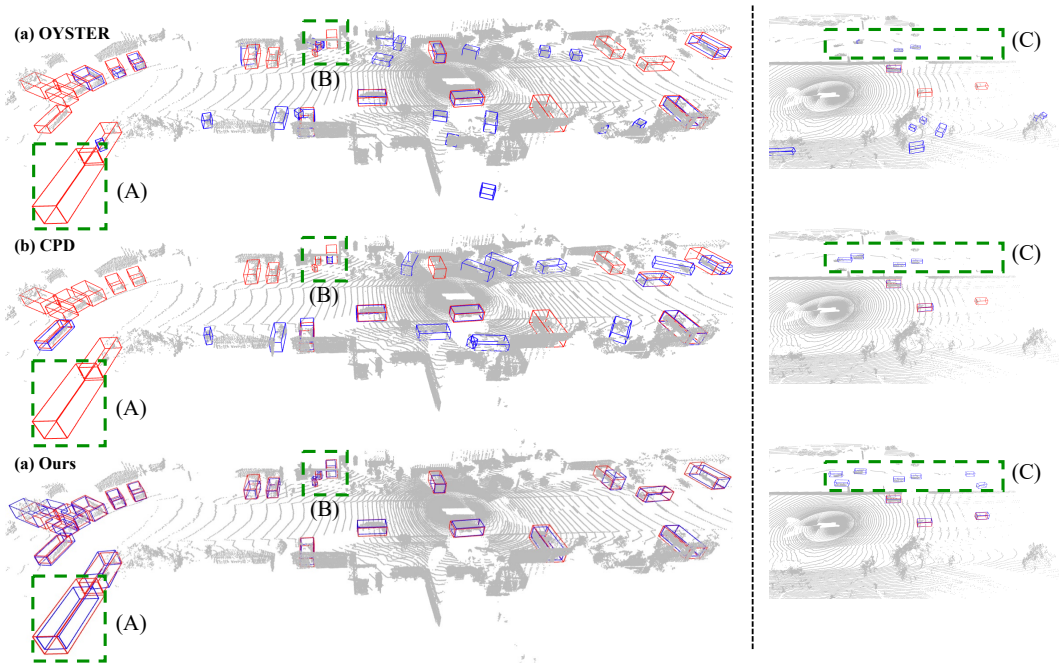


Figure 7: **Comparison of automatic annotation on WOD [33] training set.** Each row compares automatically annotated boxes with human-annotated boxes, while each column corresponds to a different scene. Blue boxes represent the automatically generated boxes, and red boxes indicate the human-annotated boxes. We visualize CPD [40] annotations filtered by a minimum CSS score threshold. Best viewed in color and zoomed in.

Table 5: **Ablation study on the *Vehicle* class for WOD [33] training set.** Surface-aware, Context-aware, and 3D-2D IoU refer to Surface-aware Refinement, Context-aware Refinement, and 3D-2D IoU alignment respectively. All results stand for the Vehicle class using AP_{3D} at IoU = 0.4.

(a) Point-level refinement

Surface-aware	Context-aware	AP_{3D}
✓		30.34
	✓	32.52
✓	✓	**38.65**

(b) Box-level refinement

Visibility-based	3D-2D IoU	AP_{3D}
✓		30.49
	✓	34.71
✓	✓	**38.65**

4.3 Ablation study

To analyze the impact of each module on automatic annotation, we conduct an ablation study under the *Scenario 2*. In Table 5-(a), we observe that applying both point-level refinement modules yields the highest performance. This is because Context-aware Refinement is applied to all instances regardless of their physical properties, whereas Surface-aware Refinement is specifically designed for rigid and static instances. Furthermore, some effects of Surface-aware Refinement are partially covered by Context-aware Refinement, which explains its relatively larger contribution when applied alone. Similarly, Table 5-(b) presents an ablation study focusing on modules that contribute to box-level refinement. The 3D-2D IoU alignment module is designed to resize and relocate boxes for static and rigid instances, while the visibility-based module is applied to dynamic and rigid instances. 3D-2D IoU alignment has a greater overall impact when combined with the visibility-based method for two main reasons: (1) the number of static vehicles in the WOD [33] dataset is significantly larger than that of dynamic vehicles, and (2) the two candidate boxes considered by 3D-2D IoU alignment typically differ by 90°, leading to a more substantial effect on the AP_{3D}.

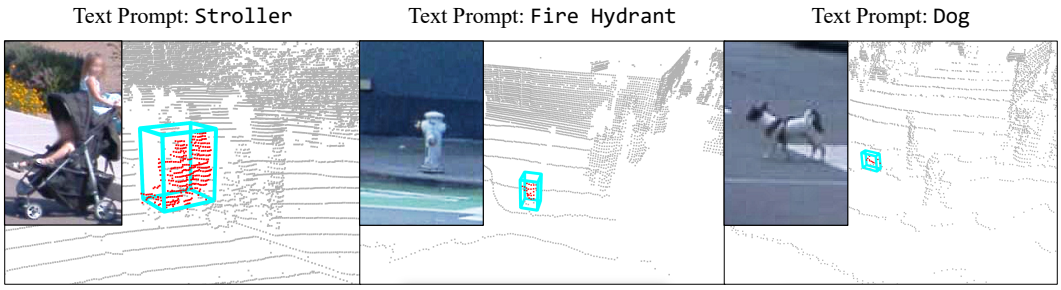


Figure 8: **Our automatic annotation results on novel classes in WOD [33].** In the visualization, red points represent instance-level point clouds, while cyan boxes indicate the automatically generated annotations. Best viewed in color and zoomed in.

4.4 Qualitative Result

As shown in Fig. 7, we compare 3D bounding boxes from automatic and human annotations. Overall, OpenBox shows higher precision and recall compared to the baselines [40, 45]. Region (A) illustrates a static travel trailer. Since [40, 45] generate boxes without considering an instance's physical properties, it remains unannotated despite being static. In contrast, OpenBox recognizes it as static, enabling annotation even with sparse point evidence. In region (B), our approach successfully detects a static car and a pedestrian inside a garage, which the baselines miss or mislocalize. This is because our method refines the point cloud to isolate instance-specific points. Region (C) shows that our method can automatically annotate vehicles on the opposite lane, even when no corresponding human annotations exist. Although both the baseline and our method detect these vehicles, ours localizes them more accurately by extending to both rigid and dynamic instances and by jointly leveraging a vision foundation model, resulting in higher recall. In addition, OpenBox enables automatic annotation of open-vocabulary objects beyond the predefined classes in existing autonomous-driving datasets [2, 10, 12, 33, 39]. As shown in Fig. 8, it successfully annotates objects such as strollers, fire hydrants, and dogs, which are essential to consider in real-world driving scenarios.

5 Conclusion

In this paper, we propose OpenBox, a novel automatic 3D bounding box annotation pipeline. OpenBox effectively leverages 2D vision foundation models to generate open-vocabulary 3D annotations. To ensure high-quality box generation, it refines instance-level point clouds and performs adaptive 3D bounding-box generation tailored to the physical properties of each instance. Our extensive experiments on diverse autonomous driving datasets validate the superiority of the proposed method in annotation quality over prior baselines, while our comprehensive ablation study substantiates the effectiveness of each component in the annotation pipeline. We hope that our method can contribute to future research on foundation models for 3D perception.

Limitations. Adverse weather reduces contrast and obscures edges, which makes 2D vision models unreliable. The 3D annotations built on those models inherit the errors and often miss instances, resulting in imprecise box boundaries. Deformable categories such as pedestrians and cyclists also suffer because pose variation makes full-extent inference unstable. The method then falls back to fixed class-level sizes, which frequently under- or over-size the box. At long range, LiDAR returns become too sparse to constrain geometry, so box fitting is ill-conditioned and localization becomes less precise.

6 Acknowledgments

This work was supported by IITP grant (RS-2021-II211343: AI Graduate School Program (Seoul National University) (5%), No.2021-0-02068: AI Innovation Hub (10%), 25-InnoCORE-01: InnoCORE program of the Ministry of Science and ICT (10%)) and NRF grant (2023R1A1C200781211 (75%)) funded by the Korea government (MSIT).

References

[1] Stefan Baur, Frank Moosmann, and Andreas Geiger. Liso: Lidar-only self-supervised 3d object detection. In *Proc. European Conference on Computer Vision (ECCV)*, 2024.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, 2013.

[4] Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo de Lutio, Janick Martinez Esturo, B. Ivanovic, Or Litany, Zan Gojcic, S Fidler, Marco Pavone, Li-Na Song, and Yue Wang. Omnire: Omni urban scene reconstruction. In *International Conference on Learning Representations (ICLR)*, 2025.

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

[6] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, 2021.

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. Second International Conference on Knowledge Discovery and Data Mining (KDD)*, 1996.

[8] Djamahl Etchegaray, Zi Huang, Tatsuya Harada, and Yadan Luo. Find n' propagate: Open vocabulary 3d object detection in urban scenes. In *Proc. European Conference on Computer Vision (ECCV)*, 2024.

[9] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6), 1981.

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *Proc. European Conference on Computer Vision (ECCV)*, 2022.

[12] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. One thousand and one hours: Self-driving motion prediction dataset. In *Conference on Robot Learning*. PMLR, 2021.

[13] Rui Huang, Henry Zheng, Yan Wang, Zhuofan Xia, Marco Pavone, and Gao Huang. Training an open-vocabulary monocular 3d detection model without 3d data. In *Advances in Neural Information Processing System (NeurIPS)*, 2024.

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[15] Linh Kästner, Vlad Catalin Frasineanu, and Jens Lambrecht. A 3d-deep-learning-based augmented reality calibration method for robotic environments using depth sensor data. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[16] In-Jae Lee, Sihwan Hwang, Youngseok Kim, Wonjune Kim, Sanmin Kim, and Dongsuk Kum. Crab: Camera-radar fusion for reducing depth ambiguity in backward projection based view transformation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[17] Seungjae Lee, Hyungtae Lim, and Hyun Myung. Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3d point cloud. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[18] Ted Lentsch, Holger Caesar, and Dariu M Gavrila. UNION: Unsupervised 3D object detection using object appearance-based pseudo-classes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[19] Feng Li, Hao Zhang, Peize Sun, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, Lei Zhang, and Jianfeng Gao. Segment and recognize anything at any granularity. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Proc. European Conference on Computer Vision (ECCV)*, 2025.

[20] Liunian Harold Li*, Pengchuan Zhang*, Haotian Zhang*, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[21] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *Proc. European Conference on Computer Vision (ECCV)*, 2022.

[22] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *Proc. European Conference on Computer Vision (ECCV)*, 2024.

[23] Zhijian Liu, Haotian Tang, Alexander Amini, Xingyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[24] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection. In *Proc. European Conference on Computer Vision (ECCV)*, 2022.

[25] Mahyar Najibi, Jingwei Ji, Yin Zhou, Charles R. Qi, Xinchen Yan, Scott Ettinger, and Dragomir Anguelov. Motion inspired unsupervised perception and prediction in autonomous driving. In *Proc. European Conference on Computer Vision (ECCV)*, 2022.

[26] Mahyar Najibi, Jingwei Ji, Yin Zhou, Charles R. Qi, Xinchen Yan, Scott Ettinger, and Dragomir Anguelov. Unsupervised 3d perception with 2d vision-language distillation for autonomous driving. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[27] OpenAI. ChatGPT: language model (Mar 2024 version). `https://chat.openai.com`, 2024. Accessed: 2024-05-09.

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

[29] Mihir Prabhudesai, Hsiao-Yu Fish Tung, Syed Ashar Javed, Maximilian Sieb, Adam W Harley, and Katerina Fragkiadaki. Embodied language grounding with 3d visual feature representations. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proc. International Conference on Machine Learning (ICML)*, 2021.

[31] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024.

[32] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[34] Zineng Tang, Long Lian, Seun Eisape, XuDong Wang, Roei Herzig, Adam Yala, Alane Suhr, Trevor Darrell, and David M. Chan. Tulip: Towards unified language-image pretraining, 2025.

[35] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet`, 2020.

[36] Ignacio Vizzo, Tiziano Guadagnino, Jens Behley, and Cyrill Stachniss. Vdbfusion: Flexible and efficient tsdf integration of range sensor data. *Sensors*, 2022.

[37] Daniel Weber, Enkelejda Kasneci, and Andreas Zell. Exploiting augmented reality for extrinsic robot calibration and eye-based human-robot collaboration. In *Proc. ACM/IEEE International Conference on Human-Robot Interaction*, HRI '22, 2022.

[38] Yi Wei, Shang Su, Jiwen Lu, and Jie Zhou. FGR: Frustum-Aware Geometric Reasoning for Weakly Supervised 3D Vehicle Detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[39] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks)*, 2021.

[40] Hai Wu, Shijia Zhao, Xun Huang, Chenglu Wen, Xin Li, and Cheng Wang. Commonsense prototype for outdoor unsupervised 3d object detection. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[41] Yanmin Wu, Xinhua Cheng, Renrui Zhang, Zesen Cheng, and Jian Zhang. Eda: Explicit text-decoupling and dense alignment for 3d visual grounding. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[42] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[43] Yurong You, Katie Luo, Cheng Perng Phoo, Wei-Lun Chao, Wen Sun, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. Learning to detect mobile objects from lidar scans without labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[44] Hu Zhang, Jianhua Xu, Tao Tang, Haiyang Sun, Xin Yu, Zi Huang, and Kaicheng Yu. Opensight: A simple open-vocabulary framework for lidar-based object detection. In *Proc. European Conference on Computer Vision (ECCV)*, 2024.

[45] Lunjun Zhang, Anqi Joyce Yang, Yuwen Xiong, Sergio Casas, Bin Yang, Mengye Ren, and Raquel Urtasun. Towards unsupervised object detection from lidar point clouds. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[46] Ruiyang Zhang, Hu Zhang, Hang Yu, and Zhedong Zheng. Approaching outside: Scaling unsupervised 3d object detection from 2d scene. In *Proc. European Conference on Computer Vision (ECCV)*, 2024.

[47] Xiao Zhang, Wenda Xu, Chiyu Dong, and John M. Dolan. Efficient l-shape fitting for vehicle detection using laser scanners. In *IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[48] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *Proc. European Conference on Computer Vision (ECCV)*, 2022.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We introduce our contributions (Cross-modal Instance Alignment and Adaptive Bounding Box Generation) in the abstract and introduction part.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation of our method in the Section 5

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our method does not include theoritical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We contain configuration of implementation details in the supplementary material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code will be released.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the information of dataset used for the experiment. We also include the additional details in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We could not because of the limited computing resource.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include those information in the appendix due to page limit.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We read and understand the Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Our research is related to social impact in autonomous driving area.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We believe that no risk is involved.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cited the reference paper, code and data properly.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will release the well-documented assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We did not conduct any research related to crowd sourcing and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We did not conduct any research related to crowd sourcing and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We use ChatGPT [27] for part of our method.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

A Overview

In this supplementary material, we provide additional details of OpenBox. Appendix B describes the experiment details. Appendix C provides a detailed approach for height refinement and surface estimation. We further present more experimental results in Appendix D.

B Implementation Details

Table 6: Training and network details for experiment

configs	Voxel R-CNN [6]	Point RCNN [32]	CenterPoint [42]
optimizer	AdamW	AdamW	AdamW
base learning rate	1e-2	1e-2	1e-4
weight decay	1e-3	1e-2	1e-2
momentum	0.9	1e-2	—
momentum range	[0.95, 0.85]	[0.95, 0.85]	—
learning rate decay	0.1	0.1	—
learning rate clip	1e-7	1e-7	—
gradient norm clip	10	10	35
batch size	16	2	32
epoch	20	60	20

We train models [6, 32, 42] on 8 NVIDIA A6000 GPUs (48GB) and 2 AMD EPYC 7763 CPUs. We also employ VDBFusion [36] for SDF. In the Context-aware Refinement part (see Sec. 3.1), the hyperparameters α, β, and δ are set to 0.3, 0.2, and 0.1, respectively. In addition, the threshold τ used in the Handling Static & Rigid Instance part (see Sec. 3.2) is set to 0.15.

C Additional details for method

C.1 ChatGPT prompt

To obtain 3D bounding box size priors and determine the rigidity of objects, we utilized the following prompts with ChatGPT-4 [27]:

Prompt: `Please provide the typical 3D bounding box size for [class].`

Response: `Here are the typical 3D bounding box dimensions (Length × Width × Height in meters) for [class], based on common datasets like nuScenes, KITTI, and the Waymo Open Dataset.`

Prompt: `Is [class] deformable or rigid?`

Response: `A [class] is considered a deformable / rigid object.`

Below is the list of categories we provided to GroundingDINO [22] as text prompts:

```
[Car, Bus, Person, Truck, Construction Vehicle, Trailer,
Barrier, Bicycle, Motorcycle, Traffic Cone, Dog, Fire Hydrant,
Stroller]
```

C.2 Height Refinement

In section 3.1, we exclude points whose z-coordinates are below a predefined threshold to remove remaining ground points after the RANSAC [9] based ground removal. This preliminary step can lead

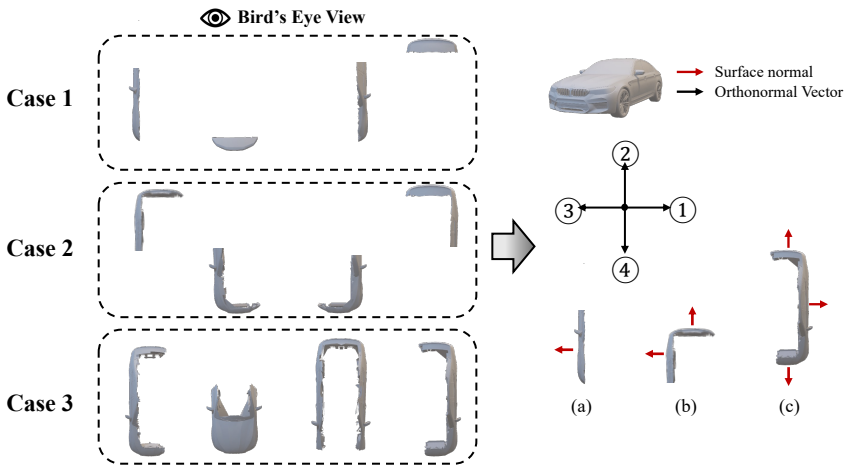


Figure 9: **Illustration of Surface Estimation.**

to bounding boxes appearing elevated above the actual object. To address this issue, we propose an additional refinement step to accurately estimate the z-coordinate of each bounding box. Specifically, given an instance with length l and width w, we calculate a radius as:

$$L_b = \frac{\sqrt{l^2 + w^2}}{2}. \tag{5}$$

We then define the set of points within this radius from the ego-position p_{ego} of the instance:

$$\mathcal{P}' = \{p \in \mathcal{P} \mid |p_{\text{ego}} - p|_2 < L_b\}, \tag{6}$$

where $\mathcal{P}$ denotes the point cloud corresponding to the frame in which the instance is located. After sorting points in $\mathcal{P}'$ by their z-coordinate in ascending order, we select the z-coordinate at the 1%, effectively removing potential noise and LiDAR reflectance outliers near the ground. This procedure ensures a robust estimation of the ground level near the instance.

C.3 Surface Estimation

We determine the surface direction of instance-level surfaces $\mathbf{S}_{\text{ins}}$ to facilitate 3D-2D IoU alignment, as described in the **Handling Static & Rigid Instance** section (Sec. 3.2). Specifically, we compute the dot product between the normal vectors of $\mathbf{S}_{\text{ins}}$ and a set of four orthonormal reference vectors to identify the surface direction.

As illustrated in Fig. 9, in (a), only the ③ direction yields a dot product greater than the predefined threshold $\gamma = 0.8$, allowing us to identify the surface direction. In (b), both ② and ③ exceed γ, while in (c), ①, ②, and ③ all surpass the threshold, indicating the presence of multiple surface orientations.

D More Experimental Results

Quantitative Results Table 7 presents the AP_{3D} results under the `LEVEL_2` of the WOD [33] validation split. Compared to the results under the `LEVEL_1` criterion shown in Table 1, which reflects performance in easier scenarios, the overall performance is lower. Nevertheless, our approach outperforms other baselines [7, 40, 43, 45], indicating that the proposed dataset annotations enable the 3D object detection network [6] to learn effectively even under more challenging conditions.

Qualitative Results We demonstrate OpenBox on the WOD [33] dataset in two scenarios. Scenario 1 compares annotation quality on the original WOD training set for vehicle, pedestrian, and cyclist classes. Scenario 2 presents annotation results for novel object categories. A detailed visualization of both scenarios is provided in the attached video (`supple_video.mp4`).

Table 7: **3D object-detection results on the WOD [33] validation set.** Models marked with * are trained and evaluated in the camera-frustum region, while others use full 360° coverage. † and ‡ denote models trained with CST and CBR from CPD [40], using the training settings given in the next sentence. For †, we flip the **OpenBox** annotations and point clouds to obtain 360° coverage; for ‡, we fill the region outside the camera frustum with CPD annotations. All values denote AP_{3D} at each IoU threshold for `LEVEL_2`. **Bold** means best performance, underlined means second-best.

Method	Modality	Vehicle $IoU_{0.5}$ / $IoU_{0.7}$	Pedestrian $IoU_{0.3}$ / $IoU_{0.5}$	Cyclist $IoU_{0.3}$ / $IoU_{0.5}$
CPD* [40]	LiDAR	26.09 / 17.91	11.87 / 9.30	3.34 / **2.96**
OpenBox* (Ours)	LiDAR + Camera	**62.74 / 28.03**	**51.55 / 15.06**	**20.08** / 1.88
DBSCAN [7]	LiDAR	1.94 / 0.25	0.19 / 0.00	0.20 / 0.00
MODEST [43]	LiDAR	15.83 / 5.48	8.96 / 0.10	0.43 / 0.20
OYSTER [45]	LiDAR	26.21 / 14.10	3.52 / 0.14	1.24 / 0.32
CPD [40]	LiDAR	50.18 / 32.13	18.01 / 13.22	5.61 / 4.87
OpenBox† (Ours)	LiDAR + Camera	**58.42** / 33.72	**47.78 / 31.77**	**20.19 / 6.81**
OpenBox‡ (Ours)	LiDAR + Camera	51.70 / **34.95**	33.02 / 23.50	7.95 / 5.99