Regulating Intermediate 3D Features for Vision-Centric Autonomous Driving

Junkai Xu[1,2*], **Liang Peng**[1,2,*], **Haoran Cheng**[1,2,*], **Linxuan Xia**[1,2,*],
Qi Zhou[1,2,*], **Dan Deng**[2], **Wei Qian**[2], **Wenxiao Wang**[3†], **Deng Cai**[1,2]

[1]State Key Lab of CAD & CG, Zhejiang University
[2]FABU Inc.
[3]School of Software Technology, Zhejiang University
{xujunkai, pengliang, haorancheng}@zju.edu.cn

Abstract

Multi-camera perception tasks have gained significant attention in the field of autonomous driving. However, existing frameworks based on Lift-Splat-Shoot (LSS) in the multi-camera setting cannot produce suitable dense 3D features due to the projection nature and uncontrollable densification process. To resolve this problem, we propose to regulate intermediate dense 3D features with the help of volume rendering. Specifically, we employ volume rendering to process the dense 3D features to obtain corresponding 2D features (*e.g.,* depth maps, semantic maps), which are supervised by associated labels in the training. This manner regulates the generation of dense 3D features on the feature level, providing appropriate dense and unified features for multiple perception tasks. Therefore, our approach is termed **Vampire**, stands for "**V**olume rendering **A**s **M**ulti-camera **P**erception **I**ntermediate feature **RE**gulator". Experimental results on the Occ3D and nuScenes datasets demonstrate that Vampire facilitates fine-grained and appropriate extraction of dense 3D features, and is competitive with existing SOTA methods across diverse downstream perception tasks like 3D occupancy prediction, LiDAR segmentation and 3D objection detection, while utilizing moderate GPU resources. We provide a video demonstration in the supplementary materials and Codes are available at github.com/cskkxjk/Vampire.

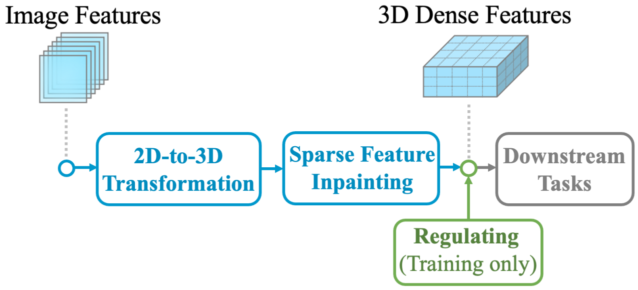


Figure 1: Method overview. The key idea is to regulate dense intermediate 3D features in training, to produce appropriate features for different downstream perception tasks.

Introduction

Vision-centric 3D surrounding perception plays an important role in modern autonomous driving and robotics due to its convenience and board applicability for downstream tasks. Vision-based perception frameworks can be broadly categorized into two paradigms (Li et al. 2023): backward projection (or Transformer-based (Li et al. 2022a)) and forward projection (or LSS-based, as it originates from the concept of "Lift, Splat, Shoot" (Philion and Fidler 2020)).

Backward projection / Transformer-based approaches set 3D points in 3D space or BEV plane and then projects these points back onto the 2D image. This procedure allows each predefined 3D or BEV position to obtain corresponding image features. Transformer (Vaswani et al. 2017) architectures are widely used in this paradigm to aggregate information from image features, generating task-specific features tailored to meet the objectives (Li et al. 2022c; Huang et al. 2023; Wei et al. 2023; Zhang, Zhu, and Du 2023; Zhou and Krähenbühl 2022; Liu et al. 2022a,b). When demonstrating promising performance on various perception tasks such as 3D object detection (Li et al. 2022c; Liu et al. 2022a), BEV map segmentation (Li et al. 2022c; Liu et al. 2022b) and 3D occupancy prediction (Huang et al. 2023; Wei et al. 2023; Zhang, Zhu, and Du 2023), they require substantial GPU memory to support the interaction between task queries and image features (Zhang et al. 2022).

In contrast, forward projection / LSS-based methods project 2D image features onto the 3D space, incorporating per-pixel depth estimation. They rely on implicit (Philion and Fidler 2020; Hu et al. 2021) or explicit (Li et al. 2022b,a; Huang et al. 2021) depth estimation to elevate image features to the 3D space, acquiring intermediate feature representations such as BEV or 3D voxel representation for task-specific heads. This paradigm is effective for object-level perception tasks, *e.g.*, 3D object detection, but struggles with dense point / grid-level perception tasks, *e.g.*, 3D occupancy prediction. Using estimated per-pixel depth and camera calibrations, these methods position 2D features at the foremost visible surface of objects in the 3D space, leading to sparse 3D features.

An intuitive resolution would be to densify the sparse 3D features using a feature inpainting module. This enables the model to guess and inpaint the empty regions based on

[*]Work performed during an internship at FABU Inc.

[†]Corresponding author

known sparse features and produce dense 3D features. However, this naive manner is uncontrollable due to the lack of regulations, and could cause some kind of "overgeneration", which means that features may be generated at the wrong places and violate the geometry constraints. Here the question arises: ***how to find decent regulations in favor of appropriate dense 3D feature extraction?***

We resort to employing occupancy to model the 3D feature space in a unified manner. Occupancy is an ideal dense 3D representation due to its fine-grained information and universality in different tasks (Huang et al. 2023; Tian et al. 2023; Sima et al. 2023). We observe that there is an analogy between the occupancy and the volume density in the implicit scene representation NeRF (Mildenhall et al. 2021; Barron et al. 2021), as they both describe whether a space is occupied. This observation motivates us to employ additional information from the 2D space to implicitly regulate our intermediate 3D features, as NeRF does.

To this end, we incorporate volume rendering (Max 1995) as a regulator for the intermediate 3D feature space (see Figure 1). Specifically, we map image features to 3D voxel space following the LSS scheme (Philion and Fidler 2020), and employ a 3D hourglass-like design (Chang and Chen 2018) as the sparse feature inpaintor. The resulting dense intermediate 3D features are then used to generate feature volumes (density, semantic) for volume rendering. We supervise the rendered depth maps and semantic maps with LiDAR projected ground-truth labels under both camera views and bird's-eye-view. In this way, we employ simple 2D supervisions to regulate dense intermediate 3D features, which ensures our sparse feature inpaintor not to generate unreasonable 3D features that could violate their 2D correspondences. We term the entire framework as **Vampire**, stands for taking volume rendering as a regulator for intermediate features in multi-camera perception. We provide the overview design in Figure 1.

We perform experiments on various multi-camera perception tasks, including 3D occupancy prediction (Tian et al. 2023), image-based LiDAR segmentation on the competitive nuScenes dataset (Caesar et al. 2020), and we also assess whether the regulated 3D features continue to exhibit effectiveness for the 3D object detection.

The contributions of this work are summarized as follows:

- We provide a new outlook on intermediate features for vision-centric perception tasks, drawing connections between the occupancy in autonomous driving and volume density in NeRF.
- We introduce Vampire, a multi-camera perception framework. The key component lies in using volume rendering as a regulator for dense intermediate 3D features. As such, different perception tasks benefit from the regulated intermediate features.
- We demonstrate that our method can handle several perception tasks in a single forward pass with moderate computational resources. The single Vampire model that consumes limited GPU memory (12GB per device) for training is comparable with other existing SOTAs across multiple perception tasks (3D occupancy prediction, image-based LiDAR segmentation and 3D objection detection).

Related Work

Multi-camera 3D Perception

3D object detection is a classic and longstanding 3D perception task. In multi-camera setting, various attempts (Philion and Fidler 2020; Li et al. 2022c; Huang et al. 2021; Li et al. 2022b) have been proposed for detecting objects in the bird's-eye-view (BEV) representations which collapse the height dimension of 3D space to achieve a balance between accuracy and efficiency. LSS (Philion and Fidler 2020) and its follow-ups (Huang et al. 2021; Li et al. 2022b,a) first estimate implicit or explicit per-pixel depth distributions to back-project the 2D image features into 3D space, then use the pooling operation or height compression to generate BEV features. Others take advantage of Transformer (Vaswani et al. 2017) and use learnable object-level queries to directly predict 3D bounding boxes (Wang et al. 2022; Liu et al. 2022a,b) or position-aware queries to produce BEV features (Li et al. 2022c; Zhou and Krähenbühl 2022). However, there are innumerable rigid and nonrigid objects with various structures and shapes in the real-world autonomous driving, which cannot be handled by classic 3D object detection. An alternative is to assign occupancy states to every spatial region within the perceptive range(Tesla 2022), namely, 3D occupancy prediction. Unlike LiDAR segmentation (Fong et al. 2022) which is designed for sparse scanned LiDAR points, the occupancy prediction task aims to achieve dense 3D surrounding perception. This area haven't been thoroughly explored yet, only a few works use transformer-based designs to deal with it. TPVFormer (Huang et al. 2023) proposes to use tri-perspective view (TPV) grid queries to interact with image features and get reasonable occupancy prediction results to describe the 3D scene. SuroundOcc (Wei et al. 2023) builds 3D volume queries to reserve 3D space information. CONet (Wang et al. 2023) and SuroundOcc (Wei et al. 2023) both generate dense occupancy labels for better prediction performance. OccFormer (Zhang, Zhu, and Du 2023) use a dual-path transformer network to get fine-grained 3D volume features. Occ3D (Tian et al. 2023) and OccNet (Sima et al. 2023) label the original nuScenes dataset to get occupancy data at different scopes. In this paper, we advocate to regulate the dense 3D features to achieve better perception.

Scene Representation Learning

Effective 3D scene representation is the core of autonomous driving perception. Voxel-based scene representations turn the 3D space into discretized voxels which is usually adopted by LiDAR segmentation (Ye et al. 2022, 2021; Zhu et al. 2021), 3D scene completion (Cao and de Charette 2022; Chen et al. 2020; Roldao, de Charette, and Verroust-Blondet 2020) and 3D occupancy prediction (Wang et al. 2023). BEV-based scene representations collapse 3D features onto the Bird's Eye View (BEV) plane and achieve a good balance between accuracy and efficiency. They show its effectiveness in 3D object detection (Li et al. 2022c; Huang et al. 2021; Li et al. 2022b,a; Zhang et al. 2022) and BEV segmentation (Philion and Fidler 2020; Li et al. 2022c; Hu et al. 2021; Xie et al. 2022) but are not applicable

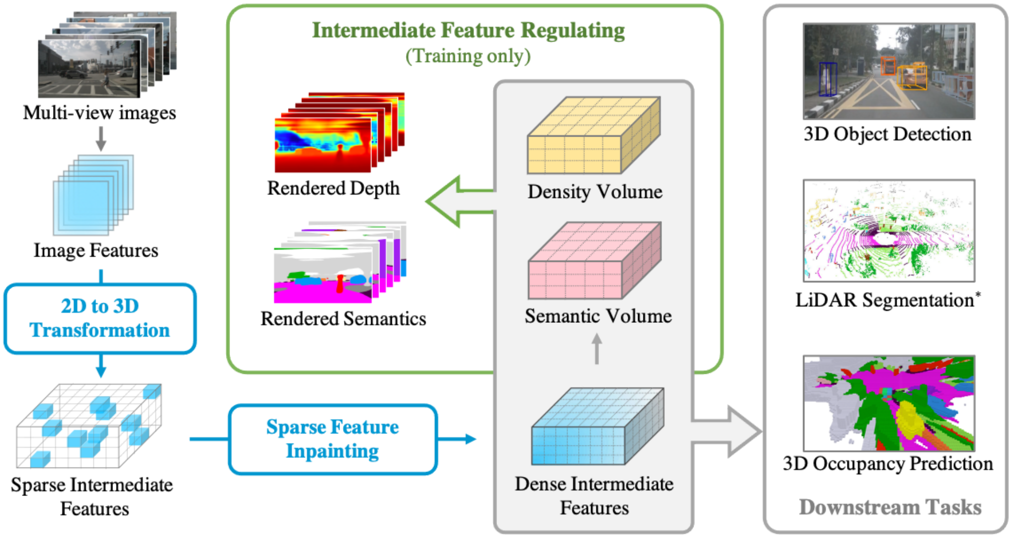


Figure 2: Framework of Vampire. We extract the 2D image features from multi-view images, and transform the 2D features to 3D volume space to generate sparse intermediate 3D features. Inpainting techniques are then applied to obtain dense intermediate features. Density volume and semantic volume are generated by forwarding the dense features with specific heads. Specifically, in the training stage, we regulate intermediate features by constructing loss between ground truth and volume rendered images. The dense features and volumes can be used for various downstream tasks such as 3D occupancy prediction, image-based LiDAR segmentation, and 3D object detection. Please note that for the image-based LiDAR segmentation task, we do not use point clouds as input but employ its evaluation protocol (Huang et al. 2023; Wei et al. 2023; Sima et al. 2023). The point clouds serve as point queries to extract features for training supervision and evaluation.

for dense perception tasks when losing the height dimension. Notably, recent implicit scene representation methods demonstrate their potential to represent meaningful 3D scenes. They learn continuous functions to consume 3D coordinates and output representation of a certain point. This kind of representation can model scenes at arbitrary-resolution and is commonly used for 3D reconstruction (Chabra et al. 2020; Park et al. 2019) and novel view synthesis (Mildenhall et al. 2021; Barron et al. 2021; Yariv et al. 2021; Wang et al. 2021).

As far as we know, very limited researches have explored on combining implicit scene representations with existing representations in autonomous driving. Tesla (Tesla 2022) firstly discusses the similarity between scene occupancy and NeRF (Mildenhall et al. 2021) but it concentrates on using volume rendering to train the occupancy representation for scene reconstruction. A recent related work is (Gan et al. 2023), which adopts similar idea to consider the occupancy the same as volume density and use volume rendering for better depth estimation. Their models use parameter-free back projection to map 2D image features to 3D volume and aggregates the 3D features with corresponding position embeddings to predict volume density and render the depth maps. Most related to ours is the model of (Pan et al. 2023), which shares the same spirit and adopts rendering-based supervision as us. This model is very similar to Vampire, but our work goes further to demonstrate that such volume-rendering-assisted perception framework benefits multiple perception tasks and achieves competitive results with state-of-the-art approaches.

Methodology

Overview

The overall framework is illustrated in Figure 2. Vampire consists of three stages: **2D-to-3D Transformation**, **Sparse Feature Inpainting** and **Intermediate Feature Regulating**. The surrounding multi-camera images are first passed to 2D image backbone to extract 2D image features. In 2D-to-3D transformation stage, the 2D image features are transformed from 2D image space to the 3D volume space. We follow the LSS scheme (Philion and Fidler 2020; Li et al. 2022b,a; Huang et al. 2021) to perform the feature mapping along depth dimension. To overcome the 3D feature sparsity of LSS transformation, we take a 3D hourglass net (Chang and Chen 2018) as feature inpaintor to conduct the sparse feature inpainting and generate dense intermediate 3D features. The final volume features (semantic volume and density volume) can be obtained by forwarding the dense intermediate 3D features with specific heads. In intermediate feature regulating stage, we sample points along the ray from camera views or BEV view and get corresponding features for rendering, the rendered images and feature maps are used to construct losses to regulate the intermediate features.

2D-to-3D Transformation

We adopt LSS paradigm (Philion and Fidler 2020) to transform 2D image features to 3D features. LSS-based transformations do not generate redundant features like parameter-free transformations (Cheng, Wang, and Fragkiadaki 2018; Sitzmann et al. 2019; Harley et al. 2019, 2022) and are more effective than transformer-based transformations (Li et al. 2022c; Huang et al. 2023; Wei et al. 2023). We use two simple 1-layer 2D convolution neural network (CNN) to conduct this process. The first one is used to predict categorical depth distribution with softmax activation, and the second one is used to lower the dimension of image features to meet our device constraints. These two CNNs work together to map image features along depth axis, and we do not explicitly supervise this mapping process with depth labels. In this way, 2D image features are placed at the front visible surface for any certain pixels.

Sparse Feature Inpainting

The aforementioned 2D-to-3D transformation produce sparse intermediate 3D features, and such sparsity is not appropriate for dense prediction tasks like occupancy prediction. To overcome this limitation, we draw inspiration from classic image inpainting (Liu et al. 2018; He et al. 2022) and use a 3D hourglass-like design (Chang and Chen 2018) to inpaint the sparse intermediate 3D features V_{sparse} and generate dense intermediate 3D features V_{dense}. Please refer to supplementary materials for network architecture details.

Intermediate Feature Regulating

In this stage, we use the dense intermediate 3D features V_{dense} to produce two volumetrically 3D features – density volume $V_{density}$, semantic volume $V_{semantic}$. Different from (Mildenhall et al. 2021), we adopt the SDF (Signed Distance Function) to model the volume density σ to facilitate the trilinear interpolation during grid sampling. Specifically, we predict the signed distance volume V_{sdf} where each value in a position in this volume represents its distance to its nearest surface. Then we transform the signed distance volume V_{sdf} to density volume $V_{density}$ by applying transformation function. We use the same transformation function as (Yariv et al. 2021):

$$V_{density} = \alpha \Psi_\beta \left(V_{sdf} \right), \qquad \Psi_\beta(s) = \begin{cases} \frac{1}{2} \exp(\frac{s}{\beta}) & \text{if } s \leq 0 \\ 1 - \frac{1}{2} \exp(-\frac{s}{\beta}) & \text{if } s > 0 \end{cases} \tag{1}$$

where $\alpha, \beta > 0$ are learnable parameters and Ψ_β is the cumulative distribution function of the Laplace distribution with zero mean and β scale, s is the predicted signed distance at coordinate $\boldsymbol{x}$. For a coordinate $\boldsymbol{x}$ in range of interest, we can get its feature embeddings including volume density $\boldsymbol{\sigma}\left(\boldsymbol{x}\right)$ and semantic logits $\boldsymbol{s}\left(\boldsymbol{x}\right)$ by grid sampling $\mathcal{G}\left(\cdot, \boldsymbol{x}\right)$ in these 3D volume features.

$$\boldsymbol{\sigma}\left(\boldsymbol{x}\right) = \mathcal{G}\left(V_{density}, \boldsymbol{x}\right), \quad \boldsymbol{s}\left(\boldsymbol{x}\right) = \mathcal{G}\left(V_{semantic}, \boldsymbol{x}\right) \tag{2}$$

To compute the depth and semantics of a single pixel, we adopt similar techniques as (Zhi et al. 2021; Kerr et al. 2023) to accumulate feature embeddings along a ray $\vec{r} = \vec{o_t} + t\vec{d}$. The rendering weights are calculated by:

$$w(t) = \int_t T(t)\boldsymbol{\sigma}(t)dt, \qquad \text{where } T(t) = \exp\left(\int_t (-\boldsymbol{\sigma}(c))dc\right) \tag{3}$$

So the rendered feature embeddings are:

$$D(r) = \int_t w(t)r(t)dt, \quad S(r) = \int_t w(t)\boldsymbol{s}(r(t))dt \tag{4}$$

In Vampire, we conduct volume rendering in both camera view and bird's eye view.

Camera View. For camera view, we render depth and semantic maps to achieve the supervision from 2D space. To render a pixel, we cast a ray from the camera center through the pixel. We sample n depth value $\{z_i | i = 1, ..., n\}$ for a pixel $[u, v]^T$ and use known camera calibration to back-project the pixel to several 3D points $\boldsymbol{x} \in \{[x_i, y_i, z_i]^T | i = 1, ..., n\}$. The corresponding volume densities and semantic logits are obtained by Equation 2, and the depth and semantic maps can be calculated by Equation 4.

Bird's Eye View. Different from rendering in camera view, we do not need camera calibration under bird's-eye-view. Instead, we render directly from the top-down height axis to obtain the BEV height maps and BEV semantic maps. See Figure 3 for reference.

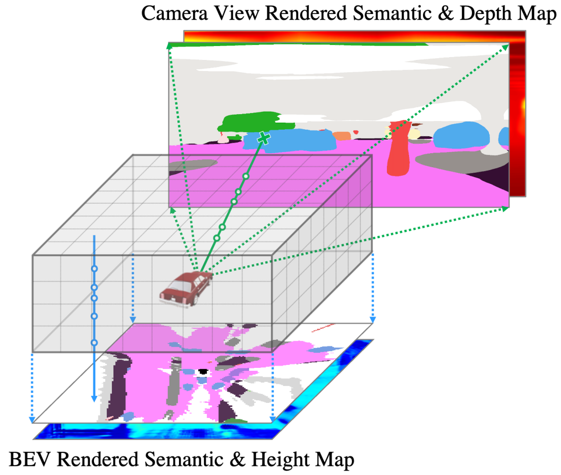

Camera View Rendered Semantic & Depth Map
BEV Rendered Semantic & Height Map

Figure 3: Rendering operations in Intermediate Feature regulating stage. For the camera view, the semantic and depth map are rendered by casting rays from the camera center through each pixel. Several 3D points are sampled along the ray to calculate density and semantic values. For the bird's-eye-view, the semantic and height map are rendered directly from the top-down height axis.

Optimization

Depth Consistency Loss. We enforce consistency between the rendered depth (or height) D and the ground-truth camera depth (or BEV height) $\bar{D}$.

$$\mathcal{L}_{dep} = \frac{1}{N^c_{valid}} \sum_{i=1}^{N^c_{valid}} \text{Smooth}_{\text{L}_1}(D^c_i - \bar{D}^c_i) + \frac{1}{N^b_{valid}} \sum_{i=1}^{N^b_{valid}} \text{Smooth}_{\text{L}_1}(D^b_i - \bar{D}^b_i) \tag{5}$$

Semantic Consistency Loss. Similarly, we impose consistency on the volume-rendered semantic logits S^c and the ground-truth semantic label $\bar{S}^c$, we employ both cross entropy (CE) loss and lovasz-softmax (LS) loss:

$$\mathcal{L}_{sem} = \frac{1}{N^c_{valid}} \sum_{i=1}^{N^c_{valid}} (\text{CE}(S^c_i, \bar{S}^c_i) + \text{LS}(S^c_i, \bar{S}^c_i)) + \frac{1}{N^b_{valid}} \sum_{i=1}^{N^b_{valid}} (\text{CE}(S^b_i, \bar{S}^b_i) + \text{LS}(S^b_i, \bar{S}^b_i)) \tag{6}$$

where N^c_{valid} is the number of pixels with ground-truth depth for all cameras (obtained by projecting the sparse LiDAR points to current camera image plane). N^b_{valid} is the number of pixels with ground-truth height in the range of BEV. $\bar{D}^c$ and $\bar{S}^c$ are the ground-truth depths and semantic labels obtained by projecting the sparse LiDAR points to the current camera image plane. For BEV, we get the ground-truth label $\bar{D}^b$ and $\bar{S}^b$ by projecting the LiDAR points to the grounding plane and take the height and semantic label of the highest point for a pixel in BEV map.

The overall loss we use to regulate our intermediate 3D features is:

$$\mathcal{L}_{reg} = \lambda_{dep}\mathcal{L}_{dep} + \lambda_{sem}\mathcal{L}_{sem} \tag{7}$$

where $\lambda_{dep}, \lambda_{sem}$ are fixed loss weights. We empirically set all weights to 1 by default.

Applications of Vampire Features

We follow the existing scheme (Huang et al. 2023; Li et al. 2022b) to use our regulated intermediate features.

3D occupancy prediction. The 3D occupancy prediction task usually covers a certain range of scene, thus we voxelize the interested scene range and conduct grid sampling from our predicted semantic volume $V_{semantic}$ with these voxel center coordinates. We use the output semantic logits to represent the semantic occupancy for each voxel.

LiDAR segmentation. Different from traditional LiDAR segmentation task, our model consumes purely RGB images to perceive 3D surroundings rather than LiDAR point cloud. To conduct LiDAR segmentation, we use LiDAR point clouds as point queries to get the corresponding semantic logits from semantic volume $V_{semantic}$.

3D object detection. We adopt a tanh function to scale the density volume $V_{density}$ to the range of [0, 1] and then use it to enhance dense 3D intermediate features V_{dense}. We collapse the height dimension and use a linear layer to squeeze the feature dimension:

$$F_{BEV} = \mathcal{HC}(V_{dense} \cdot \tanh(V_{density})) \tag{8}$$

Where $\mathcal{HC}$ stands for "height compression", which collapses the height dimension of 3D features then squeezes the feature dimension with a linear layer to get the final BEV shape features F_{BEV} for detection. F_{BEV} is then fed into the detection head to obtain final detection results. For simplicity, we adopt the detection head of BEVDepth (Li et al. 2022b) to produce 3D object detection results.

Experiments

To evaluate the proposed method, we benchmark Vampire on challenging public autonomous driving datasets nuScenes (Caesar et al. 2020) and its variants (Tian et al. 2023; Fong et al. 2022).

Datasets. The nuScenes dataset contains 1000 scenes of 20 seconds duration each, and the key samples are annotated at 2Hz. Each sample consists of RGB images from 6 surrounding cameras with 360° horizontal FOV and point cloud data from 32 beams LiDAR. The total of 1000 scenes are officially divided into training, validation and test splits with 700, 150 and 150 scenes, respectively. Occ3D-nuScenes (Tian et al. 2023) contains 700 traing scenes and 150 validation scenes. The occupancy scope is defined as $[-1.0, 5.4] \times [-40.0, 40.0] \times [-40.0, 40.0](meter)$ with a voxel size of 0.4-meter.

Implementation details. Our implementation is based on official repository of BEVDepth (Li et al. 2022b). We use ResNet-50 (He et al. 2016) as image backbone and the image resolution of 256 × 704 to meet our computational resources. For the inpainting network, we adpot an hourglass-like architecture (further details are provided in our supplementary materials). The intermediate 3D feature resolutions are 20 × 256 × 256 corresponding to the range of $[-3.0, 5.0] \times [-51.2, 51.2] \times [-51.2, 51.2](meter)$ and the 3D feature dimension are set to 16 by default. We use AdamW as an optimizer with a learning rate set to 2e-4 and weight decay as 1e-7. All models are trained for 24 epochs with a total batch size of 8 on 8 3080Ti GPUs (12GB).

Method	Backbone	Image Size	mIoU ↑
MonoScene (2023)	Effi.NetB7	900×1600	6.1
BEVDet (2023)	R101	900×1600	19.4
OccFormer (2023)			21.9
BEVFormer (2023)			26.9
TPVFormer (2023)			27.8
CTF-Occ (2023)			**28.5**
UniOcc (2023)	R50	256×704	22.0
Vampire (ours)			28.3

Table 1: 3D occupancy prediction results on Occ3D-nuScenes. "Effi.NetB7" stands for EfficientNetB7. We obtain the values of other methods from the benchmark paper (Tian et al. 2023). We use bold to indicate the highest result and underline for the second-best result. Despite image backbone and input size differences, Vampire achieves comparable performance with state-of-the-art methods.

Method	Backbone	Image Size	mIoU ↑
BEVFormer (2023)	R101	900×1600	56.2
TPVFormer (2023)			**68.9**
TPVFormer† (2023)			58.5
OccNet† (2023)			60.5
TPVFormer (2023)	R50	450×800	59.3
OccNet† (2023)		900×1600	53.0
Vampire (ours)		256×704	<u>66.4</u>
Vampire (ours) †		256×704	62.2

Table 2: LiDAR segmentation results on Panoptic nuScenes (Fong et al. 2022) validation set. We obtain the values of baselines from their respective papers. Mark † indicates methods trained without direct LiDAR supervision but only occupancy semantic labels.

3D Occupancy Prediction

We compare Vampire with previous state-of-the-art methods on the 3D occupancy prediction task in Table 1. These baseline methods including two main-stream BEV models – BEVDet (Huang et al. 2021), BEVFormer (Li et al. 2022c) and five existing 3D occupancy prediction methods – MonoScene (Cao and de Charette 2022), TPVFormer (Huang et al. 2023), OccFormer (Zhang, Zhu, and Du 2023), UniOcc (Pan et al. 2023), and CTF-Occ (Tian et al. 2023). It can be observed that our method achieves comparable performance with these methods under the mIoU metric. Our Vampire surpasses OccFormer / BEVFormer / TPVFormer by 6.4 / 1.4 / 0.5 mIoU. Although Vampire has a lower mIoU than CTF-Occ (28.3 v.s. 28.5), it is still promising since our method adopts a relatively weak image backbone ResNet-50 and lower input image resolution (256 × 704).

LiDAR Segmentation

We compare Vampire with existing image-based LiDAR segmentation methods in Table 2. These baseline methods including BEVFormer (Li et al. 2022c), TPVFormer (Huang et al. 2023) and OccNet (Sima et al. 2023). In the inference stage, we predict the semantic labels for given points in the LiDAR segmentation task. Vampire surpasses the SOTA model TPVFormer (Huang et al. 2023) with the same backbone in terms of mIoU (66.4 v.s. 59.3), but a little lower (-2.5) compared to TPVFormer-R101. Even without direct 3D LiDAR supervision, Vampire can outperform OccNet (Sima et al. 2023) models with different backbones respectively by 9.2 and 1.7 points in mIoU.

3D Object Detection

We conduct 3D object detection experiments on nuScenes validation set. The intention is to verify whether the regulated 3D features can still qualified for 3D detection task. We choose several main-stream 3D object detection baselines including BEVFormer (Li et al. 2022c), BEVDet (Huang et al. 2021) and BEVDepth (Li et al. 2022b). For fair comparisons, we report the baseline values under the setting of ResNet-50 backbone and without temporal fusion techniques. We also choose three baselines provided by (Sima et al. 2023) which conducts joint training of occupancy prediction and 3D detection task like us. As shown in Table 3, comparing to normal 3D object detection methods, Vampire surpasses BEVFormer and BEVDet in mAP (0.301 v.s. 0.286), but lower in NDS (0.354 v.s. 0.372). This could be attributed to negative transfer (Pan and Yang 2009) in joint training of multi-task. BEVDepth reports the value with EMA technique and a large batch size of 64, thus we attribute the performance gap to that. For joint training baselines, Vampire achieves a significantly higher mAP (0.301 v.s. 0.277), but has a gap on the metric of NDS (0.354 v.s. 0.390) and the metric of mean Average Velocity Error (0.541 v.s. 1.043). To summarize, Vampire can perceive the geometry details of 3D surroundings but less sensitive with object velocity. It is because the baseline methods are trained by occupancy data with additional flow annotation (occupancy velocity), which can significantly improve their performance to perceive object speed.

Ablation Studies

Architectural components. We conduct an ablation study on network structures and the proposed losses under the multi-task setting in Table 4. For 3D occupancy prediction task and LiDAR segmentation task, we report the mIoU. For 3D object detection task, we report the NDS. As a parameter-free method, Bilinear (Harley et al. 2022) can produce dense 3D features in the simplest way but also cause massive features generated at the wrong 3D spaces, resulting poor performances in all three tasks. The LSS baseline produces sparse 3D intermediate features, which can handle object detection, but fails to handle dense prediction tasks (*e.g.*, occupancy prediction). When employing the feature inpaintor, dense point / grid level tasks (*i.e.*, occupancy and segmentation) obtain significant improvements. The regulation of depth $\mathcal{L}_{dep}$ improves the occupancy prediction, but

Method	Joint.	mAP ↑	NDS ↑	mAVE ↓
BEVFormer (2022c)		0.257	0.359	0.660
BEVDet (2022b)		0.286	0.372	-
BEVDetph (2022b)		0.322	0.367	-
BEVNet (2023)	✓	0.271	**0.390**	**0.541**
VoxNet (2023)		0.277	0.387	0.614
OccNet (2023)		0.276	**0.390**	0.570
Vampire (ours)		**0.301**	0.354	1.043

Table 3: 3D object detection results on nuScenes validation set. "mAVE" stands for mean Average Velocity Error. Vampire achieves comparable mAP with baseline methods but fails to sense accurate velocity. The joint-training baselines are trained with additional occupancy flow annotation (occupancy velocity) (Sima et al. 2023), which can significantly improve their performance to perceive object speed.

Trans.	Inp.	$\mathcal{L}_{dep}$	$\mathcal{L}_{sem}$	Occ.↑	Seg.↑	Det.↑
Bilinear				21.3	56.7	0.301
LSS				21.9	56.5	**0.318**
LSS	✓			23.8	60.1	0.316
LSS	✓	✓		24.9	59.6	0.309
LSS	✓	✓	✓	**25.8**	**62.6**	**0.318**

Table 4: Ablation study for network structures and losses. “Trans.” stands for 2D-to-3D transformation. “Inp.” stands for sparse feature inpainting. “Occ.” represents 3D occupancy prediction, “Seg.” refers to LiDAR segmentation, “Det.” denotes 3D object detection.

Camera.	BEV.	Occ.↑	Seg.↑	Det.↑
✓		24.6	61.9	0.303
	✓	24.9	60.0	0.315
✓	✓	**25.8**	**62.6**	**0.318**

Table 5: Ablation study for camera and BEV views. “Camera.” stands for volume rendering loss in camera view. “BEV.” stands for volume rendering loss in BEV view. “Occ.” represents 3D occupancy prediction, “Seg.” refers to LiDAR segmentation, “Det.” denotes 3D object detection.

has a negative effect for LiDAR segmentation and detection. Such negative effect is because $\mathcal{L}_{dep}$ imposes constraints to the density volume $V_{density}$ and enhances both foreground (*e.g.*, cars) and background objects (*e.g.*, trees). $\mathcal{L}_{sem}$ provides extra semantic information, which alleviates the performance drops and achieve the best results.

Supervision of different views. We provide the ablation experiments for both views. As shown in Table 5, LiDAR segmentation is more relevant with the supervision from camera view and 3D object detection is more sensitive to the supervision of BEV view. The camera view supervision can provide fine-grained geometry information which facilitates the LiDAR segmentation. However, the upper parts of camera view has very few LiDAR points for supervision (no LiDAR in the sky), thus the upper parts of density and semantic volumes are out of control. This could explain the degradation of detection performance when only supervising the camera view. BEV view can provide extra information and squeezing the upper parts of $V_{density}/V_{semantic}$ to meet their highest surface, such occlusion information is invisible in camera views and can restrain the degradation of detection.

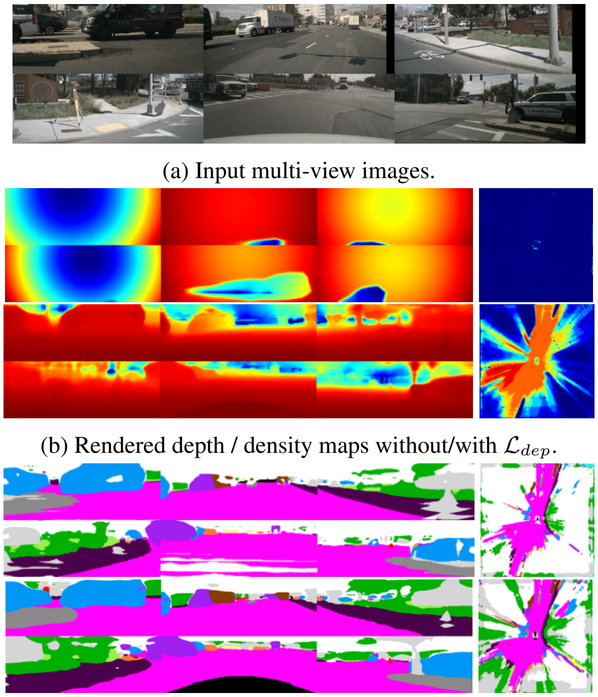
Method	Device	Params. ↓	Memory ↓	FPS↑
BEVNet (2023)	V100	39M	8G	4.5
VoxNet (2023)		72M	23G	1.9
OccNet (2023)		40M	18G	2.6
BEVFormer (2023)	RTX3090	-	4.5G	3.2
TPVFormer (2023)		-	5.1G	3.1
OccFormer (2023)		147M	5.9G	2.9
Vampire (ours)	RTX3080Ti	52M	5.0G	3.8

Table 6: Efficiency analysis. The experiments are all conducted with the corresponding device.

Efficiency Analysis

In Table 6, we compare the inference latency and memory of several methods. We find that the computational resources used in our methods are moderate. This makes the method practical and easy to use for the community.

Qualitative Results



(a) Input multi-view images.

(b) Rendered depth / density maps without/with $\mathcal{L}_{dep}$.

(c) Rendered camera / BEV semantic maps without/with $\mathcal{L}_{sem}$.

Figure 4: Visualizations of rendered results. The effectiveness of $\mathcal{L}_{dep}$ can be verified by Figure 4b, $\mathcal{L}_{dep}$ imposes constraints for learning reasonable 3D geometry information. The effectiveness of $\mathcal{L}_{sem}$ can be verified by Figure 4c, semantic regulation provides significant improvements in generating dense and meaningful features.

Conclusion

In this paper, we explore the connections between space occupancy in autonomous driving and volume density in NeRF, and propose a novel vision-centric perception framework, *i.e.*, Vampire, which takes volume rendering as the intermediate 3D feature regulator in the multi-camera setting. Vampire predicts per-position occupancy as the volume density and accumulate the intermediate 3D features to 2D planes to obtain additional 2D supervisions. Extensive experiments show that our method is competitive with existing state-of-the-arts across multiple downstream tasks.

Acknowledgements

This work was supported in part by The National Nature Science Foundation of China (Grant Nos: 62303406, 62273302, 62036009, 61936006), in part by Ningbo Key R&D Program (No.2023Z231, 2023Z229), in part by Yongjiang Talent Introduction Programme (Grant No: 2023A-194-G), in part by the Key R&D Program of Zhejiang Province, China (2023C01135).

References

Barron, J. T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; and Srinivasan, P. P. 2021. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5855–5864.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11621–11631.

Cao, A.-Q.; and de Charette, R. 2022. Monoscene: Monocular 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3991–4001.

Chabra, R.; Lenssen, J. E.; Ilg, E.; Schmidt, T.; Straub, J.; Lovegrove, S.; and Newcombe, R. 2020. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, 608–625. Springer.

Chang, J.-R.; and Chen, Y.-S. 2018. Pyramid stereo matching network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5410–5418.

Chen, X.; Lin, K.-Y.; Qian, C.; Zeng, G.; and Li, H. 2020. 3d sketch-aware semantic scene completion via semi-supervised structure prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4193–4202.

Cheng, R.; Wang, Z.; and Fragkiadaki, K. 2018. Geometry-aware recurrent neural networks for active visual recognition. *Advances in Neural Information Processing Systems*, 31.

Fong, W. K.; Mohan, R.; Hurtado, J. V.; Zhou, L.; Caesar, H.; Beijbom, O.; and Valada, A. 2022. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *IEEE Robotics and Automation Letters*, 7(2): 3795–3802.

Gan, W.; Mo, N.; Xu, H.; and Yokoya, N. 2023. A Simple Attempt for 3D Occupancy Estimation in Autonomous Driving. *arXiv preprint arXiv:2303.10076*.

Harley, A. W.; Fang, Z.; Li, J.; Ambrus, R.; and Fragkiadaki, K. 2022. Simple-BEV: What Really Matters for Multi-Sensor BEV Perception? *arXiv preprint arXiv:2206.07959*.

Harley, A. W.; Lakshmikanth, S. K.; Li, F.; Zhou, X.; Tung, H.-Y. F.; and Fragkiadaki, K. 2019. Learning from unlabelled videos using contrastive predictive neural 3d mapping. *arXiv preprint arXiv:1906.03764*.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16000–16009.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hu, A.; Murez, Z.; Mohan, N.; Dudas, S.; Hawke, J.; Badrinarayanan, V.; Cipolla, R.; and Kendall, A. 2021. FIERY: future instance prediction in bird's-eye view from surround monocular cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 15273–15282.

Huang, J.; Huang, G.; Zhu, Z.; Ye, Y.; and Du, D. 2021. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*.

Huang, Y.; Zheng, W.; Zhang, Y.; Zhou, J.; and Lu, J. 2023. Tri-Perspective View for Vision-Based 3D Semantic Occupancy Prediction. *arXiv preprint arXiv:2302.07817*.

Kerr, J.; Kim, C. M.; Goldberg, K.; Kanazawa, A.; and Tancik, M. 2023. LERF: Language Embedded Radiance Fields. *arXiv preprint arXiv:2303.09553*.

Li, Y.; Bao, H.; Ge, Z.; Yang, J.; Sun, J.; and Li, Z. 2022a. Bevstereo: Enhancing depth estimation in multi-view 3d object detection with dynamic temporal stereo. *arXiv preprint arXiv:2209.10248*.

Li, Y.; Ge, Z.; Yu, G.; Yang, J.; Wang, Z.; Shi, Y.; Sun, J.; and Li, Z. 2022b. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*.

Li, Z.; Wang, W.; Li, H.; Xie, E.; Sima, C.; Lu, T.; Qiao, Y.; and Dai, J. 2022c. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, 1–18. Springer.

Li, Z.; Yu, Z.; Wang, W.; Anandkumar, A.; Lu, T.; and Alvarez, J. M. 2023. FB-BEV: BEV Representation from Forward-Backward View Transformations. *arXiv preprint arXiv:2308.02236*.

Liu, G.; Reda, F. A.; Shih, K. J.; Wang, T.-C.; Tao, A.; and Catanzaro, B. 2018. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European conference on computer vision (ECCV)*, 85–100.

Liu, Y.; Wang, T.; Zhang, X.; and Sun, J. 2022a. Petr: Position embedding transformation for multi-view 3d object detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*, 531–548. Springer.

Liu, Y.; Yan, J.; Jia, F.; Li, S.; Gao, Q.; Wang, T.; Zhang, X.; and Sun, J. 2022b. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv preprint arXiv:2206.01256*.

Max, N. 1995. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2): 99–108.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Pan, M.; Liu, L.; Liu, J.; Huang, P.; Wang, L.; Zhang, S.; Xu, S.; Lai, Z.; and Yang, K. 2023. UniOcc: Unifying Vision-Centric 3D Occupancy Prediction with Geometric and Semantic Rendering. *arXiv preprint arXiv:2306.09117*.

Pan, S. J.; and Yang, Q. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10): 1345–1359.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 165–174.

Philion, J.; and Fidler, S. 2020. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, 194–210. Springer.

Roldao, L.; de Charette, R.; and Verroust-Blondet, A. 2020. Lmscnet: Lightweight multiscale 3d semantic completion. In *2020 International Conference on 3D Vision (3DV)*, 111–119. IEEE.

Sima, C.; Tong, W.; Wang, T.; Chen, L.; Wu, S.; Deng, H.; Gu, Y.; Lu, L.; Luo, P.; Lin, D.; and Li, H. 2023. Scene as Occupancy.

Sitzmann, V.; Thies, J.; Heide, F.; Nießner, M.; Wetzstein, G.; and Zollhofer, M. 2019. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2437–2446.

Tesla. 2022. Tesla AI Day. https://www.youtube.com/watch?v=ODSJsviD_SU.

Tian, X.; Jiang, T.; Yun, L.; Wang, Y.; Wang, Y.; and Zhao, H. 2023. Occ3D: A Large-Scale 3D Occupancy Prediction Benchmark for Autonomous Driving. *arXiv preprint arXiv:2304.14365*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, P.; Liu, L.; Liu, Y.; Theobalt, C.; Komura, T.; and Wang, W. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. *Advances in Neural Information Processing Systems*, 34: 27171–27183.

Wang, X.; Zhu, Z.; Xu, W.; Zhang, Y.; Wei, Y.; Chi, X.; Ye, Y.; Du, D.; Lu, J.; and Wang, X. 2023. OpenOccupancy: A Large Scale Benchmark for Surrounding Semantic Occupancy Perception. *arXiv preprint arXiv:2303.03991*.

Wang, Y.; Guizilini, V. C.; Zhang, T.; Wang, Y.; Zhao, H.; and Solomon, J. 2022. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, 180–191. PMLR.

Wei, Y.; Zhao, L.; Zheng, W.; Zhu, Z.; Zhou, J.; and Lu, J. 2023. SurroundOcc: Multi-Camera 3D Occupancy Prediction for Autonomous Driving. *arXiv preprint arXiv:2303.09551*.

Xie, E.; Yu, Z.; Zhou, D.; Philion, J.; Anandkumar, A.; Fidler, S.; Luo, P.; and Alvarez, J. M. 2022. Mˆ 2bev: Multi-camera joint 3d detection and segmentation with unified birds-eye view representation. *arXiv preprint arXiv:2204.05088*.

Yariv, L.; Gu, J.; Kasten, Y.; and Lipman, Y. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34: 4805–4815.

Ye, D.; Zhou, Z.; Chen, W.; Xie, Y.; Wang, Y.; Wang, P.; and Foroosh, H. 2022. Lidarmultinet: Towards a unified multi-task network for lidar perception. *arXiv preprint arXiv:2209.09385*.

Ye, M.; Wan, R.; Xu, S.; Cao, T.; and Chen, Q. 2021. Drinet++: Efficient voxel-as-point point cloud segmentation. *arXiv preprint arXiv:2111.08318*.

Zhang, Y.; Zhu, Z.; and Du, D. 2023. OccFormer: Dual-path Transformer for Vision-based 3D Semantic Occupancy Prediction. *arXiv preprint arXiv:2304.05316*.

Zhang, Y.; Zhu, Z.; Zheng, W.; Huang, J.; Huang, G.; Zhou, J.; and Lu, J. 2022. Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving. *arXiv preprint arXiv:2205.09743*.

Zhi, S.; Laidlow, T.; Leutenegger, S.; and Davison, A. J. 2021. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 15838–15847.

Zhou, B.; and Krähenbühl, P. 2022. Cross-view transformers for real-time map-view semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13760–13769.

Zhu, X.; Zhou, H.; Wang, T.; Hong, F.; Ma, Y.; Li, W.; Li, H.; and Lin, D. 2021. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9939–9948.