VoxDet: Rethinking 3D Semantic Occupancy Prediction as Dense Object Detection

Wuyang Li[1] **Zhu Yu**[2] **Alexandre Alahi**[1]
[1]École Polytechnique Fédérale de Lausanne (EPFL) [2]Zhejiang University
wuyang.li@epfl.ch yu_zhu@zju.edu.cn alexandre.alahi@epfl.ch

Project Page: https://vita-epfl.github.io/VoxDet/

Abstract

3D semantic occupancy prediction aims to reconstruct the 3D geometry and semantics of the surrounding environment. With dense voxel labels, prior works typically formulate it as a *dense segmentation task*, independently classifying each voxel. However, this paradigm neglects critical instance-centric discriminability, leading to instance-level incompleteness and adjacent ambiguities. To address this, we highlight a "free lunch" of occupancy labels: the voxel-level class label implicitly provides insight at the instance level, which is overlooked by the community. Motivated by this observation, we first introduce a training-free **Voxel-to-Instance (VoxNT) trick**: a simple yet effective method that freely converts voxel-level class labels into instance-level offset labels. Building on this, we further propose **VoxDet**, an instance-centric framework that reformulates the voxel-level occupancy prediction as *dense object detection* by decoupling it into two sub-tasks: offset regression and semantic prediction. Specifically, based on the lifted 3D volume, VoxDet first uses (a) Spatially-decoupled Voxel Encoder to generate disentangled feature volumes for the two sub-tasks, which learn task-specific spatial deformation in the densely projected tri-perceptive space. Then, we deploy (b) Task-decoupled Dense Predictor to address this task via dense detection. Here, we first regress a 4D offset field to estimate distances (6 directions) between voxels and the corresponding object boundaries in the voxel space. The regressed offsets are then used to guide the instance-level aggregation in the classification branch, achieving instance-aware prediction. Experiments show that VoxDet can be deployed on both camera and LiDAR input, jointly achieving state-of-the-art results on both benchmarks. VoxDet is not only highly efficient, but also gives 63.0 IoU on the SemanticKITTI test set, **ranking 1**st on the online leaderboard.

1 Introduction

Spatial AI [14] is crucial for autonomous systems to perceive and interpret the complex physical world. As a critical step, precise reconstruction of geometric structures and semantics lays the foundation for scene understanding, underpinning the downstream forecasting and planning [1]. This capability is indispensable for applications such as autonomous driving and robotic navigation.

To this end, 3D semantic occupancy prediction, derived from semantic scene completion [52], has attracted significant attention by simultaneously inferring complete 3D geometry and semantics through voxel representation. Prior works can be broadly categorized into LiDAR-based [55, 51, 66] and camera-based approaches [86, 35, 82, 23]. The former uses sparse 3D inputs (e.g., point clouds) to provide precise geometric information. In contrast, camera-based methods have recently demonstrated promising potential due to their flexibility and computational efficiency. They employ

Preprint. Under review.

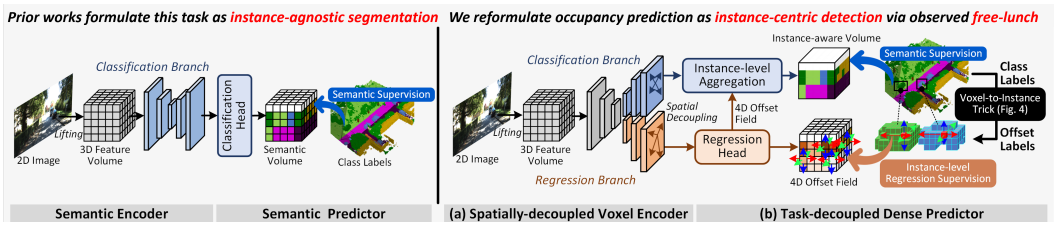


Figure 1: Schematic comparison of previous paradigm [6, 2, 82] and our VoxDet. **Left:** Prior works use a 3D semantic segmentation formulation, directly predicting voxel labels agnostic to object instances. **Right:** VoxDet reformulates this task as dense object detection to explicitly learn with instances, which decouples it into instance-aware regression and classification. This is achieved via a Voxel-to-Instance (VoxNT) trick (Fig. 4), inspired by our observed free-lunch in labels (Sec. 3).

dedicated vision-centric algorithms to lift 2D imagery into 3D space, including Features Line-of-Sight Projection (FLoSP) [6], depth-driven back projection [35, 23, 80], and Lift-Splat-Shoot (LSS) [47] based voxels [82, 25, 2]. These methods offer unique advantages in resource-constrained settings such as robotics, owing to their low cost, high flexibility, and real-time processing capabilities.

While achieving great progress, existing methods uniformly formulate occupancy prediction as a semantic segmentation task[1] on dense voxels, which fail to understand the instance-level semantics and geometry. This paradigm typically employs 3D encoders [86, 82] to encode semantic patterns from the lifted volumes (see Fig. 1 **Left**), subsequently classifying each voxel independently. Due to the absent instance labels, this voxel-centric paradigm may seem viable, but it has to face severe instance-level incompleteness and adjacent ambiguities (see Fig. 7). Although prior works, such as Symphonize [23], have noticed this issue and attempted to mitigate it with object queries from 2D images, they still optimize the 3D space via dense segmentation on each voxel. The substantial gap between the image and voxel domains prevents effective instance-driven learning, making the segmentation-based formulation fail to infer the complex environment with a lot of dynamic agents.

To address this issue, we first highlight a "free-lunch" of the occupancy labels: ***the voxel-level class label has implicitly told the instance-level insight, which is ever-overlooked by the community.*** First, unlike in 2D images where occlusion causes overlapping objects to conflate into a single entity, instances in 3D voxels are inherently separable. Every voxel is deterministically assigned to a single class without occlusion-induced ambiguity, making the instance-level discovery essentially realistic. Second, instance boundaries in 3D voxel space are naturally tractable and regressible, ensuring clear distinctions between instances. More details can be found in Sec. 3.

Inspired by these observations, we first develop a **Voxel-to-Instance (VoxNT) Trick** (Fig. 4) that can freely convert the voxel-level class labels to the instance-level offset labels, fully harnessing the free-lunch (Sec. 3). Here, the offset is defined as the Euclidean distance between each voxel and its associated instance borders. Then, these *free offset labels* prompt us to rethink the segmentation-based formulation. In response to this, we propose **VoxDet** to reformulate Voxel-level occupancy prediction as instance-centric object Detection (see Fig. 1 **Right**). Unlike prior works, VoxDet decouples this task into two sub-tasks: offset regression and semantic prediction. We achieve this with (a) Spatially-decoupled Voxel Encoder (SVE) and (b) Task-decoupled Dense Predictor (TDP) at the representation and prediction levels, respectively: Given 3D volumes lifted from 2D images [47], SVE first learns task-specific features by spatially deforming in the tri-perceptive space, which are sent to the two branches of TDP, respectively. Within TDP, we regress each voxel to the associated instance boundary by predicting a novel 4D offset field, which guides an instance-level aggregation for instance-centric occupancy prediction.

Hence, with our new detection-based formulation, VoxDet achieves true instance-centric perception solely using voxel-level labels, which can be effortlessly extended to the LiDAR settings. VoxDet is not only highly efficient by reducing 57.9% parameters with 1.3 $\times$ speed-up, but also achieves state-of-the-art results on both camera and LiDAR benchmarks. Notably, our method gives 63.0 IoU, **ranking 1st** on the CodaLab leaderboard, without using extra labels/data/temporal information/models.

In summary, our contributions lie in the following aspects:

[1]For simplicity, we uniformly refer to voxel recognition and completion [6] as dense classification.

- By analyzing the difference of 2D pixels and 3D voxels (Sec. 3), we reveal the overlooked free lunch of occupancy labels for instance-level learning. With this observation, we propose a VoxNT trick to freely convert voxel-level labels to instance-level offsets. As a byproduct, this trick can also identify wrong labels of dynamic objects (see Appendix and Fig. 7).
- We reformulate semantic occupancy prediction as a dense object detection task by advancing VoxDet, decoupling into two sub-tasks: offset regression and semantic prediction, for instance-level perception.
- We design a Spatially-decoupled Voxel Encoder (SVE) to decouple 3D volumes for our new formulation, which learns task-specific spatial deformation in the densely projected tri-perceptive space for the two sub-tasks, avoiding the misalignment between them.
- We propose a Task-decoupled Dense Predictor (TDP) to enable instance-driven prediction. This comprises a regression branch that predicts a 4D offset field, delineating instance boundaries, and a classification branch using learned offsets for instance-aware aggregation.

2 Related Work

3D Semantic Occupancy Prediction, derived from Semantic Scene Completion [55, 7, 9], aims to jointly reconstruct the semantics and geometry of a surrounding environment with voxelization. Existing studies can be generally divided into LiDAR-based and camera-based methods. The former utilizes point clouds to achieve high accuracy with precise depth, which is limited by the computational cost. Camera-based methods rely solely on 2D visual inputs to generate 3D scene understanding. With the advancement of monocular vision like LSS [47], these approaches offer great advantages in efficiency. MonoScene [6] pioneered the camera-based setting by connecting 2D images with the 3D voxel space via the FLoSP. VoxFormer [35] uses 3D-to-2D back-projection and disseminates the semantics of the visible queries across the entire 3D volume via MAE [20]. Subsequent works have further enhanced voxel representations using techniques such as tri-perceptive enhancement [82, 2], diffusion models [36], vision-language models [58, 79], geometric depth [81, 73, 72, 71], and extra modalities [19, 60, 70], boosting downstream applications [42, 27, 41, 76]. However, existing methods uniformly treat this task as dense segmentation, lacking explicit instance-level perception. In contrast, we reformulate it as a dense object detection task with explicit instance-level awareness.

Dense Object Detection, such as point-based FCOS [56, 62], is a fully convolutional paradigm known for its lightweight design, efficiency, and performance, which garnered significant attention prior to the DETR series [8, 63, 89, 44]. The core insight is the notion of "***densely detecting like segmentation***": every pixel within a bounding box regresses its distances in four directions (up, down, left, right) while simultaneously predicting the instance-level class label, which is a dense process like the per-pixel segmentation. The following works focus on improving this paradigm from different aspects, including considering better label assignment [84, 83] like ATSS [84], architecture search [59], spatially task decoupling [11], border enhancement [48], dense feature distillation [87, 74, 32, 31], optimization signals [68, 40]. Besides, dense detection has been extended to 3D vision. For instance, FCOS-3D [62] predicts 3D targets in the 2D space by regressing 3D attributes for each pixel, and UVTR [33] predicts a 3D position for each pixel to enhance instance-level localization. Unlike previous works, we reverse the philosophy via a "***segmenting like dense detection***" framework that endows voxel-based segmentation with instance-level awareness, thereby eliminating semantic and geometric ambiguity in occupancy prediction. Notably, our approach not only lifts the pixel-based regression into the 3D voxel space but also eliminates the need for instance-level bounding box labels.

3 Preliminaries and Motivation

We start by analyzing the differences between 2D pixel and 3D voxel space with respect to semantic-level classification and instance-level regression, from the perspective of dense perception [56]. We then clarify our observations and insights on the ever-overlooked "free-lunch" in voxel-level class labels and explain the associated motivation for the following methodology.

Semantic-based Dense Classification. In Fig. 2 (**Top**), it can be observed that occlusion in the 2D space often leads to the merging of distinct object instances (e.g., the blue-highlighted cars) into a single entity (**Left**). Due to the perceptive projection [15], multiple 3D points at varying depths in the world coordinate system converge into a single pixel of the image. In contrast, the 3D voxel space

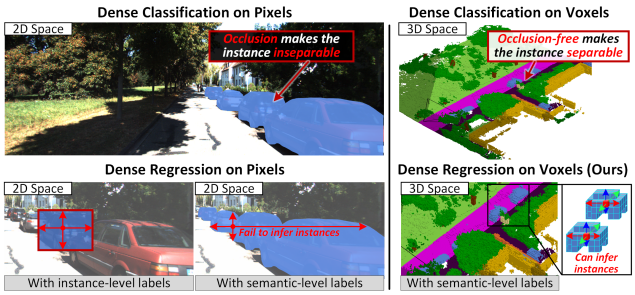


Figure 2: Conceptual comparison between 2D pixels and 3D voxels regarding semantic-level dense classification and instance-level regression. **Left:** 2D occlusion makes instance perception fail with semantic labels. **Right:** The occlusion-free nature of 3D space enables instances separable only using semantic labels.

inherently avoids most of such occlusion, as each voxel is uniquely assigned a class label, including the *empty* class (**Right**). *Consequently, the occlusion-free property of 3D voxels facilitates the natural separation of object instances*, although minor ambiguities may arise in cases such as the densely overlapping foliage of adjacent trees, which typically do not affect the scene understanding.

Instance-based Dense Regression. In Fig. 2 (**Bottom**), each pixel and voxel aims to regress the distance to the instance border [56]. In the 2D domain (**Left**), such regression supports instance-level perception via bounding box labels; however, it fails to infer instance cues with semantic labels. *Conversely, in the 3D voxel space, the natural separability of instances makes it possible to regress instances without instance labels* (**Right**). For example, consider the red voxel inside the highlighted car in the figure. As the voxel space is occlusion-free, we can regress from this voxel to its instance boundary by identifying adjacent voxels with differing semantic labels (see Fig. 4). This property enables us to infer instances using only semantic annotations, a valuable yet overlooked *free lunch*.

Motivation and Insight. Inspired by these observations, we aim to rethink Voxel-level semantic occupancy prediction as object Detection [56], termed VoxDet, which adopts a novel "segmenting like dense detection" philosophy. Instead of directly recognizing each independent voxel [86, 82, 2, 35], VoxDet decouples it into regression and classification sub-tasks in the 3D voxel space, where, in particular, the core innovative regression (or offset) branch enables explicit instance-level 3D regression. The learned offsets are subsequently used for instance-level aggregation, facilitating scene understanding. Note that this fundamentally differs from and advances beyond the traditional segmentation paradigm by ***explicitly lifting voxels to instances***, with potential applicability to the broader occupancy community, such as multi-camera settings.

4 Methodology

Overview. Fig. 3 shows the overall workflow of our VoxDet. Given RGB input, we follow previous works [82] to conduct **(a)** 2D-to-3D lifting to generate 3D feature volumes $\mathbf{V}$, which uses the estimated depth $\mathbf{Z}$ given by the arbitrary depth estimator [5, 53]. Then, we send $\mathbf{V}$ to **(b)** Spatially-decoupled Voxel Encoder to generate the disentangled feature for dense classification $\mathbf{V}_{\text{cls}}$ and regression $\mathbf{V}_{\text{reg}}$. Here, we encourage the two tasks to learn spatially decoupled features to avoid task-misalignment [68], which is achieved in a densely projected tri-perceptive (TPV) space. Next, the decoupled volumes are sent to **(c)** Task-decoupled Dense Predictor. In this part, we first regress a 4D offset field Δ to estimate the distance between each voxel $\mathbf{V}_{i,j,k}$ to the instance boundary $\Delta_{i,j,k} \in \mathbb{R}^6$ in six directions (see Fig. 4). The learned offset Δ is subsequently sent to the classification branch to guide instance-level aggregation, thereby achieving instance-aware semantic occupancy prediction.

4.1 2D-to-3D Lifting

We follow previous works [82, 2, 35] to conduct the same 2D-to-3D lifting (Fig. 3a), outputting the 3D feature volume $\mathbf{V} \in \mathbb{R}^{X \times Y \times Z \times C}$, where X, Y, Z, and C is the depth, width, height, and channel respectively. The process is briefly described as follows. More details are in the Appendix.

Given the RGB image $\mathbf{I}$, we extract the 2D image feature $\mathbf{F}^{\text{2D}}$ using the image encoder, and estimate the depth map $\mathbf{Z}$ with the frozen depth estimator [5, 53] following the unified practice [82, 35, 2]. Then, we estimate the depth probability $\mathbf{D}$ using LSS [47]. Based on these, we can establish the 3D feature $\mathbf{F}^{\text{3D}}$ using the fused depth probability $\mathbf{D}$ and pre-extracted 2D feature $\mathbf{F}^{\text{2D}}$ [82], which is subsequently projected onto the voxel grid to generate the 3D volume $\mathbf{V}$. In this procedure, each voxel in $\mathbf{V}$ is able to query the information from 3D features $\mathbf{F}^{\text{3D}}$ via deformable cross-attention [13].

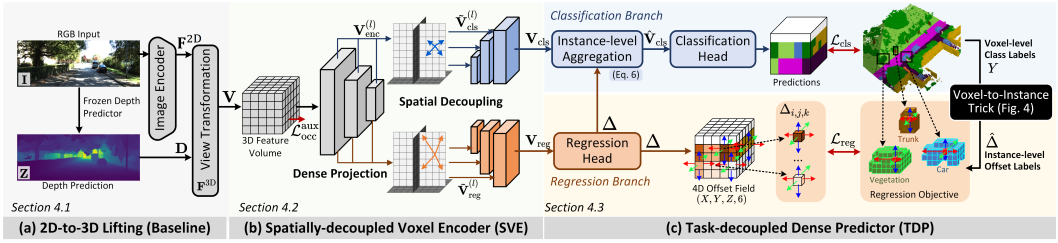


Figure 3: Overview of our VoxDet. After 2D-to-3D lifting, VoxDet spatially decouples 3D volumes $\mathbf{V}$ into two task-specific branches, learning different spatial deformations in the densely projected tri-perceptive space. Then, VoxDet regresses a 4D offset field Δ towards instance boundaries with $\mathbf{V}_{\text{reg}}$, serving for the instance-level aggregation with $\mathbf{V}_{\text{cls}}$ in the classification branch.

4.2 Spatially-decoupled Voxel Encoder

Given the 3D volume $\mathbf{V}$, we aim to extract task-specific representations for the two tasks. Prior encoders [28, 12] are designed for voxel segmentation, which lacks the spatial context for regression with task-misalignment [54, 30, 11]. Hence, we spatially decouple $\mathbf{V}$ into $\mathbf{V}_{\text{cls}}$ for classification and $\mathbf{V}_{\text{reg}}$ for regression (Fig. 3b) in a densely projected TPV space. Fig. 6 shows the decoupling effect.

Dense Projection. Given the 3D volume $\mathbf{V}$, we use a shared encoder to extract L-level task-shared feature volumes $\{\mathbf{V}_{\text{enc}}^{(l)}\}_{l=1}^{L}$. Due to the task-misaligned feature preference [68], directly using the shared volume for the two tasks will severely influence the instance-level perception. To resolve this, we spatially decouple task-specific features in a densely projected tri-perceptive (TPV) space [22]. This can avoid the large computational cost of decoupling in 3D, effectively reducing the dimension to 2D. To this end, a dense projection $\mathcal{P}_d(\cdot)$ is proposed to generate 2D feature planes for our detection-based formulation. Specifically, given $\mathbf{V}_{\text{enc}}^{(l)}$, we learn per-voxel dense weight with a linear layer $\boldsymbol{\eta} = \mathbf{W}_d(\mathbf{V}_{\text{enc}}^{(l)}), \boldsymbol{\eta} \in \mathbb{R}^{X\times Y\times Z\times 3}$, which guides the adaptive pooling for the three TPV planes. Then, we conduct pooling across each axis (X, Y, Z) to generate projected 2D feature planes:

$$\mathcal{P}_d^Z, \mathcal{P}_d^Y, \mathcal{P}_d^X(\mathbf{V}) = \sum_{z=1}^{Z} \mathcal{S}_Z(\boldsymbol{\eta}_{[:,:,:,1]}) \circ \mathbf{V}_{[:,:,z]}, \sum_{y=1}^{Y} \mathcal{S}_Y(\boldsymbol{\eta}_{[:,:,:,2]}) \circ \mathbf{V}_{[:,y,:]}, \sum_{x=1}^{X} \mathcal{S}_X(\boldsymbol{\eta}_{[:,:,:,3]}) \circ \mathbf{V}_{[x,:,:]}\,. \tag{1}$$

Here, $\mathcal{P}_d^{(A)}$ indicates the (A)-axis projection, $\mathcal{S}_{(A)}$ is $\mathrm{softmax}$ on the (A) axis, and $\circ$ is Hadamard product. Next, we deploy a 2D refinement layer to generate TPV feature at XY, XZ, YZ planes:

$$\begin{aligned}
\mathbf{V}_{\text{cls}}^{XY} &= \text{Conv}_{\text{cls}}^{XY}(\mathcal{P}_d^Z(\mathbf{V}_{\text{enc}}^{(l)})), \mathbf{V}_{\text{cls}}^{XZ} = \text{Conv}_{\text{cls}}^{XZ}(\mathcal{P}_d^Y(\mathbf{V}_{\text{enc}}^{(l)})), \mathbf{V}_{\text{cls}}^{YZ} = \text{Conv}_{\text{cls}}^{YZ}(\mathcal{P}_d^X(\mathbf{V}_{\text{enc}}^{(l)})), \\
\mathbf{V}_{\text{reg}}^{XY} &= \text{Conv}_{\text{reg}}^{XY}(\mathcal{P}_d^Z(\mathbf{V}_{\text{enc}}^{(l)})), \mathbf{V}_{\text{reg}}^{XZ} = \text{Conv}_{\text{reg}}^{XZ}(\mathcal{P}_d^Y(\mathbf{V}_{\text{enc}}^{(l)})), \mathbf{V}_{\text{reg}}^{YZ} = \text{Conv}_{\text{reg}}^{YZ}(\mathcal{P}_{\text{ada}}^X(\mathbf{V}_{\text{enc}}^{(l)})).
\end{aligned} \tag{2}$$

Here, $\text{Conv}_{(T)}^{(P)}(\cdot)$ is 2D convolution module for plane (P) and task (T). Compared with the conventional TPV pooling [22], our dense projection can adaptively discover the essential voxels during the 3D-to-2D dimensional reduction, thereby mitigating the information loss for the dense perception.

Spatial Decoupling. With projected 2D features, we spatially decouple it in each TPV plane with different spatial offsets, encouraging two tasks to focus on task-specific regions. Then, we use a Conv layer to fuse the decoupled features with the same expanded size, which is carried out as follows:

$$\begin{aligned}
\tilde{\mathbf{V}}_{\text{cls}}^{(l)} &= \text{Conv}_{\text{cls}}^{\text{fuse}}\left(\text{DefConv}_{\text{cls}}^{XY}(\mathbf{V}_{\text{cls}}^{XY}) + \text{DefConv}_{\text{cls}}^{XZ}(\mathbf{V}_{\text{cls}}^{XZ}) + \text{DefConv}_{\text{cls}}^{YZ}(\mathbf{V}_{\text{cls}}^{YZ})\right); \\
\tilde{\mathbf{V}}_{\text{reg}}^{(l)} &= \text{Conv}_{\text{reg}}^{\text{fuse}}\left(\text{DefConv}_{\text{reg}}^{XY}(\mathbf{V}_{\text{reg}}^{XY}) + \text{DefConv}_{\text{reg}}^{XZ}(\mathbf{V}_{\text{reg}}^{XZ}) + \text{DefConv}_{\text{reg}}^{YZ}(\mathbf{V}_{\text{reg}}^{YZ})\right).
\end{aligned} \tag{3}$$

Here, $\text{DefConv}(\cdot)$ is the 2D deformable convolution module, which spatially decouples the dense voxel features for the two tasks. This decoupling reduces the computational burden and preserves task-driven spatial context, effectively addressing the misaligned feature preference of the two tasks. Finally, we send the decoupled multi-level features to the lightweight FPN [39] branches, respectively, to improve the task-specific learning, and collect the last-layer output with the same resolution as $\mathbf{V}$ for the two tasks, which is denoted as: $\mathbf{V}_{\text{cls}} = \text{FPN}(\{\tilde{\mathbf{V}}_{\text{cls}}^{(l)}\}_{l=1}^{L})$ and $\mathbf{V}_{\text{reg}} = \text{FPN}(\{\tilde{\mathbf{V}}_{\text{reg}}^{(l)}\}_{l=1}^{L})$.

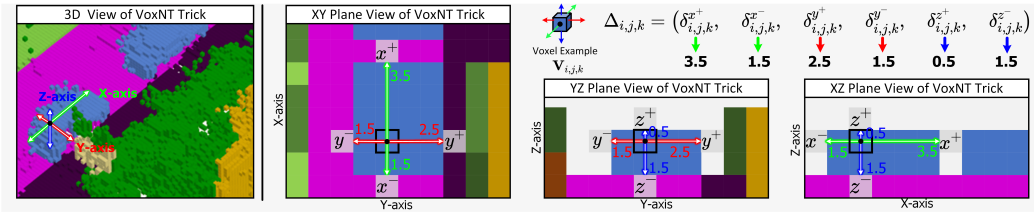


Figure 4: Illustration of our regression objective. For each voxel $\mathbf{V}_{i,j,k}$, we scan in 6 directions and calculate the distance to the instance boundary to generate labels $\hat{\Delta}_{i,j,k}$, using the free-lunch (Sec. 3).

4.3 Task-decoupled Dense Predictor

Then, the decoupled features $\mathbf{V}_{\text{cls}}$ and $\mathbf{V}_{\text{reg}}$ are sent to the classification and regression branches (see Fig. 3c). Here, we first densely regress the instance boundary to identify 3D objects, then aggregate instance-level semantics in the classification branch, achieving an instance-centric prediction.

Regression Branch. Given the volume for regression $\mathbf{V}_{\text{reg}} \in \mathbb{R}^{X\times Y\times Z\times C}$, we aim to regress the distance to the instance boundary for each voxel $\mathbf{V}_{i,j,k} \in \mathbf{V}_{\text{reg}}$ (see Fig. 4) in six directions, which form a 4D offset field $\Delta \in \mathbb{R}^{X\times Y\times Z\times 6}$. We use 6 directions as it is the minimum number to determine a 3D bounding box, similar to the 2D bounding boxes determined with 4 directions [56]. Thus, considering the voxel in $\mathbf{V}_{\text{reg}}$ at position (i, j, k), the spatially associated element $\Delta_{i,j,k}$ in the offset field represents a 6-channel vector for the specific offset δ along the X, Y, and Z axes, which is divided into positive ($^+$) and negative ($^-$) directions. This can be denoted as follows,

$$\Delta = \text{RegressionHead}(\mathbf{V}_{\text{reg}}), \text{ where } \Delta_{i,j,k} = \left(\delta^{x^+}_{i,j,k}, \delta^{x^-}_{i,j,k}, \delta^{y^+}_{i,j,k}, \delta^{y^-}_{i,j,k}, \delta^{z^+}_{i,j,k}, \delta^{z^-}_{i,j,k}\right) \in \mathbb{R}^6. \quad (4)$$

Here, RegressionHead$(\cdot)$ is a lightweight head to predict the offset field, which is simply deployed as a two-convolution module to enable non-linearity with 6-channel output, followed by `Sigmoid` for normalization. Compared with the anchor-based design [49], this design is highly computationally efficient due to its fully convolutional nature when deployed in the dense 3D voxel space.

Regression Loss. In this task, we only have dense class labels for the voxel grid $Y \in \mathbb{N}^{X\times Y\times Z}$ without instance-level labels for regression, such as bounding boxes. Hence, to break through this, we propose a **Voxel-to-Instance (VoxNT) Trick** to freely transform the class labels to the instance-level offsets, fully using the free lunch in Sec. 3. The algorithm details are in the Appendix.

Fig. 4 shows the VoxNT trick. In brief, for each voxel $\Delta_{i,j,k}$, e.g., the blue example, we scan across 6 directions $\mathbf{d} \in \{x^+, x^-, y^+, y^-, z^+, z^-\}$ in labels Y, and stop when the class of next scanned voxel changes, indicating approaching the border. Then, we save the scanning distance as the offset labels: $\hat{\Delta} \in \mathbb{R}^{X\times Y\times Z\times 6}$. For stability, we round up $\hat{\Delta}$ to the integer and normalize into $[0, 1]$ via the volume size. Finally, we deploy L1 loss to optimize the regression with enough gradient:

$$\mathcal{L}_{\text{reg}} = \Sigma^{X,Y,Z,6}_{i,j,k,d} |\Delta_{i,j,k,d} - \hat{\Delta}_{i,j,k,d}|, \quad (5)$$

where $\Delta \in \mathbb{R}^{X\times Y\times Z\times 6}$ is the predicted 4D offset field. Surprisingly, as a by-product, the offset field can identify wrongly labeled dynamic objects (like cars) for more accurate training (see Appendix).

Classification Branch. Based on the regressed offset Δ, we aim to enhance instance-level perception by aggregating the semantics within instances. A natural idea [18] is to crop 3D bounding boxes, which is extremely computationally expensive. Hence, we propose an alternative solution by adaptively aggregating the voxels at regressed positions, considering the informative nature of borders [24, 88]. For each voxel $\mathbf{V}_{i,j,k} \in \mathbf{V}_{\text{cls}}$, we have the predicted offsets $\Delta_{i,j,k}$ associating with 6 voxels, which are extracted as a instance-level voxel set $\mathbf{V}_\delta$, where $\delta \in \Delta_{i,j,k}$ and $|\Delta_{i,j,k}|$=6. Then, each voxel $\mathbf{V}_{i,j,k}$ will query the semantics from the associated voxel set $\mathbf{V}_\delta$ attentively:

$$\hat{\mathbf{V}}_{i,j,k} = \text{Norm}\Big(\sum_{\delta\in\Delta_{i,j,k}} \frac{\exp\left(\mathbf{W}_q\,\mathbf{V}_{i,j,k}\right)^\top \left(\mathbf{W}_k\,\mathbf{V}_\delta\,/\sqrt{d_k}\right)}{\sum_{\delta'\in\Delta_{i,j,k}} \exp\left(\mathbf{W}_q\,\mathbf{V}_{i,j,k}\right)^\top \left(\mathbf{W}_k\,\mathbf{V}_{\delta'}\,/\sqrt{d_k}\right)} \mathbf{W}_v\,\mathbf{V}_\delta + \mathbf{V}_{i,j,k}\Big). \quad (6)$$

Here, $\hat{\mathbf{V}}_{i,j,k} \in \hat{\mathbf{V}}_{\text{cls}}$ is refined feature, each $\mathbf{W}_{(\cdot)}$ is a linear layer, d_k is the channel number, and Norm is `Group Normalization`. With $N = 4$ aggregation layers, each voxel is able to gain sufficient perception of the whole instance, enhancing instance-level semantics. This design is justified in Fig. 5 and the Appendix. Finally, $\hat{\mathbf{V}}_{\text{cls}}$ is sent to a simple classification head to generate class predictions.

Table 1: Camera-based results on SemanticKITTI [3] hidden test set. **T** is using extra temporal information. The best and the second best results are in **bold** and underlined, respectively. R-50 indicates ResNet-50, and Eff-B7 is the stronger EfficientNet-B7 backbone with more parameters. † indicates setting the same weight (3.0) on the cross-entropy loss as the recent work [60].

Method	Arch.	T	IoU	mIoU	road	sidewalk	parking	other-grnd.	building	car	truck	bicycle	motorcycle	other-veh.	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traf.-sign
MonoScene* [6]	Eff-B7		34.16	11.08	54.70	27.10	24.80	5.70	14.40	18.80	3.30	0.50	0.70	4.40	14.90	2.40	19.50	1.00	1.40	0.40	11.10	3.30	2.10
TPVFormer [22]	Eff-B7		34.25	11.26	55.10	27.20	27.40	6.50	14.80	19.20	3.70	1.00	0.50	2.30	13.90	2.60	20.40	1.10	2.40	0.30	11.00	2.90	1.50
SurroundOcc [65]	Eff-B7		34.72	11.86	56.90	28.30	30.20	6.80	15.20	20.60	1.40	1.60	1.20	4.40	14.90	3.40	19.30	1.40	2.00	0.10	11.30	3.90	2.40
OccFormer [85]	Eff-B7		34.53	12.32	55.90	30.30	31.50	6.50	15.70	21.60	1.20	1.50	1.70	3.20	16.80	3.90	21.30	2.20	1.10	0.20	11.90	3.80	3.70
IAMSSC [67]	R-50		43.74	12.37	54.00	25.50	24.70	6.90	19.20	21.30	3.80	1.10	0.60	3.90	22.70	5.80	19.40	1.50	2.90	0.50	11.90	5.30	4.10
VoxFormer [35]	R-50		42.95	12.20	53.90	25.30	21.10	5.60	19.80	20.80	3.50	1.00	0.70	3.70	22.40	7.50	21.30	1.40	2.60	0.20	11.10	5.10	4.90
VoxFormer [35]	R-50	✓	43.21	13.41	54.10	26.90	25.10	7.30	23.50	21.70	3.60	1.90	1.60	4.10	24.40	8.10	24.20	1.60	1.10	0.00	13.10	6.60	5.70
DepthSSC [78]	R-50		44.58	13.11	55.64	27.25	25.72	5.78	20.46	21.94	3.74	1.35	0.98	4.17	23.37	7.64	21.56	1.34	2.79	0.28	12.94	5.87	6.23
Symphonize [23]	R-50		42.19	15.04	58.40	29.30	26.90	11.70	24.70	23.60	3.20	3.60	2.60	5.60	24.20	10.00	23.10	**3.20**	1.90	**2.00**	16.10	7.70	8.00
HASSC [61]	R-50		43.40	13.34	54.60	27.70	23.80	6.20	21.10	22.80	4.70	1.60	1.00	3.90	23.80	8.50	23.30	1.60	4.00	0.30	13.10	5.80	5.50
HASSC [61]	R-50	✓	42.87	14.38	55.30	29.60	25.90	11.30	23.10	23.00	2.90	1.90	1.50	4.90	24.80	9.80	26.50	1.40	3.00	0.00	14.30	7.00	7.10
StereoScene [26]	Eff-B7		43.34	15.36	61.90	31.20	30.70	10.70	24.20	22.80	2.80	3.40	2.40	**6.10**	23.80	8.40	27.00	2.90	2.20	0.50	16.50	7.00	7.20
H2GFormer [64]	R-50		44.20	13.72	56.40	28.60	26.50	4.90	22.80	23.40	4.80	0.80	0.90	4.10	24.60	9.10	23.80	1.20	2.50	0.10	13.30	6.40	6.30
H2GFormer [64]	R-50	✓	43.52	14.60	57.90	30.40	30.00	6.90	24.00	23.70	5.20	0.60	1.20	5.00	25.20	10.70	25.80	1.10	0.10	0.00	14.60	7.50	9.30
MonoOcc [86]	R-50		-	13.80	55.20	27.80	25.10	9.70	21.40	23.20	5.20	2.20	1.50	5.40	24.00	8.70	23.00	1.70	2.00	0.20	13.40	5.80	6.40
CGFormer [82]	Eff-B7		44.41	16.63	64.30	34.20	34.10	12.10	25.80	26.10	4.30	3.70	1.30	2.70	24.50	11.20	29.30	1.70	3.60	0.40	18.70	8.70	9.30
L2COcc-C [60]	Eff-B7		44.31	17.03	**66.00**	35.00	33.10	13.50	25.10	27.20	3.00	3.50	3.60	4.30	25.20	11.50	30.10	1.50	2.40	0.20	20.50	9.10	8.90
HTCL [25]	Eff-B7	✓	44.23	17.09	64.40	34.80	33.80	12.40	25.90	**27.30**	5.70	1.80	2.20	5.40	25.30	10.80	31.20	1.10	3.10	0.90	21.10	9.00	8.30
VoxDet (Ours)	R-50		47.27	18.47	64.70	35.50	34.80	**14.40**	28.10	26.90	**6.10**	**5.90**	5.10	5.00	28.70	**13.60**	31.70	3.10	4.00	1.30	21.50	10.10	10.30
VoxDet† (Ours)	R-50		**47.81**	**18.67**	65.50	**36.10**	**35.50**	13.20	**28.40**	**27.30**	5.40	4.60	**5.40**	5.40	**29.50**	13.10	**32.00**	3.10	**6.10**	0.90	**22.10**	**10.20**	**11.10**

Classification Loss. In dense detection [56], classification is optimized as segmentation in a per-pixel dense manner. Hence, to optimize the classification branch in our VoxDet, we naturally deploy the dense semantic prediction loss following the previous arts [86, 82, 85]:

$$\mathcal{L}_{\text{cls}} = \mathcal{L}_{\text{ce}} + \mathcal{L}_{\text{aff}}^{\text{bin}} + \mathcal{L}_{\text{aff}}^{\text{sem}}, \tag{7}$$

where $\mathcal{L}_{\text{ce}}$ is the cross-entropy loss weighted by class frequencies, and $\mathcal{L}_{\text{aff}}^{\text{bin}}$ and $\mathcal{L}_{\text{aff}}^{\text{sem}}$ are affinity loss with binary and semantic settings. We set the consistent loss weight as 1.0 for convenience.

4.4 Optimization

To train the proposed VoxDet, we implement the whole optimization objective written as follows,

$$\mathcal{L}_{\text{VoxDet}} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} + \lambda \mathcal{L}_{\text{occ}}^{\text{aux}}, \tag{8}$$

where $\mathcal{L}_{\text{cls}}$ is dense classification loss (Eq. 7) and $\mathcal{L}_{\text{reg}}$ is dense regression loss (Eq. 5). Following the baseline [86], we retain the voxel-centric segmentation loss (cross-entropy and affinity terms) applied to $\mathbf{V}$ as an auxiliary prior constraint ($\mathcal{L}_{\text{occ}}^{\text{aux}}$) before instance-centric learning, which can stabilize the optimization. This also enhances the multiple-run robustness, which is justified in the Appendix. The weighting factor $\lambda = 0.2$ is empirically adjusted to balance the model learning at the instance level.

5 Experiments

5.1 Comparison with State-of-the-art Methods

5.1.1 Camera-based Benchmarks

Tab. 1 shows the comparison on the hidden test set of SemanticKITTI [3]. VoxDet achieves new records with an IoU of 47.27 and mIoU of 18.47. Compared with the methods using additional temporal labels like VoxFormer-T [35], HASSC-T [61], H2GFormer-T [64], and HTCL-T [25], VoxDet comprehensively surpasses them and gives noticeable 3.04 and 1.38 gains on IoU and mIoU over the previous best entry [25]. This clearly verifies the effectiveness of our method.

We further list the comparison on SSCBench-KITTI-360 [37] in Tab. 2. VoxDet achieves an IoU of 48.59 and mIoU of 21.40, outperforming other camera-based methods. *Notably, VoxDet achieves the best on all instance-related classes, showing an impressive capacity in understanding outdoor agents.*

5.1.2 Extension to LiDAR-based Benchmark

In Tab. 3, we deploy our method with LiDAR settings, denoted as **VoxDet-L**, and give a comparison on the hidden test set (online evaluation). Our VoxDet-L achieves the new record with a 63.0 IoU

Table 2: Camera-based results on SSCBench-KITTI360 test set. * is corrected using the consistent dataset version [19]. The best and second best results are are in **bold** and underlined, respectively.

Method	IoU	mIoU	car	bicycle	motorcycle	truck	other-veh.	person	road	parking	sidewalk	other-grnd.	building	fence	vegetation	terrain	pole	traf.-sign	other-struct.	other-obj.
LiDAR-based methods																				
SSCNet [55]	53.58	16.95	31.95	0.00	0.17	10.29	0.00	0.07	65.70	17.33	41.24	3.22	44.41	6.77	43.72	28.87	0.78	0.75	8.69	0.67
LMSCNet [51]	47.35	13.65	20.91	0.00	0.00	0.26	0.58	0.00	62.95	13.51	33.51	0.20	43.67	0.33	40.01	26.80	0.00	0.00	3.63	0.00
Camera-based methods																				
MonoScene [6]	37.87	12.31	19.34	0.43	0.58	8.02	2.03	0.86	48.35	11.38	28.13	3.32	32.89	3.53	26.15	16.75	6.92	5.67	4.20	3.09
TPVFormer [22]	40.22	13.64	21.56	1.09	1.37	8.06	2.57	2.38	52.99	11.99	31.07	3.78	34.83	4.80	30.08	17.52	7.46	5.86	5.48	2.70
OccFormer [85]	40.27	13.81	22.58	0.66	0.26	9.89	3.82	2.77	54.30	13.44	31.53	3.55	36.42	4.80	31.00	19.51	7.77	8.51	6.95	4.60
VoxFormer [35]	38.76	11.91	17.84	1.16	0.89	4.56	2.06	1.63	47.01	9.67	27.21	2.89	31.18	4.97	28.99	14.69	6.51	6.92	3.79	2.43
IAMSSC [67]	41.80	12.97	18.53	2.45	1.76	5.12	3.92	3.09	47.55	10.56	28.35	4.12	31.53	6.28	29.17	15.24	8.29	7.01	6.35	4.19
DepthSSC [78]	40.85	14.28	21.90	2.36	4.30	11.51	4.56	2.92	50.88	12.89	30.27	2.49	37.33	5.22	29.61	21.59	5.97	7.71	5.24	3.51
Symphonies* [23]	43.41	17.82	26.86	4.21	4.90	14.20	7.76	6.57	57.30	13.58	35.24	4.57	39.20	7.95	34.33	19.19	14.04	15.78	8.23	6.04
SGN-S [45]	46.22	17.71	28.20	2.09	3.02	11.95	3.68	4.20	59.49	14.50	36.53	4.24	39.79	7.14	36.61	23.10	14.86	16.14	8.24	4.95
SGN-T [45]	47.06	18.25	29.03	3.43	2.90	10.89	5.20	2.99	58.14	15.04	36.40	4.43	42.02	7.72	38.17	23.22	16.73	16.38	9.93	5.86
CGFormer [82]	48.07	20.05	29.85	3.42	3.96	17.59	6.79	6.63	**63.85**	17.15	**40.72**	5.53	42.73	8.22	38.80	24.94	16.24	17.45	10.18	6.77
SGFormer [19]	46.35	18.30	27.80	0.91	2.55	10.73	5.67	4.28	61.04	13.21	37.00	5.07	43.05	7.46	38.98	24.87	15.75	16.90	8.85	5.33
VoxDet (Ours)	**48.59**	**21.40**	**29.92**	**5.13**	**8.36**	**19.13**	**8.04**	**7.84**	62.83	**18.99**	40.10	**5.58**	**44.47**	**10.62**	**39.03**	**26.16**	**18.19**	**20.78**	**11.66**	**8.34**

Table 3: LiDAR-based results on SemanticKITTI [3] hidden test set with single frame (no extra temporal information) for fair comparison. Our VoxDet only uses point cloud input. The best and the second best results are in **bold** and underlined, respectively. Previous SoTA is [57].

Method	T	IoU	mIoU	road	sidewalk	parking	other-grnd.	building	car	truck	bicycle	motorcycle	other-veh.	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traf.-sign
SSCNet [55]		29.8	9.5	27.6	17.0	15.6	6.0	20.9	10.4	1.8	0.0	0.0	0.1	25.8	11.9	18.2	0.0	0.0	0.0	14.4	7.9	3.7
SSCNet-full [55]		50.0	16.1	51.2	30.8	27.1	6.4	34.5	24.3	1.2	0.5	0.8	4.3	35.3	18.2	29.0	0.3	0.3	0.0	19.9	13.1	6.7
TS3D [16]		29.8	9.5	28.0	17.0	15.7	4.9	23.2	10.7	2.4	0.0	0.0	0.2	24.7	12.5	18.3	0.0	0.1	0.0	13.2	7.0	3.5
TS3D/DNet [4]		25.0	10.2	27.5	18.5	18.9	6.6	22.1	8.0	2.2	0.1	0.0	4.0	19.5	12.9	20.2	2.3	0.6	0.0	15.8	7.6	7.0
LMSCNet [51]		55.3	17.0	64.0	33.1	24.9	3.2	38.7	29.5	2.5	0.0	0.0	0.1	40.5	19.0	30.8	0.0	0.0	0.0	20.5	15.7	0.5
LMSCNet-SS [51]		56.7	17.6	64.8	34.7	29.0	4.6	38.1	30.9	1.5	0.0	0.0	0.8	41.3	19.9	32.1	0.0	0.0	0.0	20.5	15.7	0.8
Local-DIFs [50]		57.7	22.7	67.9	42.9	**40.1**	11.4	40.4	34.8	4.4	3.6	2.4	4.8	42.2	26.5	39.1	2.5	1.1	0.0	29.0	21.3	17.5
JS3C-Net [69]		56.6	23.8	64.7	39.9	34.9	14.1	39.4	33.3	**7.2**	**14.4**	8.8	**12.7**	43.1	19.6	40.5	**8.0**	**5.1**	0.4	30.4	18.9	15.9
SSA-SC [75]		58.8	23.5	72.2	43.7	37.4	10.9	43.6	36.5	5.7	13.9	4.6	7.4	43.5	25.6	41.8	4.4	2.6	0.7	30.7	14.5	6.9
L2COcc-D [60]		45.3	18.1	68.2	36.9	34.6	**16.2**	25.8	28.3	4.5	4.9	3.3	7.2	26.2	11.9	32.0	2.1	2.4	0.3	21.6	9.6	9.5
L2COcc-L [60]		60.3	23.3	68.5	40.6	33.2	6.1	41.5	36.8	5.4	8.7	4.1	9.0	42.6	28.7	36.9	1.4	2.9	1.0	27.7	27.0	21.9
OccMamba [29]	✓	-	24.6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
VPNet [57]	✓	60.4	25.0	72.4	**44.3**	40.5	14.8	44.0	37.2	4.3	14.0	**9.8**	8.2	45.3	**30.9**	42.1	4.9	2.0	**2.4**	32.7	17.1	8.8
VoxDet-L (Ours)		**63.0**	**26.0**	**73.0**	43.6	37.5	10.3	**44.5**	**37.7**	6.6	9.9	6.2	11.8	**45.9**	30.7	**43.5**	2.7	3.2	1.3	**34.0**	**27.8**	**23.7**

and 26.0 mIoU, significantly surpassing state-of-the-art methods [60, 57]. Our method achieves the best on large-scale objects like *building, vegetation*, middle-scale objects like *car*, and small-scale objects such as *pole, traffic-sign*. ***Additionally, our method is also state-of-the-art in terms of model efficiency.*** Our VoxDet-L (with only 22.1 M parameters) significantly surpasses the latest counterpart VPNet (with 35.8 M parameters) [57] published on NeurIPS 2024 and L2COcc-L (with 36.2 M parameters) published on ArXiv 25 on IoU, mIoU, and model parameters.

Notably, with our effective regression-include formulation, VoxDet-L achieves a significantly higher IoU of 63.0 (the completion metric) without using extra labels/data/temporal information/models, ranking 1st on the CodaLab leaderboard[2] (Updated on May 22). This capability is essentially crucial for autonomous navigation in preventing collisions with obstacles.

5.2 Quantitative Study

Ablation Study. Tab. 4 presents the ablation study, highlighting the following insightful observations. ***Line (b):*** Compared with the baseline, introducing the REG branch in TDP gives a significant 2.71 IoU gains owing to the better instance-level perception. Interestingly, the limited 0.14 mIoU shows that the REG does not contribute to the semantic discrimability, aligning with our motivation. ***Line (c):*** After deploying the CLS with instance-level aggregation, a significant 1.43 mIoU gain can be

[2]Online leaderboard: https://codalab.lisn.upsaclay.fr/competitions/7170#results

Table 4: Ablation study on each key module.

	TDP		SVE		IoU	mIoU	N_{param}
	REG	CLS	REG	CLS	(%)	(%)	(M)
(a)					42.71	16.28	48.7
(b)	✓				45.42	16.42	48.9
(c)	✓	✓			46.79	17.85	49.4
(d)	✓	✓	✓		47.14	18.02	51.1
(e)	✓	✓		✓	47.08	18.27	51.1
(f)	✓	✓	✓	✓	**47.36**	**18.73**	52.8

Table 5: Further analysis of varied designs.

	Detailed designs	IoU (%)	mIoU (%)
TDP	Δ guidance → Self-attention	46.82	18.15
	Eq. 6 → Weighted fusion	47.01	18.38
SVE	Eq. 1 → Average pooling	47.18	18.49
	DefConv → Conv	46.92	18.08
	TPV → ResBlock	47.02	18.32
	Task-decoupled → Task-shared	46.81	17.88
	Full model	**47.36**	**18.73**

Table 6: Efficiency comparison with SoTA methods.

Method	N_{param} ↓	T_{inf} ↓	IoU (%) ↑	mIoU (%) ↑
OccFormer [85]$_{[ICCV'23]}$	214	199	36.42	13.50
StereoScene [26]$_{[IJCAI'24]}$	117	258	43.85	15.43
CGFormer [82]$_{[NeurIPS'24]}$	122	205	45.99	16.89
SGFormer [19]$_{[CVPR'25]}$	126	-	45.01	16.68
ScanSSC [2]$_{[CVPR'25]}$	145	261	45.95	17.12
VoxDet (Ours)	**53**	**159**	**47.36**	**18.73**

Table 7: Results with monocular depth.

Method	IoU (%)	mIoU (%)
VoxFormer-S [35]	38.68	10.67
VoxFormer-T [35]	38.08	11.27
Symphonize [23]	38.37	12.20
OccFormer [85]	36.50	13.46
CGFormer [82]	41.82	14.06
VoxDet (Ours)	**43.92**	**16.35**

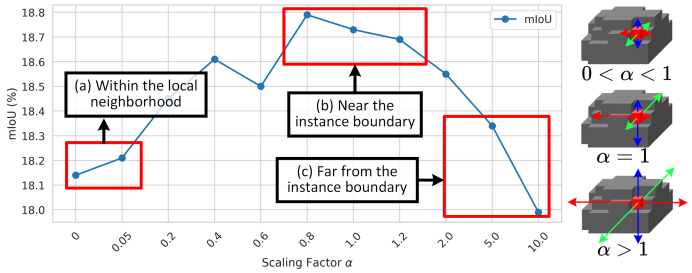


Figure 5: Justifying the aggregation design using voxels at regressed positions $\mathbf{V}_\delta$. By modulating regressed offsets Δ with a scaling factor $\alpha \in \mathbb{R}$, for each voxel $\mathbf{V}_{i,j,k}$, we select the voxels $\mathbf{V}_{\tilde{\delta}}$ based on modulated offsets $\tilde{\Delta} = \alpha \Delta_{i,j,k}$, where increasing α will select voxels at larger distances and vice versa.

observed, revealing the importance of instance guidance in voxel prediction. ***Line (d-f):*** By gradually decoupling the volume, both metrics improve progressively, eliminating the task misalignment.

Further Analysis. In Tab. 5, we delve into each module with different design variants. For **TDP**, replacing the Δ-guided instance-level aggregation (Eq. 6) with learned deformable self-attention and soft-weights both decreases the performance, revealing the necessity of explicit instance guidance. *The Δ guidance and attentive design may relieve the negative influence of regression outliers far from instances, consistent with Fig. 5.* In **SVE**, we observe a consistent decline by replacing dense projection, DefConv, TPV, and decoupling with average pooling, Conv, ResBlock, and task-shared designs, respectively. This can verify that our spatial task-decoupling in the TPV space is optimal.

Model Efficiency. In Tab. 6, we compare with state-of-the-art works in parameters (N_{param}) and inference time (T_{inf}). VoxDet uses significantly fewer parameters (M) and less inference time (ms), setting new records on all IoU metrics, highlighting the effectiveness of our new formulation.

Comparison with Monocular Depth. In Tab. 7, we make a comparison with state-of-the-art methods by using monocular depth [5]. Our VoxDet also achieves the best results on both metrics, surpassing the previous best entry with 2.10 IoU and 2.29 mIoU, showing significant robustness on depth.

Exploring Instance-level Aggregation. Fig. 5 carefully analyzes our aggregation designs (Eq. 6). Instead of directly aggregating regressed voxels $\mathbf{V}_\delta$ with Δ, we modulate it by a scaling factor $\alpha \in \mathbb{R}$, and aggregate at the scaled offset $\tilde{\Delta} = \alpha\Delta$, revealing three observations. ***(1) Local aggregation does not work*** ***($\alpha \to 0$).*** When α approaches zero, performance degrades noticeably. Naively aggregating neighborhoods yields no benefit, as it degenerates to convolutional kernels with local receptive fields. ***(2) Near-boundary aggregation helps*** ***($\alpha \approx 1$).*** As α grows toward 1.0, performance steadily improves, indicating that the boundary contributes more informative signals for dense perception. Notably, $\alpha = 0.8$ slightly outperforms our default of $\alpha = 1.0$, likely by suppressing outliers outside instances. ***(3) Outside-instance aggregation hurts*** ***($\alpha > 1$).*** When α exceeds 1.0 substantially, performance declines sharply, revealing the negative impact of aggregating voxels beyond the instance extent.

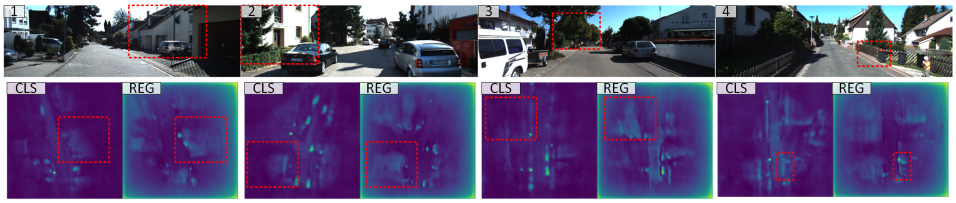


Figure 6: Visualization of the decoupled feature for classification (CLS) and regression (REG).

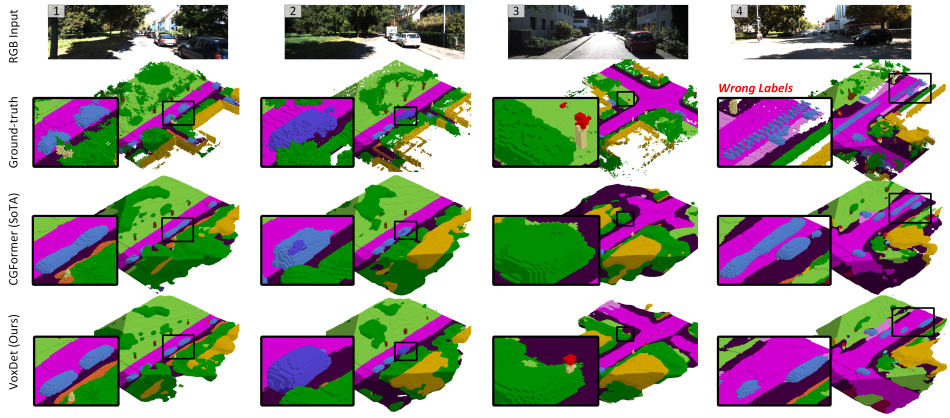


Figure 7: Qualitative comparison with state-of-the-art method [82]. Zoom in for a better view.

5.3 Qualitative Study

Disentangled Volume. In Fig. 6, we visualize the decoupled feature volume (Eq. 3) at Bird's Eye View (BEV) view for the classification (CLS) and regression (REG), revealing the following three insightful observations. **(1)** Different from CLS, focusing on semantically informative local regions, REG naturally seeks to discover the more complete boundary of instances. This capacity is critical, as completing the scene is the key focus. **(2)** Although REG has not been trained with semantic supervision, it can discover informative cues of potential objects (4^{th} sample) with orthogonal effects with CLS. **(3)** We can see the four borders of the BEV map are activated, as the 3D volume borders are unified boundaries of adjacent large-scale instances, like vegetation and buildings.

Semantic Occupancy Prediction. Fig. 7 visually compares our VoxDet with state-of-the-art method [82]. VoxDet shows noticeable gains in geometric completion, such as complete *car* borders in the 1^{st} sample, better infers instance-level semantics, such as the *truck* in the 2^{nd} sample, and can correctly detect the *challenging yet important traffic sign* (3^{rd} sample). This is mainly owing to the proposed instance-centric formulation, which enhances spatial perception and semantic understanding of object instances. Notably, in the 4^{th} sample, instead of wrongly fitting wrong labels, VoxDet gives more reasonable predictions on dynamic objects, which is further discussed in the Appendix.

6 Conclusion

In this work, we first reveal an ever-overlooked free lunch in occupancy labels: the voxel-level class label has implicitly provided the instance-level insight thanks to its occlusion-free nature. Then, we propose **VoxNT**, a simple yet effective algorithm that freely converts class labels to the instance-level offsets. Based on this, we propose **VoxDet**, a new detection-based formulation for instance-centric prediction. VoxDet first spatially decouples 3D volumes to generate task-specific features, learning spatial deformations in the tri-perceptive space. Next, it adopts a task-decoupled predictor to generate instance-aware predictions guided by the regressed 4D offset field. Extensive experiments reveal a lot of insightful phenomena for the following works and verify the state-of-the-art role of VoxDet.

6.1 Acknowledgment

We would like to express our gratitude to Reyhaneh Hosseininejad, Yasamin Borhani, Yuanfan Zheng, and Hengyu Liu for their insightful discussions, which contributed to the improvement of this work. Additionally, we gratefully acknowledge the financial support provided by Valeo.

Appendix

We provide comprehensive supplementary materials to clarify novelty, practical applications, border impacts, experimental justifications, and future directions, potentially inspiring the following works, which are organized as follows. **Bold** highlights the primary sections with more insights.

A Additional Quantitative Results

A.1 Robustness Analysis of Multiple Runs

In Fig. 8, we report the results of multiple runs (5 times independent experiments) to justify the robustness of our VoxDet. The curves summarize the IoU and mIoU results for each epoch on the SemanticKITTI validation set. To better illustrate the difference, we also demonstrate the results of the previous state-of-the-art method, CGFormer [82], using the officially released training log.

It can be observed that our VoxDet achieves impressive robustness with multiple runs, giving very similar performance on the IoU and mIoU in different runs. Note that there is some tradeoff between IoU and mIoU metrics, i.e., some runs achieve slightly higher IoU while sacrificing a little mIoU. Additionally, our method achieves visually significant gains over CGFormer on both the robustness and IoU/mIoU metrics, highlighting the strength VoxDet. ***Hence, due to the superior effectiveness, robustness, and efficiency, we believe that VoxDet is qualified to serve as a powerful, lightweight, and efficient baseline model for the following works.***

A.2 How to Define Object Instances?

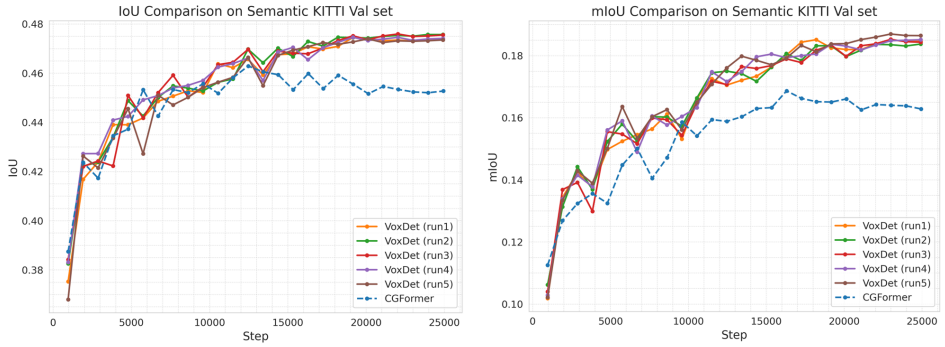


Figure 8: Robustness analysis of our VoxDet with multiple runs. We report the per-epoch validation results. The results from the official training log (not our reproduced results) given by the previous state-of-the-art method, CGFomrer [82], are also listed for comparison. We can observe visually significant improvements in both performance and robustness of our method.

Table 9: Quantitative results on SemanticKITTI [3] validation set. T indicates using extra temporal information. The best and the second best results are in bold and underlined, respectively.

Method	T	IoU	mIoU	road (15.30%)	sidewalk (11.13%)	parking (1.12%)	other-grnd. (0.56%)	building (14.1%)	car (3.92%)	truck (0.16%)	bicycle (0.03%)	motorcycle (0.03%)	other-veh. (0.20%)	vegetation (39.3%)	trunk (0.51%)	terrain (9.17%)	person (0.07%)	bicyclist (0.07%)	motorcyclist (0.05%)	fence (3.90%)	pole (1.29%)	trf.-sign (0.08%)
MonoScene [6]		36.86	11.08	56.52	26.72	14.27	0.46	14.09	23.26	6.98	0.61	0.45	1.48	17.89	2.81	29.64	1.86	1.20	0.00	5.84	4.14	2.25
TPVFormer [22]		35.61	11.36	56.50	25.87	20.60	0.85	13.88	23.81	8.08	0.36	0.05	4.35	16.92	2.26	30.38	0.51	0.89	0.00	5.94	3.14	1.52
OccFormer [85]		36.50	13.46	58.85	26.88	19.61	0.31	14.40	25.09	25.53	0.81	1.19	8.52	19.63	3.93	32.62	2.78	2.82	0.00	5.61	4.26	2.86
IAMSSC [67]		44.29	12.45	54.55	25.85	16.02	0.70	17.38	26.26	8.74	0.60	0.15	5.06	24.63	4.95	30.13	1.32	3.46	0.01	6.86	6.35	3.56
VoxFormer [35]		44.02	12.35	54.76	26.35	15.50	0.70	17.65	25.79	5.63	0.59	0.51	3.77	24.39	5.08	29.96	1.78	3.32	0.00	7.64	7.11	4.18
VoxFormer [35]	✓	44.15	13.35	53.57	26.52	19.69	0.42	19.54	26.54	7.26	1.28	0.56	7.81	26.10	6.10	33.06	1.93	1.97	0.00	7.31	9.15	4.94
Symphonize [23]		41.92	14.89	56.37	27.58	15.28	0.95	21.64	28.68	20.44	2.54	2.82	13.89	25.72	6.60	30.87	3.52	2.24	0.00	8.40	9.57	5.76
HASSC [61]		44.82	13.48	57.05	28.25	15.90	**1.04**	19.05	27.23	9.91	0.92	0.86	5.61	25.48	6.15	32.94	2.80	**4.71**	0.00	6.58	7.68	4.05
H2GFormer [64]		44.57	13.73	56.08	29.12	17.83	0.45	19.74	28.21	10.00	0.50	0.47	7.39	26.25	6.80	34.42	1.54	2.88	0.00	7.24	7.88	4.68
H2GFormer [64]	✓	44.69	14.29	57.00	29.37	21.74	0.34	20.51	28.21	6.80	0.95	0.91	9.32	27.44	7.80	36.26	1.15	0.10	0.00	7.98	9.88	5.81
SGN [45]		43.60	14.55	59.32	30.51	18.46	0.42	21.43	31.88	13.18	0.58	0.17	5.68	25.98	7.43	34.42	1.28	1.49	0.00	9.66	9.83	4.71
SGN [45]	✓	46.21	15.32	59.10	29.41	19.05	0.33	25.17	33.31	6.03	0.61	0.46	9.84	**28.93**	9.58	38.12	0.47	0.10	0.00	9.96	13.25	7.32
CGFormer [82]		45.99	16.87	65.51	32.31	20.82	0.16	23.52	34.32	19.44	**4.61**	2.71	7.67	26.93	8.83	39.54	2.38	4.08	0.00	9.20	10.67	7.84
HTCL [25]	✓	45.51	17.13	63.70	32.48	**23.27**	0.14	24.13	34.30	20.72	3.99	2.80	11.99	26.96	8.79	37.73	2.56	2.30	0.00	11.22	11.49	6.95
VoxDet (Ours)		**47.36**	**18.73**	**65.55**	**34.22**	20.88	0.04	**25.79**	**34.50**	**31.05**	3.95	**5.14**	**14.65**	**28.93**	**10.20**	**41.27**	**4.48**	3.14	0.00	**11.73**	**12.19**	**8.28**

In Tab. 8, we explore different definitions of objects by removing the instance-level regression for specific classes, including (1) background, including road, sidewalk, etc (2) empty class. We empirically find that a generalized definition of objects, i.e., all the semantic classes, performs best. This is different from the intuition in the 2D image domain, which only considers objects with well-defined shapes.

Table 8: Analysis of object definitions by removing instance-level regression for specific classes.

Setting	IoU	mIoU
w/o. regress background	46.78	18.19
w/o. regress empty	47.08	18.05
Full model	**47.36**	**18.73**

A.3 Results on SemanticKITTI Validation Set

Tab. 9 presents the comparison on the SemanticKITTI validation set. Our VoxDet archives the best results of 47.36 IoU and 18.73 mIoU, surpassing the second-best counterpart [25] with a noticeable 1.60 mIoU gains. Compared with the previous state-of-the-art method [82] with 45.99 IoU and 16.87 mIoU, our VoxDet gives a 1.37 IoU and 1.86 mIoU gains, verifying the superior effectiveness. Besides, VoxDet achieves the best and second-best results in 17 of the 19 classes, which shows its effectiveness. Specifically, our method performs well on object instances, achieving the best on *building, car, truck, motorcycle, traffic sign, etc*, revealing the superior instance-centric learning.

A.4 Sensitivity Analysis

In this section, we further give a detailed analysis of the hyperparameters used in VoxDet. The experiments are conducted with the same random seed to ensure fair comparison. The results are reported on the SemanticKITTI validation set as the unified practice.

Table 10: Analysis on instance-level aggregation layers N.

N	IoU	mIoU
1	47.06	18.20
2	**47.38**	18.32
3	47.34	18.54
4	47.36	18.73
5	**47.39**	**18.79**

Table 11: Analysis on the loss weight λ deployed on $\mathcal{L}_{\text{occ}}^{\text{aux}}$.

λ	IoU	mIoU
0.1	47.15	18.53
0.2	47.36	**18.73**
0.4	**47.43**	18.69
0.8	47.28	18.59
1.0	47.39	18.42

Table 12: Analysis on the loss weight λ_{reg} deployed on $\mathcal{L}_{\text{Reg}}$.

λ_{reg}	IoU	mIoU
0.5	47.19	18.46
1.0	47.36	**18.73**
1.5	47.48	18.68
2.0	47.55	18.59
2.5	**47.68**	18.55

Table 13: Comparison of model efficiency and accuracy with SoTA on SemanticKITTI test set.

Venue	CVPR' 23	CVPR' 23	CVPR' 23	ICCV'23	IJCAI' 24	NeurIPS' 24	CVPR' 25	
Method	TPVFormer [22]	VoxFormer [35]	Symphonize [23]	OccFormer [85]	StereoScene [26]	CGFormer [82]	ScanSSC [2]	**VoxDet**
Param. (M) ↓	107	59	59	214	117	122	145	**53**
Inf. Time (ms) ↓	207	204	216	199	258	205	261	**159**
IoU ↑	34.25	42.95	42.19	34.53	43.34	44.41	44.54	**47.27**
mIoU ↑	11.26	12.20	15.04	12.20	15.36	16.63	17.40	**18.47**

Number of Instance-level Aggregation Layers. Tab. 10 reports the performance changes as we vary the number of aggregation layers N. Performance improves steadily with increasing N, confirming the benefit of more effective aggregation. Although extending to five layers (one more than our default $N = 4$) yields a slight additional gain, the marginal improvement will lead to extra computational overhead. We therefore adopt $N = 4$ as our default setting.

Weight of Auxiliary Loss. Tab. 11 presents the sensitivity of our model to the auxiliary segmentation loss weight λ in $\mathcal{L}_{\text{occ}}^{\text{aux}}$, which governs the strength of voxel-level supervision. Overall, mIoU remains relatively stable across a wide range of λ values: as we increase λ, performance steadily improves, peaking at $\lambda = 0.2$, and then gradually declines for larger weights. This trend indicates that a moderate auxiliary loss provides beneficial guidance for voxel-wise feature learning, while an overly large weight interferes with the subsequent instance-centric optimization. Consequently, we adopt $\lambda = 0.2$ as our default setting, as it yields the best mIoU.

Loss weight of Regression. In Tab. 12, we add a loss weight on the regression λ_{reg} and explore the effect of varying the loss weight. A moderate increase in λ_{reg} yields further IoU gains, likely due to improved fitting of instance contours. However, excessively large weights cause a slight mIoU drop, probably from conflicts among the combined loss terms. We therefore set $\lambda_{\text{reg}} = 1.0$ to balance the effect and eliminate the term in the main paper for convenience.

A.5 Model Efficiency

Tab 13 presents a comprehensive comparison between our VoxDet and current state-of-the-art methods on the SemanticKITTI test set, evaluating model size, inference speed, and prediction performance. VoxDet requires the fewest learnable parameters, achieves the fastest inference time, and attains the highest per-class IoU as well as the best overall mIoU. Compared to its strongest competitor [2], our approach slashes the parameter count, accelerates inference to real-time performance, and delivers a notable gain in mIoU. These results underscore the strength of our effective formulation, giving a lightweight yet powerful framework that simultaneously advances speed, efficiency, and accuracy.

B Voxel-to-Instance (VoxNT) Trick

B.1 VoxNT Trick Can Freely Understand Instance Scales

In Fig. 9, we conduct a statistical analysis on the instance scale along different axes based on the proposed Voxel-to-Instance (VoxNT) Trick. To better visualize the scale patterns, we visualize the distribution for different classes (in different colors), highlighting their specific scale patterns. The resolution of the whole scene is $256 \times 256 \times 32$. To better explore the instance-level geometries, we sum up the positive and negative directions of the 4D offset field Δ to represent the scale information

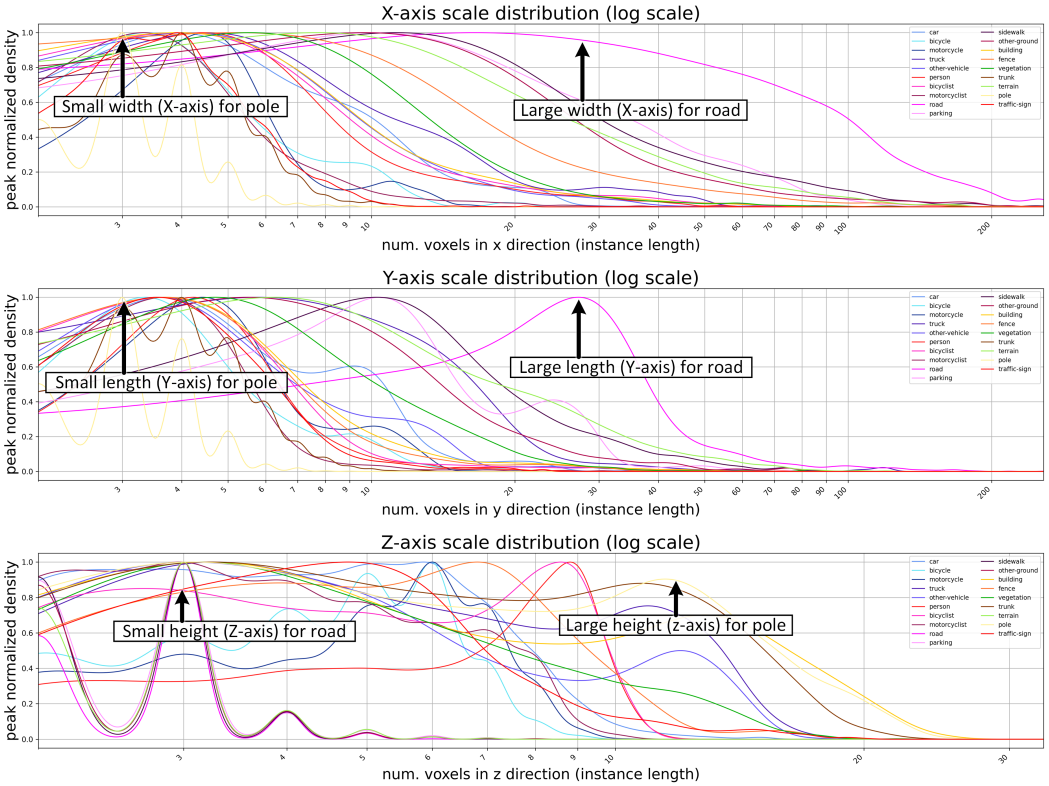


Figure 9: Instance-level scale distribution in X, Y, Z axis given by our voxel-to-instance trick. We randomly sample voxels in all classes and calculate the instance scale (l) with coupled offset terms (positive and negative directions), e.g., $l^x = \delta^{x^+} + \delta^{x^-}$ in X-axis. The horizontal axis represents the scale (the number of voxels) on a log scale. The scale of the whole scene in X, Y, Z is $256, 256, 32$ respectively. The vertical represents the number of samples, which is peak normalized to $[0, 1]$ by dividing the maximum number of each class for a better view. Zoom in for a better view.

in different axes. This process is denoted as follows,

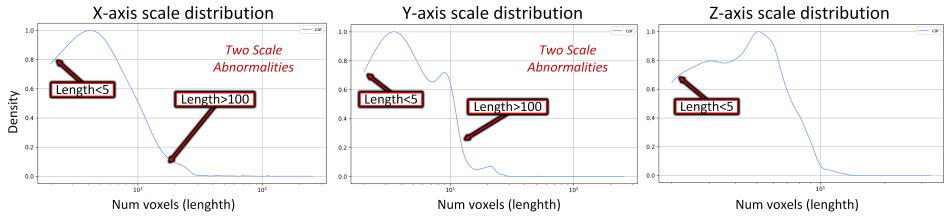
$$\begin{aligned} l^x &= \delta^{x^+} + \delta^{x^-}; \\ l^y &= \delta^{y^+} + \delta^{y^-}; \\ l^z &= \delta^{z^+} + \delta^{z^-}. \end{aligned} \tag{9}$$

Here $\{\delta^{x^+}, \delta^{x^-}, \delta^{y^+}, \delta^{x^-}, \delta^{z^+}, \delta^{z^-}\}$ indicates the offset in the six directions. Surprisingly, we find that Δ is able to freely give sufficient instance-level cues beyond the spatially agnostic semantic labels: the essential scale information. Besides, the geometric scale distribution of different classes also demonstrates different unique patterns, greatly aligning with the real-world environment. The following are intuitive observations.

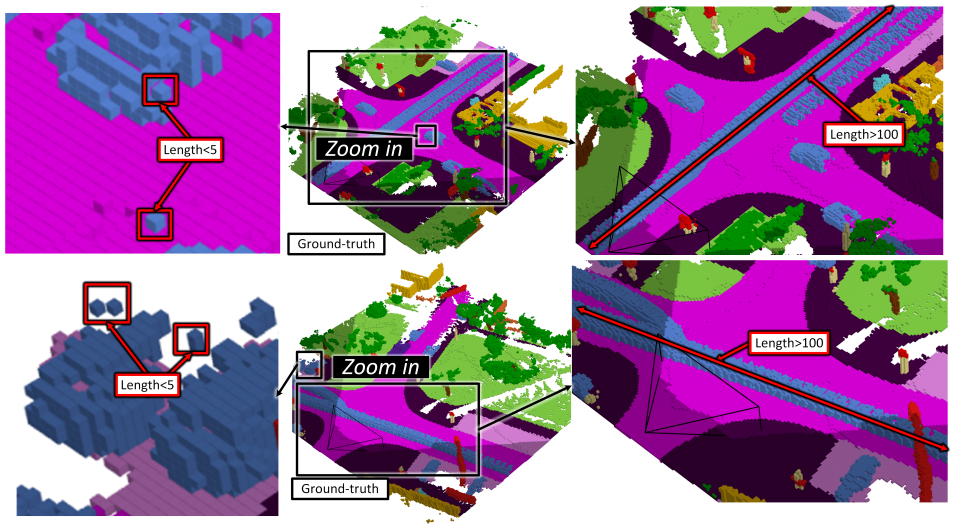
(1) For tall instances, such as *pole* in the light yellow color, the scales tend to give small values in X and Y axes (usually lower than six voxel width/length), and provide a large scale on the Z axis (e.g., larger than 10-voxel height.). This is aligned with the comment scene as the width and length of these things are small, while the height is significant.

(2) For the flat items, such as *road* in the pink color, we can see a significant value in X and Y axes (usually larger than 30 voxel width/length), while giving a small scale on the Z axis (smaller than 5 voxels), which also aligns with our intuition.

(3) For some large objects, such as *vegetation* in the green color, we can see that the scale in all three axes can be relatively significant, aligning with the real-world environment.



(a) The scale distribution of dynamic object Car in three axis given by VoxNT trick.



(b) Illustration of the two scale abnormalities of dynamic object Car.

Figure 10: More observations from VoxNT. (a) Scale distribution (same as App. B) of the car along the X, Y, Z axis. **We can see two types of abnormality: minimal and large lengths.** (b) Visualizting the abnormality with ground-truth. The left shows the isolated voxels leading to the minimal-scale abnormality. The right shows the large-length abnormality caused by the traces of moving objects.

Hence, these observations further highlight the value of the proposed VoxNT trick, which can convey extensive valuable information in geometry at the instance level. Note that this information is not available in the original voxel-level class labels.

B.2 VoxNT Trick Can Freely Identify Wrong Labels

Observe the Scale Abnormality of Dynamic Objects with VoxNT Trick. We first study the *car* class[3], the most prevalent and dynamic object category in autonomous driving. Fig. 10 (a) demonstrates the distribution of the instance scale along different axes, generated by our VoxNT trick. Given the whole scale of $256 \times 256 \times 32$ in the scene, we find that the scale of the car is problematic: In the X and Y axis, there are lots of samples with tiny scale with **length less than 5** and huge scale with **larger than 100**. In the Z axis, many samples have **length less than 5**. Note that the typical scale of a car is less than $30 \times 20 \times 10$, indicating that the ground-truth has significant abnormality.

Delve into the Abnormality with Label Visualization. To study the essential reason, we visualize the ground-truth voxels in Fig. 10 (b). We can ground the observed issues mentioned above according to the visualized ground truth, which is voxelized from point clouds. In the left sub-figure, we find that the reason for the samples with abnormally small length is **isolated voxels**. In the right sub-figure, we can see that the failed filter object dynamics lead to abnormally large lengths.

B.3 VoxNT Trick Can Freely Eliminate the Influence of Wrong Labels in Training

We can see that the scale information in our 4D offset field has a valuable property for solving this issue, which, for the first time, makes the abnormality identification realistic. In this section, we discuss some tricky uses of the proposed VoxNT trick to inspire the follow-up works. Note that

[3]Cars serve as the key target and dominate the evaluation of detection benchmarks [10, 62]

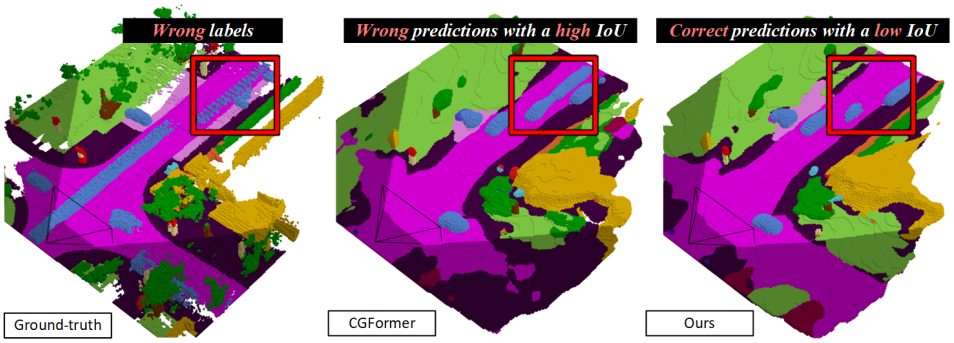


Figure 11: Illustration of the problems in the existing evaluation metrics. **Left:** The ground-truth wrongly labels the car with a sequential afterimage. **Middle:** Previous works wrongly fit these errors by predicting a long car. **Right:** Differently, our VoxDet can give more reasonable predictions without fitting the error, but the car IoU in this sample is worse due to the wrong labels.

these operations improve a more reasonable prediction but cannot improve the mIoU evaluation, because the ground-truth is noisy (see App. B.4). In brief, the key idea is to use scale information to identify the voxel with an offset that is too large or too small. Specifically, to identify the impact of the isolated voxels, we can use a binary voxel-level mask $\mathbf{M}^{\text{min}}_{i,j,k} \in \{0,1\}^{X\times Y\times Z}$ to **identify the voxels that have abnormally small offsets in all axes**:

$$\mathbf{M}^{\text{min}}_{i,j,k} = \begin{cases} 1.0, & (l^x_{i,j,k} < K^x_{\text{min}}) \cap (l^y_{i,j,k} < K^y_{\text{min}}) \cap (l^y_{i,j,k} < K^z_{\text{min}}); \\ 0.0, & \text{otherwise}, \end{cases} \tag{10}$$

where $K_{\text{min}} \in \mathbb{N}^3$ is a scale threshold for the three axes for measuring the removed minimal scale and can be empirically set to 3. Similarly, to filter out the huge-scale samples, e.g., the car in Fig. 10, a similar mask can be deployed $\mathbf{M}^{\text{max}}_{i,j,k} \in \{0,1\}^{X\times Y\times Z}$ to **identify the voxels with abnormally large offsets appearing in one of the axes**. This can be written as follows,

$$\mathbf{M}^{\text{max}}_{i,j,k} = \begin{cases} 1.0, & (l^x_{i,j,k} \geq K^x_{\text{max}}) \cup (l^y_{i,j,k} \geq K^y_{\text{max}}) \cup (l^y_{i,j,k} \geq K^z_{\text{max}}) \\ 0.0, & \text{otherwise}, \end{cases} \tag{11}$$

where K_{max} is the threshold filtering out the large lengths. Another class-based mask can also be adopted $\mathbf{M}^{\text{max}}_{i,j,k}$ with class labels $Y_{i,j,k}$ and desired class K_c for the wrong label filtering, such as car.

By using these two types of masks, we can clearly identify and localize those wrong voxels with the specific positions (i,j,k) in the volumes. To remove their influence on training, it is possible to directly refine the ground-truth labels by ignoring these voxels with a straightforward yet effective label transformation. In practice, it can be implemented as the following equations,

$$Y^{\text{refined}}_{i,j,k} = \begin{cases} 255, & (\mathbf{M}^{\text{max}}_{i,j,k} = 1) \cup (\mathbf{M}^{\text{min}}_{i,j,k} = 1) \cap (Y_{i,j,k} = K_{car}); \\ K_{car}, & (\mathbf{M}^{\text{max}}_{i,j,k} = 0) \cap (\mathbf{M}^{\text{min}}_{i,j,k} = 0) \cap (Y_{i,j,k} = K_{car}); \\ Y_{i,j,k}, & \text{otherwise}. \end{cases} \tag{12}$$

Here, the key idea is to set the label as ignored (255) if the scale of the car voxel is too large or too small. This can effectively remove lots of wrong predictions in the dynamic car.

B.4 Rethink the Evaluation on Dynamic Objects

Based on the observation about wrong labels, we further delve into the evaluation on the dynamic object *car*. In Fig. 11, we visualize the ground-truth of the sample from SemanticKITTI [3] validation (Left), the corresponding prediction from the state-of-the-art method [82] (Middle), and the prediction from our method (Right). We can see that previous works are prone to overfitting these wrong labels while generating higher results on the IoU metric for the car. Differently, our method gives a more reasonable prediction. However, this advantage cannot be demonstrated by using the conventional IoU metric due to the wrong labels. We hope this observation can inspire the following works to notice and address this issue, thereby pushing forward the community.

C Experimental Settings

C.1 Datasets and Metrics

Benchmark Setting. Following the unified setting in this community, we evaluate our approach on two benchmarks: SemanticKITTI [4] and SSCBench-KITTI-360 [34], which are derived from the original KITTI Odometry [17] and KITTI-360 [38] datasets, respectively. In all experiments, we follow the unified setup [35, 82, 23], restricting to the frustum of size 51.2 m in the forward direction, ± 25.6 m laterally, and 6.4 m vertically above the sensor. This volume is voxelized into a grid of size $256 \times 256 \times 32$, with each voxel measuring 0.2 m on a side. Details are as follows.

(1) SemanticKITTI consists of 22 sequences (00–21) of LiDAR scans and synchronized stereo images. We adopt the standard split of 10 sequences for training (00–07, 09–10), one sequence for validation (08), and 11 sequences for online evaluation on the hidden test server (11–21). Input images are provided at 1226×370 resolution, and ground-truth occupancy grids are annotated with 20 labels (19 semantic classes + 1 empty class).

(2) SSCBench-KITTI-360 is a recently released extension that re-labels a subset of the KITTI-360 sequences. It comprises 9 sequences in total, of which 7 are used for training, 1 for validation, and 1 held out for final testing. RGB images are captured at 1408×376 resolution, and each voxel is annotated with one of 19 labels (18 semantic categories + 1 free-space class).

Evaluation Metrics. We assess performance along two complementary axes, i.e., geometry completion and semantic completion, using the Intersection over Union (IoU) and mean Intersection over Union (mIoU) metrics, for occupied voxel grids and voxel-wise semantic predictions. All reported results follow the evaluation protocols from previous works, including the online test-server evaluations for SemanticKITTI [4], ensuring a fair comparison with prior methods [35, 22, 82].

C.2 Implementation Details

Backbone Network. Following the main stream of occupancy prediction works [35, 86], we use ResNet-50 [21] as the 2D image feature extractor. We follow the unified SSC settings in the recent 2-year publications for the depth estimation and use MobileStereoNet [53] and Adabins [5] as the depth estimators. Note that the compared methods use the same depth images for the fair comparison.

View Transformation. In the main paper, the 2D-to-3D view transformation follows [82, 2]. Specifically, given the input image $\mathbf{I}$, we first extract 2D image feature $\mathbf{F}^{\text{2D}}$ and depth map $\mathbf{Z}$. Then, we adopt a depth refinement module that takes the 2D feature $\mathbf{F}^{\text{2D}}$ and depth $\mathbf{Z}$ as input. It first estimates the depth distribution via LSS [47], and then generates voxel queries $\mathbf{V_Q} \in \mathbb{R}^{X \times Y \times Z \times C}$. Here, (X, Y, Z) is spatial resolution $128 \times 128 \times 16$, and $C = 128$ is the channel.

Based on this, the pixel (u, v) can be transformed to the 3D point (x, y, z) using the camera intrinsic matrix ($\in \mathbb{R}^{4\times4}$) and extrinsic matrix ($\in \mathbb{R}^{4\times4}$), termed the query proposals $\mathbf{Q}$ in the projected voxel space. Finally, the 3D deformable cross-attention is deployed to query the information from 2D image to the 3D voxel space. The number of deformable cross-attention layers is 3, and the number of sampling points around each reference point is set to 8. This can generate the 3D feature volume with dimensions of $128 \times 128 \times 16$ and 128 channels, which is the $\mathbf{V}$ in the main paper. For fair comparison, these operations in view transformation are the same as previous works [82, 2, 60]. Kindly refer to [82] for more details.

VoxDet. The number of instance-driven aggregation layers N is set to 4. The loss weight terms of λ and β are empirically set to 1.0 and 0.2, respectively. For the task-shared voxel encoder in our spatially-decoupled voxel encoder, we directly use the encoder part of the conventional 3D UNet in other works [82]. The regression branch is simply deployed as `Conv` $\rightarrow$ `GroupNorm` $\rightarrow$ `ReLU` $\rightarrow$ `Conv`, with the last convolution outputting 6 channels. Code and models will be released.

Model Training. We train our VoxDet with a batch size of 4 using AdamW [43] optimizer. Following [82], the cosine annealing schedule is adopted, with the first 5% iterations of warm-up, maximum learning rate of 3×10^{-4}, weight decay of 0.01 and $\beta_1 = 0.9$, $\beta_2 = 0.99$, The experiments are conducted on 2 NVIDIA A100 GPUs (40G) with two samples for each GPU. The final prediction has dimensions of $128 \times 128 \times 16$, which is upsampled to $256 \times 256 \times 32$ through trilinear interpolation to align with the ground truth. We use the VoxNT trick to remove the wrong labels as Eq. 12, with the scale threshold set to 30 for each axis. The efficiency experiments are conducted on a NVIDIA

4090 (commercial GPU), considering the more practical deployment property. The training requires around 19 GB of memory per sample, with about 9.0 training hours for SemanticKITTI and 18 hours for SSCBench-KITTI360, which is very friendly for the research groups with commercial GPUs.

C.3 Algorithmic details of the VoxNT Trick

We present the detailed process of our VoxNT Trick in Algorithm 1 with PyTorch-style pseudo code. By calling the function $\mathrm{compute\_all\_direction\_distances}()$, this algorithm aims to generate the ground-truth of 4D offset field $\hat{\Delta}$ only using the per-voxel class labels $Y \in \mathbb{N}^{X \times Y \times Z}$ (gt_occ). In Y, each item satisfies $0 < Y_{i,j,k} < K$ with K representing the number of classes.

In brief, for every scanning direction $\mathbf{d} \in \{x^+, x^-, y^+, y^-, z^+, z^-\}$, we begin by initializing a zero-valued matrix $R \in \mathbb{N}^{X \times Y \times Z}$. Then, as we iteratively traverse the voxels along the chosen direction d, we compare consecutive voxels. Given the two adjacent voxels $\mathbf{V}_i$ and $\mathbf{V}_{i+1}$, suppose the class label of them matches with each other: $Y_i = Y_{i+1}$ considering the scanning direction, the corresponding entry in R is incremented by one; otherwise, the accumulation halts once a class change is detected, revealing approaching the instance boundary.

This trick efficiently captures the spatial continuity of object instances and provides reliable ground truth offsets for regression. Considering the symmetrical property between the two scanning directions (forward and backward), we deploy a flip operation along the selected axis for efficient implementation. Based on this, we can generate the 4D tensor saving these relative distances in 6 directions, where each dimension is then normalized into $[0, 1]$ divided by the volume size X, Y, Z accordingly. Thus, we can obtain a normalized offset field $\hat{\Delta}$ ground-truth from semantic labels.

Implementing VoxDet with LiDAR Input. VoxDet is designed on the lifted 3D volumes, which can be effortlessly transferred to a LiDAR-based pipeline. To further analyze this flexibility, we deploy a LiDAR-based VoxDet, denoted as **VoxDet-L**. We achieve this by replacing the 2D-to-3D lifting with a simple point cloud encoder. This can directly generate the 3D feature volume $\mathbf{V}$ with 3D ResNet-50, using LiDAR point cloud as input. The 2D image encoder, depth estimator, and view transformation are removed. *All the implementations only use a single model without multi-frame distillation and only a single frame input without temporal information,* which will be open-sourced.

D Discussion

D.1 Difference with Related Works

Occupancy prediction assisted with 3D object detection. Some existing works have attempted to assist the voxel prediction with 3D object detection [46, 77]. Although they consider the instance-level representation, our VoxDet is essentially different from these works in the following aspects.

(1) Extra-label Free. Different from these works that require extra instance-level bounding box labels, VoxDet does not need any handcrafted bounding box labels for the object instances thanks to the proposed VoxNT trick, which significantly reduces the labeling labor with more practical usage.

(2) Extension to Image and LiDAR based Pipelines. Only using occupancy labels, VoxDet can be extended to both camera-based and LiDAR-based settings in a unified pipeline, achieving state-of-the-art results on both benchmarks. This differs from most existing works, which are tailored only for a single type of input modality. Hence, our method has more substantial flexibility and transferability.

(3) Flexible Definition of Object Instances. These works [46, 77] highly rely on the specific definition of instances in the detection datasets, which usually only considers some primary objects like cars and humans. However, extending to a more profound class set, such as buildings, is challenging and almost impossible, making this instance-level perception un-extensible. Differently, our VoxNT can handle the instances in arbitrary classes, providing an extremely flexible definition for object instances. For example, we can discover the length, width, and height of buildings, vegetation, etc (see Fig. 9), which is highly valuable for autonomous navigation.

(4) Technical Difference. Existing works [46, 77] introduce a separate 3D-object-detection branch to support occupancy prediction. In contrast, we consolidate both tasks into a single, detection-

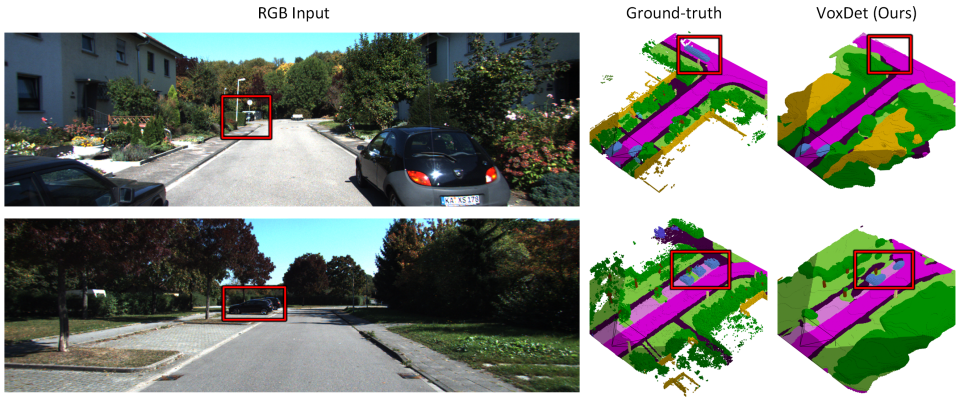


Figure 12: Failure cases of the proposed VoxDet. It is difficult for our method to correctly detect objects at extremely far distances satisfactorily due to limited semantic cues on the RGB input images.

driven formulation with an ultra-lightweight design, which can avoid the potential conflict between occupancy prediction and 3D object detection. Our feature decoupled designs also avoided the task-misalignment issue, which is ignored in existing works. The proposed dense-regression design, i.e., each voxel regresses the distance to the instance boundary, is a different paradigm for instance-level perception, which is implemented purely with convolutional layers, eliminating the complexity and computational overhead of object-query–based attention.

D.2 Broader Impacts

Impacts to the Broader Occupancy Community. Our VoxNT and VoxDet can be directly used in any voxel-included settings, such as multi-view occupancy prediction, as it is based on a similar voxel representation. By freely transferring semantic voxel labels into instance-level offset labels, this work may also inspire the occupancy community to reconsider the instance-level perception (e.g., instance and panoptic segmentation) achieved using only semantic labels, potentially contributing to training scale-up with practical label usage.

Impacts to the 3D Point Cloud Community. As voxelization is a key procedure in point-cloud understanding, our method, fully based on voxel representation, has great potential for the point cloud community. We achieve state-of-the-art results on the LiDAR-based SemanticKITTI benchmark, which justifies our potential impact on the point cloud community.

Impacts to the 3D Object Detection Community. Our VoxNT bridges the gap between dense occupancy and sparse 3D object detection. It converts semantically rich voxel-wise labels (e.g., car, building, vegetation) into instance-level offset labels that capture precise object extents and class-aware geometric priors. By integrating these instance labels into the occupancy, 3D detectors can revisit their image-driven origins, i.e., detecting objects directly from 2D inputs, while evolving toward a voxel-based instance detection paradigm that more faithfully reflects the physical 3D world.

D.3 Limitations and Future Work

Generate Sparse 3D Bounding Box Visualization. The current VoxDet uses instance-level supervision to guide the dense occupancy prediction, which can not directly generate spare 3D bounding boxes. The reason is that in dense detection [56, 40, 24], the Non-Maximum Suppression (NMS) is required to remove low-quality boxes for final visualizations. However, there is a gap between the 2D pixels and 3D voxels. To solve this, we will develop a 3D NMS algorithm tailored for voxels, bridging the gap between the voxel-level occupancy prediction and instance-level 3D object detection.

Lack the Usage of Extra Temporal Information. The current VoxDet is based on single-frame input. While state-of-the-art, the performance can be further improved with extra temporal information [25, 35], which can correct the cross-frame inconsistency. This will be our future work.

Implementation on Multi-View Pipelines. The current VoxDet is evaluated with a single camera input, which may limit its application scope. Our method can be transferred to the multi-view settings

Results

#	User	Entries	Date of Last Entry	mIoU ▲	completion ▲	Detailed Results
1	VITA-a	3	05/21/25	26.0 (9)	63.0 (1)	View
2	DPS2CNet	2	03/17/25	26.5 (7)	62.6 (2)	View
3	VITA	10	05/20/25	24.8 (19)	61.8 (3)	View
4	OccFiner_anonymous	3	03/06/24	37.8 (2)	61.7 (4)	View
5	JM	6	10/27/23	24.9 (16)	61.4 (5)	View
6	auto23	10	01/19/25	24.8 (17)	60.9 (6)	View
7	Lubo_Wang	4	03/01/24	25.6 (12)	60.7 (7)	View
8	sixwood	4	12/22/24	26.2 (8)	60.6 (8)	View
9	jdgalviss	8	08/03/23	27.1 (6)	60.6 (9)	View
10	Hailey	2	07/29/23	20.8 (29)	60.2 (10)	View

Figure 13: Online leaderboard of SemanticKITTI hidden test set.

effortlessly, as it is deployed on the 3D feature volume, which is also the representation for multi-view occupancy prediction. This will be our future work.

Rely on the Depth Prediction Accuracy. Although VoxDet achieves state-of-the-art performance, we empirically find that it will also suffer from some false-negative predictions of the objects at a long distance (see App. E). The sub-optimal depth estimation may be the reason. In the main paper, we also find that replacing the stereo depth with monocular depth leads to some performance drop, highlighting the reliance on depth accuracy. Therefore, improving the depth models in the future may further enhance the performance and serve as our future work.

D.4 Ethical Claims

Our VoxDet uses only publicly available datasets and produces 3D semantic volumes without retaining any personally identifiable information. We do not foresee any significant negative impacts: the method does not enable individual tracking or intrusive surveillance, poses no additional privacy or safety risks beyond standard camera perception systems, and does not rely on sensitive demographic attributes. By focusing on high-level scene understanding for benign applications such as autonomous navigation, our approach raises no known ethical, fairness, or regulatory concerns.

E Additional Qualitative Results

E.1 Failure Cases

In Fig. 12, we present some failure cases generated by the proposed VoxDet. We observe that our method fails to give correct detection for the object at extremely far distances (top sample), especially when objects of similar color are highly overlapped in the 2D images (bottom sample). This may be caused by the minimal visual cues on the 2D images, limiting the capacity. Some potential solutions may be (1) deploying more powerful visual feature extractors tailored for high-quality dense perception; (2) introducing extra modality, such as LiDAR, to enhance the information in the far distance; (3) developing tailored algorithms refining the features of objects in the far distance.

E.2 More Comparison with Other Methods

In Fig. 14 and 15, we provide additional qualitative comparisons against the state-of-the-art methods CGFormer [82] and OccFormer [85]. Our approach exhibits visually superior instance-level completeness, greater environmental consistency, and enhanced semantic perception.

E.3 Online Leaderboard

Fig. 13 illustrates the online leaderboard of SemanticKITTI hidden test set. Our method achieves the best 63.0 IoU (completion) only using the single-frame LiDAR input, showing the effectiveness.

Algorithm 1 PyTorch Style Pseudocode of Voxel-to-Instance (VoxNT) Trick.

```
def compute_all_direction_distances(gt_occ):
    B, X, Y, Z = gt_occ.shape

    dist_x_pos = run_length_along_dim(gt_occ, 1, "positive")
    dist_x_neg = run_length_along_dim(gt_occ, 1, "negative")
    dist_y_pos = run_length_along_dim(gt_occ, 2, "positive")
    dist_y_neg = run_length_along_dim(gt_occ, 2, "negative")
    dist_z_pos = run_length_along_dim(gt_occ, 3, "positive")
    dist_z_neg = run_length_along_dim(gt_occ, 3, "negative")

    # Stack into shape (B, 6, X, Y, Z)
    distances = torch.stack([
        dist_x_pos, dist_x_neg,
        dist_y_pos, dist_y_neg,
        dist_z_pos, dist_z_neg
    ], dim=1)
    return distances

def run_length_along_dim(t, dim, direction):
    if direction == "positive":
        return run_length_positive(t, dim)
    else:
        # flip, compute positive, then flip back
        tf = torch.flip(t, dims=(dim,))
        of = run_length_positive(tf, dim)
        return torch.flip(of, dims=(dim,))

def run_length_positive(t, dim):
    shape = t.shape
    L = shape[dim]
    out = torch.empty_like(t, dtype=torch.int32)

    # Initialize the last slice along dim to 1
    idx_last = [slice(None)] * len(shape)
    idx_last[dim] = - 1
    out[tuple(idx_last)] = torch.tensor(1, dtype=torch.int32, device=t.device)

    for i in range(L - 2, - 1, - 1):
        idx = [slice(None)] * len(shape); idx[dim] = i
        idx_next = [slice(None)] * len(shape); idx_next[dim] = i + 1

        current = t[tuple(idx)]
        nxt = t[tuple(idx_next)]

        # Check whether the next voxel is in the same class
        same = (current == nxt)
        out_next = out[tuple(idx_next)]

        out_val = torch.where(
            same,
            out_next + 1,
            torch.tensor(1, dtype=torch.int32, device=t.device)
        )
        out[tuple(idx)] = out_val

    return out
```

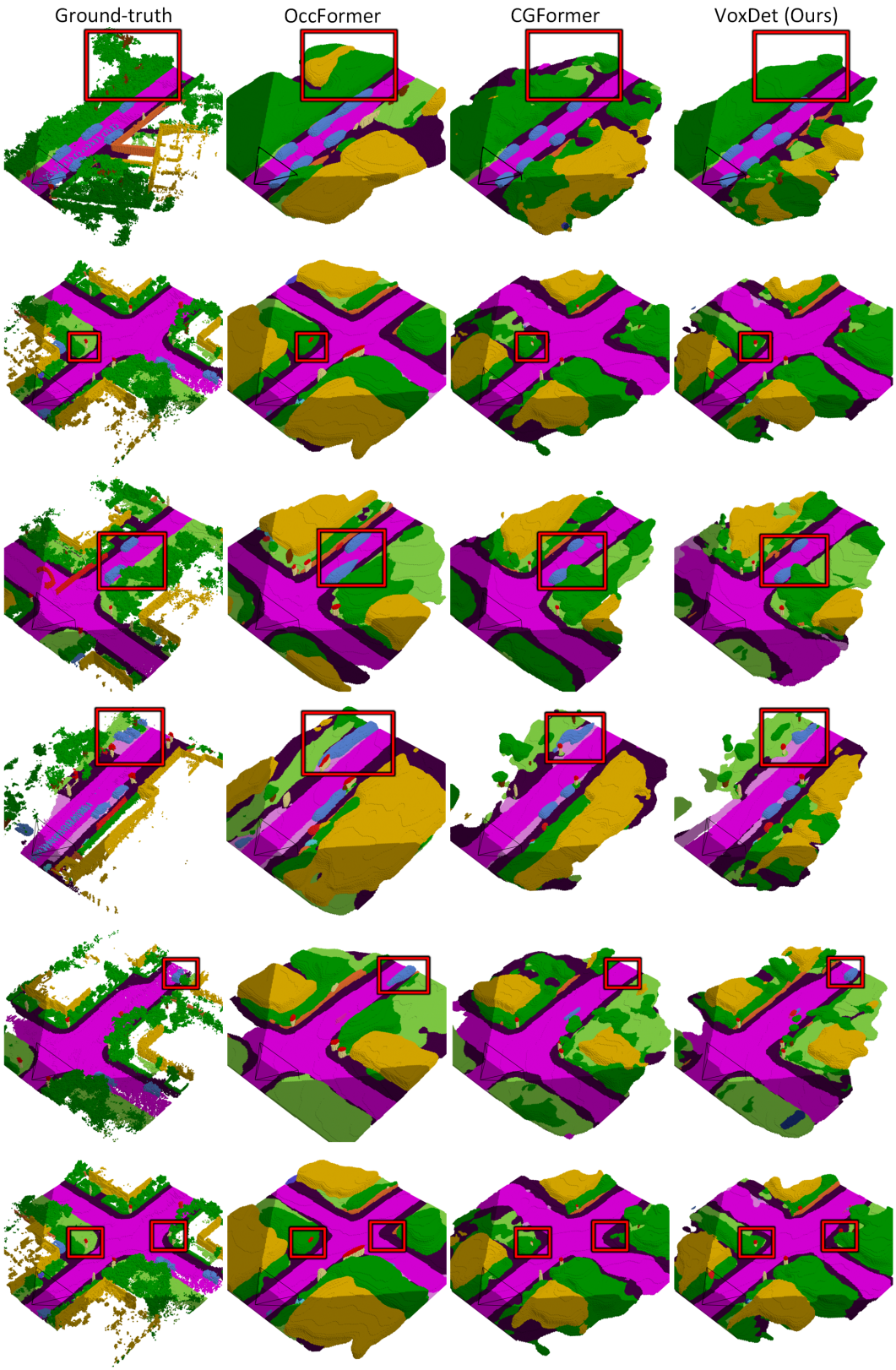


Figure 14: More qualitative comparisons on SemanticKITTI validation set.

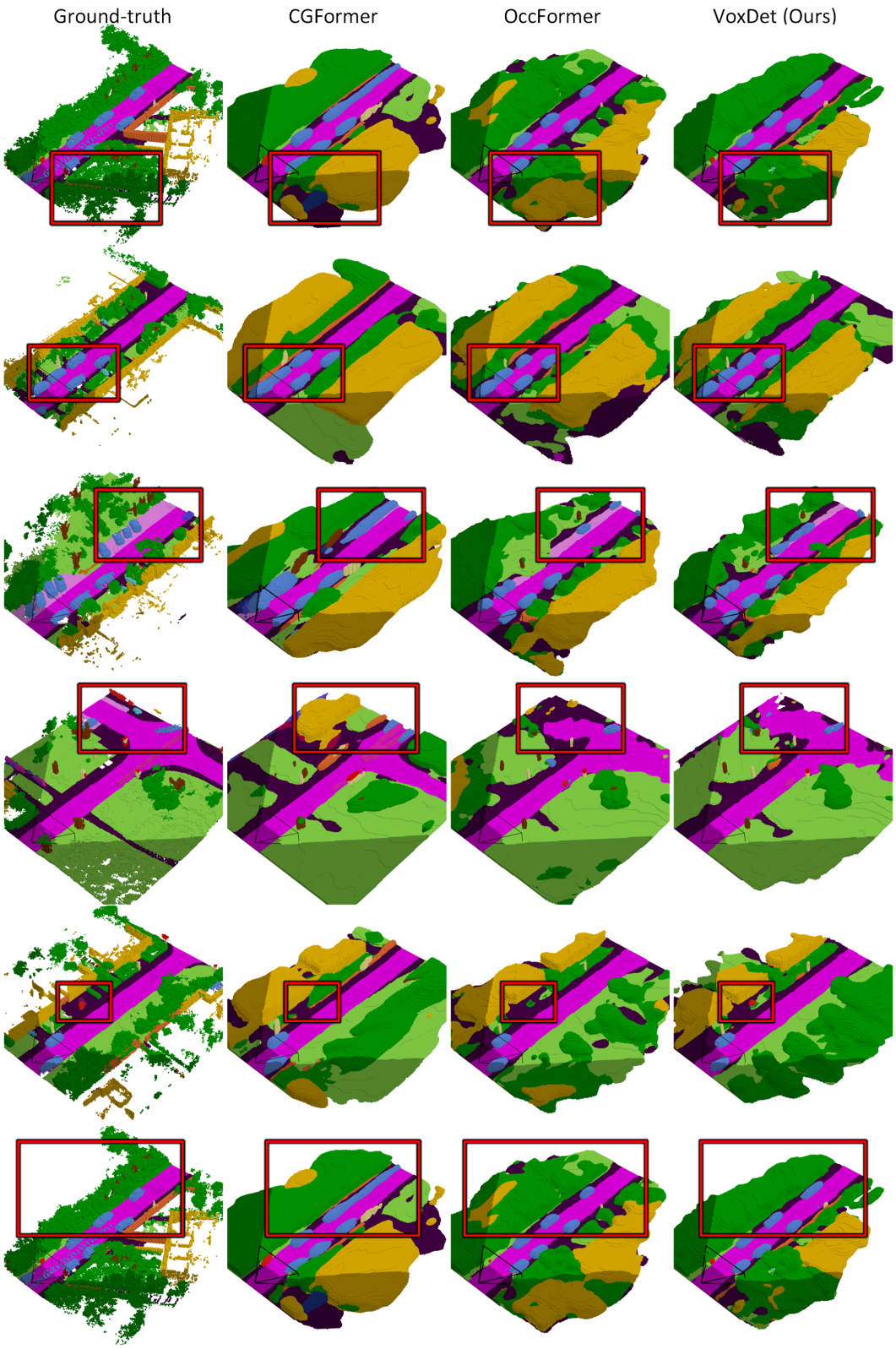


Figure 15: More qualitative comparisons on SemanticKITTI validation set.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[2] Jongseong Bae, Junwoo Ha, and Ha Young Kim. Three cars approaching within 100m! enhancing distant geometry by tri-axis voxel scanning for camera-based semantic scene completion. *arXiv preprint arXiv:2411.16129*, 2024.

[3] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9297–9307, 2019.

[4] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019.

[5] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4009–4018, 2021.

[6] Anh-Quan Cao and Raoul de Charette. Monoscene: Monocular 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3981–3991, 2022.

[7] Anh-Quan Cao, Angela Dai, and Raoul De Charette. Pasco: Urban 3d panoptic scene completion with uncertainty awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14554–14564, 2024.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, pages 213–229, 2020.

[9] LoÃ■ck Chambon, Eloi Zablocki, Alexandre Boulch, MickaÃl Chen, and Matthieu Cord. Gaussrender: Learning 3d occupancy with gaussian rendering. *arXiv preprint arXiv:2502.05040*, 2025.

[10] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3339–3348, 2018.

[11] Zehui Chen, Chenhongyi Yang, Qiaofei Li, Feng Zhao, Zheng-Jun Zha, and Feng Wu. Disentangle your dense object detector. In *Proceedings of the 29th ACM international conference on multimedia*, pages 4939–4948, 2021.

[12] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[13] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[14] Haoyi Duan, Hong-Xing Yu, Sirui Chen, Li Fei-Fei, and Jiajun Wu. Worldscore: A unified evaluation benchmark for world generation. *arXiv preprint arXiv:2504.00983*, 2025.

[15] Olivier Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.

[16] Martin Garbade, Yueh-Tung Chen, Johann Sawatzky, and Juergen Gall. Two stream 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[17] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[18] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[19] Xiyue Guo, Jiarui Hu, Junjie Hu, Hujun Bao, and Guofeng Zhang. Sgformer: Satellite-ground fusion for 3d semantic scene completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2025.

[20] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9223–9232, 2023.

[23] Haoyi Jiang, Tianheng Cheng, Naiyu Gao, Haoyang Zhang, Wenyu Liu, and Xinggang Wang. Symphonize 3d semantic scene completion with contextual instance queries. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[24] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, pages 734–750, 2018.

[25] Bohan Li, Jiajun Deng, Wenyao Zhang, Zhujin Liang, Dalong Du, Xin Jin, and Wenjun Zeng. Hierarchical temporal context learning for camera-based semantic scene completion. In *European Conference on Computer Vision*, 2024.

[26] Bohan Li, Yasheng Sun, Xin Jin, Wenjun Zeng, Zheng Zhu, Xiaoefeng Wang, Yunpeng Zhang, James Okae, Hang Xiao, and Dalong Du. Stereoscene: Bev-assisted stereo matching empowers 3d semantic scene completion. *arXiv preprint arXiv:2303.13959*, 2023.

[27] Chenxin Li, Hengyu Liu, Zhiwen Fan, Wuyang Li, Yifan Liu, Panwang Pan, and Yixuan Yuan. Instantsplamp: Fast and generalizable stenography framework for generative gaussian splatting. In *The Thirteenth International Conference on Learning Representations*, 2025.

[28] Chenxin Li, Xinyu Liu, Wuyang Li, Cheng Wang, Hengyu Liu, Yifan Liu, Zhen Chen, and Yixuan Yuan. U-kan makes strong backbone for medical image segmentation and generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 4652–4660, 2025.

[29] Heng Li, Yuenan Hou, Xiaohan Xing, Yuexin Ma, Xiao Sun, and Yanyong Zhang. Occmamba: Semantic occupancy prediction with state space models. *arXiv preprint arXiv:2408.09859*, 2024.

[30] Wuyang Li, Zhen Chen, Baopu Li, Dingwen Zhang, and Yixuan Yuan. Htd: Heterogeneous task decoupling for two-stage object detection. *IEEE Transactions on Image Processing*, 30:9456–9469, 2021.

[31] Wuyang Li, Xinyu Liu, Xiwen Yao, and Yixuan Yuan. Scan: Cross domain object detection with semantic conditioned adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[32] Wuyang Li, Xinyu Liu, and Yixuan Yuan. Sigma: Semantic-complete graph matching for domain adaptive object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5291–5300, 2022.

[33] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. *Advances in Neural Information Processing Systems*, 35:18442–18455, 2022.

[34] Yiming Li, Sihang Li, Xinhao Liu, Moonjun Gong, Kenan Li, Nuo Chen, Zijun Wang, Zhiheng Li, Tao Jiang, Fisher Yu, Yue Wang, Hang Zhao, Zhiding Yu, and Chen Feng. Sscbench: A large-scale 3d semantic scene completion benchmark for autonomous driving, 2024.

[35] Yiming Li, Zhiding Yu, Christopher B. Choy, Chaowei Xiao, José M. Álvarez, Sanja Fidler, Chen Feng, and Anima Anandkumar. Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9087–9098, 2023.

[36] Li Liang, Naveed Akhtar, Jordan Vice, Xiangrui Kong, and Ajmal Saeed Mian. Skip mamba diffusion for monocular 3d semantic scene completion. *arXiv preprint arXiv:2501.07260*, 2025.

[37] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 3292–3310, 2022.

[38] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022.

[39] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[40] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[41] Hengyu Liu, Yifan Liu, Chenxin Li, Wuyang Li, and Yixuan Yuan. Lgs: A light-weight 4d gaussian splatting for efficient surgical scene reconstruction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 660–670. Springer, 2024.
[42] Yifan Liu, Wuyang Li, Weihao Yu, Chenxin Li, Alexandre Alahi, Max Meng, and Yixuan Yuan. X-grm: Large gaussian reconstruction model for sparse-view x-rays to computed tomography. *arXiv preprint arXiv:2505.15235*, 2025.
[43] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
[44] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. Detrs beat yolos on real-time object detection. *arXiv preprint arXiv:2304.08069*, 2023.
[45] Jianbiao Mei, Yu Yang, Mengmeng Wang, Junyu Zhu, Xiangrui Zhao, Jongwon Ra, Laijian Li, and Yong Liu. Camera-based 3d semantic scene completion with sparse guidance network. *arXiv preprint arXiv:2312.05752*, 2023.
[46] Zhenxing Ming, Julie Stephany Berrio, Mao Shan, and Stewart Worrall. Inverse++: Vision-centric 3d semantic occupancy prediction assisted with 3d object detection. *arXiv preprint arXiv:2504.04732*, 2025.
[47] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Proceedings of the European Conference on Computer Vision*, pages 194–210, 2020.
[48] Han Qiu, Yuchen Ma, Zeming Li, Songtao Liu, and Jian Sun. Borderdet: Border feature for dense object detection. In *European Conference on Computer Vision*, pages 549–564. Springer, 2020.
[49] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.
[50] Christoph B Rist, David Emmerichs, Markus Enzweiler, and Dariu M Gavrila. Semantic scene completion using local deep implicit functions on lidar data. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7205–7218, 2021.
[51] Luis Roldão, Raoul de Charette, and Anne Verroust-Blondet. Lmscnet: Lightweight multiscale 3d semantic completion. In *Proceedings of the International Conference on 3D Vision*, pages 111–119, 2020.
[52] Luis Roldao, Raoul De Charette, and Anne Verroust-Blondet. 3d semantic scene completion: A survey. *International Journal of Computer Vision*, 130(8):1978–2005, 2022.
[53] Faranak Shamsafar, Samuel Woerz, Rafia Rahim, and Andreas Zell. Mobilestereonet: Towards lightweight deep networks for stereo matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2417–2426, 2022.
[54] Guanglu Song, Yu Liu, and Xiaogang Wang. Revisiting the sibling head in object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11563–11572, 2020.
[55] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas A. Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 190–198, 2017.
[56] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019.
[57] Lubo Wang, Di Lin, Kairui Yang, Ruonan Liu, Qing Guo, Wuyuan Xie, Miaohui Wang, Lingyu Liang, Yi Wang, and Ping Li. Voxel proposal network via multi-frame knowledge distillation for semantic scene completion. *Advances in Neural Information Processing Systems*, 37:101096–101115, 2024.
[58] Meng Wang, Huilong Pi, Ruihui Li, Yunchuan Qin, Zhuo Tang, and Kenli Li. Vlscene: Vision-language guidance distillation for camera-based 3d semantic scene completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 7808–7816, 2025.
[59] Ning Wang, Yang Gao, Hao Chen, Peng Wang, Zhi Tian, Chunhua Shen, and Yanning Zhang. Nas-fcos: Fast neural architecture search for object detection. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11943–11951, 2020.
[60] Ruoyu Wang, Yukai Ma, Yi Yao, Sheng Tao, Haoang Li, Zongzhi Zhu, Yong Liu, and Xingxing Zuo. L2cocc: Lightweight camera-centric semantic scene completion via distillation of lidar model. *arXiv preprint arXiv:2503.12369*, 2025.
[61] Song Wang, Jiawei Yu, Wentong Li, Wenyu Liu, Xiaolu Liu, Junbo Chen, and Jianke Zhu. Not all voxels are equal: Hardness-aware semantic scene completion with self-distillation. *arXiv*

preprint arXiv:2404.11958, 2024.

[62] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 913–922, 2021.

[63] Yue Wang, Vitor Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, , and Justin M. Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, pages 180–191, 2021.

[64] Yu Wang and Chao Tong. H2gformer: Horizontal-to-global voxel transformer for 3d semantic scene completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5722–5730, 2024.

[65] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21729–21740, 2023.

[66] Zhaoyang Xia, Youquan Liu, Xin Li, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, and Yu Qiao. Scpnet: Semantic scene completion on point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17642–17651, 2023.

[67] Haihong Xiao, Hongbin Xu, Wenxiong Kang, and Yuqiong Li. Instance-aware monocular 3d semantic scene completion. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[68] Dongli Xu, Jinhong Deng, and Wen Li. Revisiting ap loss for dense object detection: Adaptive ranking pair selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14187–14196, 2022.

[69] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3101–3109, 2021.

[70] Zhiqiang Yan, Jianhao Jiao, Zhengxue Wang, and Gim Hee Lee. Event-driven dynamic scene depth completion. *arXiv preprint arXiv:2505.13279*, 2025.

[71] Zhiqiang Yan, Xiang Li, Le Hui, Zhenyu Zhang, Jun Li, and Jian Yang. Rignet++: Semantic assisted repetitive image guided network for depth completion. *International Journal of Computer Vision*, pages 1–23, 2025.

[72] Zhiqiang Yan, Yuankai Lin, Kun Wang, Yupeng Zheng, Yufei Wang, Zhenyu Zhang, Jun Li, and Jian Yang. Tri-perspective view decomposition for geometry-aware depth completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4874–4884, 2024.

[73] Zhiqiang Yan, Kun Wang, Xiang Li, Zhenyu Zhang, Jun Li, and Jian Yang. Rignet: Repetitive image guided network for depth completion. In *European Conference on Computer Vision*, pages 214–230. Springer, 2022.

[74] Chenhongyi Yang, Mateusz Ochal, Amos Storkey, and Elliot J Crowley. Prediction-guided distillation for dense object detection. In *European conference on computer vision*, pages 123–138. Springer, 2022.

[75] Xuemeng Yang, Hao Zou, Xin Kong, Tianxin Huang, Yong Liu, Wanlong Li, Feng Wen, and Hongbo Zhang. Semantic segmentation-assisted scene completion for lidar point clouds. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3555–3562. IEEE, 2021.

[76] Yifeng Yang, Hengyu Liu, Chenxin Li, Yining Sun, Wuyang Li, Yifan Liu, Yiyang Lin, Yixuan Yuan, and Nanyang Ye. Concealgs: Concealing invisible copyright information in 3d gaussian splatting. *arXiv preprint arXiv:2501.03605*, 2025.

[77] Zhen Yang, Yanpeng Dong, Heng Wang, Lichao Ma, Zijian Cui, Qi Liu, and Haoran Pei. Daocc: 3d object detection assisted multi-sensor fusion for 3d occupancy prediction. *arXiv preprint arXiv:2409.19972*, 2024.

[78] Jiawei Yao and Jusheng Zhang. Depthssc: Depth-spatial alignment and dynamic voxel resolution for monocular 3d semantic scene completion. *arXiv preprint arXiv:2311.17084*, 2023.

[79] Zhu Yu, Bowen Pang, Lizhe Liu, Runmin Zhang, Qihao Peng, Maochun Luo, Sheng Yang, Mingxia Chen, Si-Yuan Cao, and Hui-Liang Shen. Language driven occupancy prediction. *arXiv preprint arXiv:2411.16072*, 2024.

[80] Zhu Yu, Zehua Sheng, Zili Zhou, Lun Luo, Si-Yuan Cao, Hong Gu, Huaqi Zhang, and Hui-Liang Shen. Aggregating feature point cloud for depth completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8732–8743, 2023.

[81] Zhu Yu, Zehua Sheng, Zili Zhou, Lun Luo, Si-Yuan Cao, Hong Gu, Huaqi Zhang, and Hui-Liang Shen. Aggregating feature point cloud for depth completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8732–8743, 2023.

[82] Zhu Yu, Runmin Zhang, Jiacheng Ying, Junchen Yu, Xiaohai Hu, Lun Luo, Si-Yuan Cao, and Hui-liang Shen. Context and geometry aware voxel transformer for semantic scene completion. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[83] Haoyang Zhang, Ying Wang, Feras Dayoub, and Niko Sunderhauf. Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8514–8523, 2021.

[84] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020.

[85] Yunpeng Zhang, Zheng Zhu, and Dalong Du. Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9433–9443, 2023.

[86] Yupeng Zheng, Xiang Li, Pengfei Li, Yuhang Zheng, Bu Jin, Chengliang Zhong, Xiaoxiao Long, Hao Zhao, and Qichao Zhang. Monoocc: Digging into monocular semantic occupancy prediction. *arXiv preprint arXiv:2403.08766*, 2024.

[87] Zhaohui Zheng, Rongguang Ye, Ping Wang, Dongwei Ren, Wangmeng Zuo, Qibin Hou, and Ming-Ming Cheng. Localization distillation for dense object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9407–9416, 2022.

[88] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 850–859, 2019.

[89] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.