Multi-modal 3D Semantic Occupancy Representation for Autonomous Driving



Samuel Sze Tian Hong

Wolfson College

University of Oxford

A thesis submitted for the degree of

MSc by Research

Supervisors: Lars Kunze and Daniele De Martini

Hilary Term 2025

Abstract

Autonomous driving relies on advanced perception systems capable of accurately reconstructing and interpreting complex environments in three dimensions. Traditional 3D perception methods, such as object detection and semantic segmentation, contribute to scene understanding but often provide only a fragmented view of the surroundings. To bridge this gap, 3D semantic occupancy prediction offers a dense, voxelized representation of the environment, encoding both geometric structure and semantic information. However, developing an effective 3D semantic occupancy prediction framework presents challenges, including the need for real-time performance, robustness across diverse driving conditions, and efficient annotation methods to reduce reliance on labor and resource-intensive dense 3D labels, all while maintaining high semantic accuracy.

This thesis presents a multi-modal framework for 3D semantic occupancy prediction that combines LiDAR and cameras. The first contribution introduces a sparse convolution architecture optimized for processing 3D spatial data. By leveraging sparsity in voxelized representations, this approach reduces memory consumption and computational cost while maintaining spatial integrity and semantic accuracy. The second contribution, MinkOcc, reduces dependence on dense 3D annotations by incorporating weak supervision signals such as accumulated LiDAR sweep and 2D pseudo-labels generated from vision-language models. Additionally, it enhances the extraction of relevant information from both LiDAR and camera inputs, improving overall semantic and geometric accuracy from the first contribution.

The framework is validated on Occ3D-nuScenes, an extension of the nuScenes dataset with dense 3D occupancy annotations. Extensive experiments show that the proposed methods achieve competitive performance in controlled urban scenarios. The evaluations demonstrate improvements in computational efficiency, robustness, and generalization across diverse driving environments while maintaining label efficiency.

Keywords – 3D Semantic occupancy prediction, Sparse convolution, Sensor fusion, Weak supervision,

Label-efficient

Statement of Authorship

This thesis is submitted to the Department of Engineering Science, The University of Oxford, in fulfilment of the requirements for the degree of Master of Science by Research. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Samuel Sze Tian Hong, Feb 2025.

Funding

The work described in this thesis was funded by the Agency for Science, Technology and Research National Science Scholarship (Masters) and Oxford Robotics Institute's research project RobotCycle.

Acknowledgement

I would first like to thank my two advisors, Lars Kunze and Daniele De Martini, for their unwavering support and guidance. Thank you to everyone at the Oxford Robotics Institute and Wolfson College for being so welcoming and supportive. I am deeply grateful to all my friends, colleagues, teammates, administrators, and faculty who made my time at Oxford both transformative and memorable.

Last but not least, a huge thank you to my family for being constant sources of strength, love and encouragement.

Contents

1 Introduction **7**

1.1 How Autonomous Vehicles See the World: The Case for 3D Semantic Occupancy 7

1.2 Problem statement and contributions 11

1.3 Thesis Roadmap 12

2 Background **15**

2.1 Feature Extraction 16

2.1.1 Camera Feature Extraction 16

2.1.2 LiDAR Feature Extraction 21

2.2 Camera-Centric Semantic Occupancy 25

2.2.1 View Transformation 26

2.3 LiDAR-Centric Semantic Occupancy 31

2.3.1 Scene Completion 32

2.4 Spatial Fusion 33

2.4.1 Sensor Fusion 36

2.5 3D Occupancy Head 37

2.5.1 Convolutional Heads 38

2.5.2 Mask Decoder Heads 38

2.5.3 Linear Projection Heads 39

2.6 Training Objectives 39

2.7 Evaluation Metrics 41

2.8 Semantic Occupancy Datasets 44

2.8.1 nuScenes 44

2.9 Seminal Methods 46

2.10 Deploying Semantic Occupancy Models . . . 48
2.11 Generalized Semantic Occupancy Models . . . 49
2.11.1 Differentiable Rendering . . . 50
2.12 Summary . . . 50

3 Real-time 3D semantic occupancy prediction for autonomous vehicles using memory-efficient sparse convolution 53

3.1 Design Choice: Sparse Convolution . . . 62
3.1.1 Minkowski Engine . . . 62
3.1.2 Generative Minkowski Engine Operations . . . 64
3.2 Discussion . . . 65
3.3 Immediate Improvements . . . 66
3.4 Illustrative Results . . . 67
3.5 Further Work . . . 69

4 MinkOcc: Towards real-time label-efficient semantic occupancy prediction 71

4.1 Design Choice: Spherical Differentiable Rendering . . . 80
4.1.1 Point Rendering . . . 80
4.1.2 Spherical Rendering . . . 81
4.1.3 Comparison to Neural Radiance Fields (NeRF) . . . 83
4.1.4 Comparison to Gaussian Splatting . . . 83
4.2 Discussion . . . 84
4.2.1 Open-Vocabulary Mapping . . . 85

5 Discussion 88

5.1 Deployment . . . 88
5.1.1 Problem Statement . . . 88
5.1.2 Considerations and Criterion . . . 89
5.1.3 Analysis on Approach . . . 90
5.2 Robustness . . . 92
5.2.1 Problem Statement . . . 92
5.2.2 Considerations and Criterion . . . 92

5.2.3 Analysis on Approach . . . 93
5.3 Label-efficient . . . 100
5.3.1 Problem Statement . . . 100
5.3.2 Considerations and Criterion . . . 100
5.3.3 Analysis on Approach . . . 101

6 Conclusion and Future Work **108**
6.1 Conclusion . . . 108
6.2 Future Work . . . 110
6.2.1 Improving Weak-supervision . . . 110
6.2.2 Other Datasets . . . 111

Chapter 1

Introduction

1.1 How Autonomous Vehicles See the World: The Case for 3D Semantic Occupancy

The rise of autonomous vehicles (AVs) marks a transformative shift in mobility, promising to reduce human error and improve efficiency in urban transportation. At the core of an AV's ability to "see" is the perception module, a neural network-based pipeline that interprets raw sensor data and produces essential information for localization, planning, and control. Modern AVs leverage an array of sensors: cameras capture high-resolution texture and color in two dimensions, while LiDAR provide direct three-dimensional range measurements and light intensity. Despite their complementary strengths, cameras and LiDAR/radar fundamentally differ in representation—cameras operate in a two-dimensional (2D) image plane, whereas LiDAR natively produce three-dimensional (3D) coordinates.

Referring to figure 1.1, camera-based 2D perception has advanced significantly due to foundational breakthroughs in computer vision, driven by neural network architectures optimized for perspective-view image processing. These methods rely exclusively on RGB cameras to infer environmental attributes and spatial re-

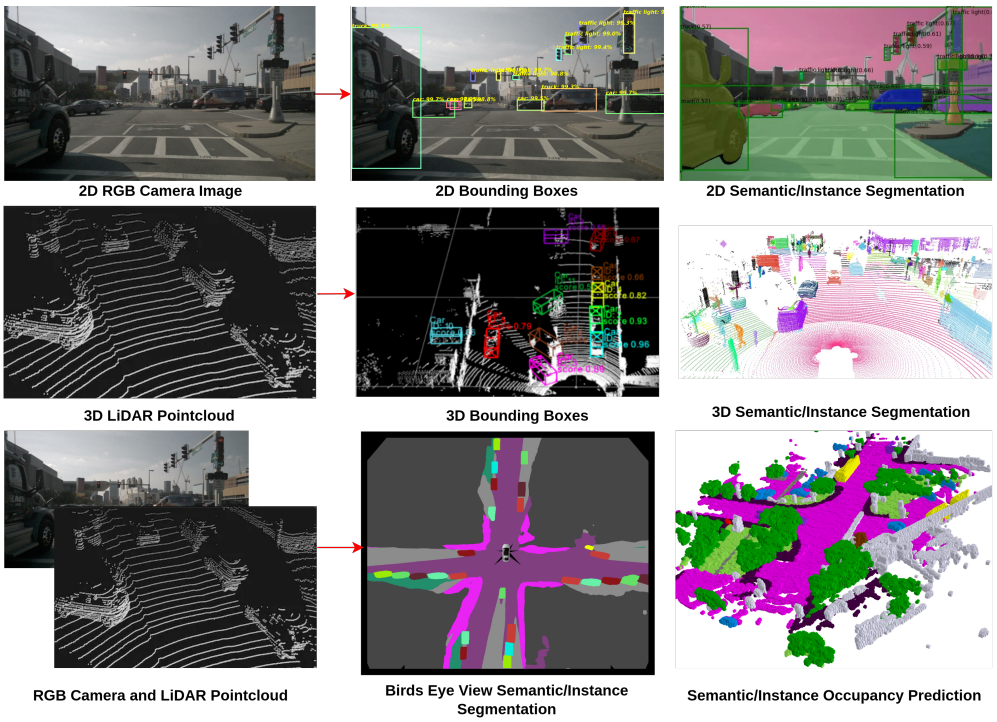


Figure 1.1: Conventional perception tasks utilizing 2D camera images include object detection and semantic segmentation in the perspective view. Similarly, with 3D LiDAR point clouds, one can perform 3D object detection and 3D semantic segmentation. Recently, with a combination of camera and LiDAR inputs, much research has aimed to improve 3D perception through BEV and semantic occupancy maps.

lationships within the camera's perspective frame. 2D object detection identifies objects such as vehicles and pedestrians using bounding boxes, offering an efficient localization and detection method. Expanding on this, segmentation techniques classify each pixel into "things" (e.g., vehicles, pedestrians, obstacles) and "stuff" (e.g., road, sky, vegetation) using convolutional neural networks and vision transformers, enabling detailed scene interpretation.

LiDAR-based 3D detection operates in real-world coordinates by analyzing point clouds generated from laser measurements. These point clouds allow neural networks to estimate object positions and dimensions, efficiently processing irregular 3D data. Among LiDAR-based tasks, 3D bounding box detection remains a primary focus, as it enables networks to predict oriented boxes that accurately localize objects and estimate their sizes in 3D space. In contrast, LiDAR seg-

mentation presents greater challenges due to the inherent sparsity of LiDAR data, particularly in outdoor environments and at longer distances. As a result, 3D segmentation primarily applies to scenarios with high-density 64-beam LiDAR sensors, focusing on larger objects such as vehicles, buildings, and road surfaces. Despite its sparsity, LiDAR provides precise depth measurements, offering essential spatial information for accurate distance estimation and reliable obstacle detection.

In recent years, Bird's Eye View (BEV) representation has emerged as a key approach for achieving unified spatial understanding across different sensor modalities. By projecting camera features into a top-down planar format through geometric or probabilistic methods and fusing them with height-flattened LiDAR features, BEV enables efficient sensor fusion within a unified reference frame. Early BEV research has prioritized key classes such as dynamic vehicles, drivable areas, and lane markings, balancing scene interpretation with computational efficiency. At the same time, improvements in neural network architectures for multi-dimensional processing, alongside advancements in specialized computing hardware, have significantly reduced the computational burden of BEV training and inference, making real-time perception increasingly feasible. The development of large-scale datasets with BEV and 3D semantic ground truth annotations further supports this progress, providing a foundation comparable in scale to camera and LiDAR datasets used in AV research.

However, in complex driving environments where semantic classes overlap vertically, the top-down BEV perspective struggles to capture many structures, limiting comprehensive scene understanding. This limitation becomes apparent in scenarios such as objects beneath overpasses, layered traffic with overhanging lights, or static and dynamic objects obstructing road surfaces. A natural solution involves

transitioning to a full 3D scene representation, where the environment appears as a dense volumetric grid, with each voxel evaluated for occupancy and assigned a semantic label. A full 3D representation not only captures vertical structures but also encapsulates scene features more comprehensively, allowing downstream tasks such as motion forecasting and path planning to operate with greater spatial awareness and precision. Advances in 3D data processing efficiency have further enabled research into these perception pipelines.

In conclusion, by leveraging the complementary strengths of camera and LiDAR sensors, along with improvements in neural network architectures and multi-modal datasets, 3D semantic occupancy prediction has emerged as a viable solution for AV perception, ensuring robust scene understanding in complex driving scenarios.

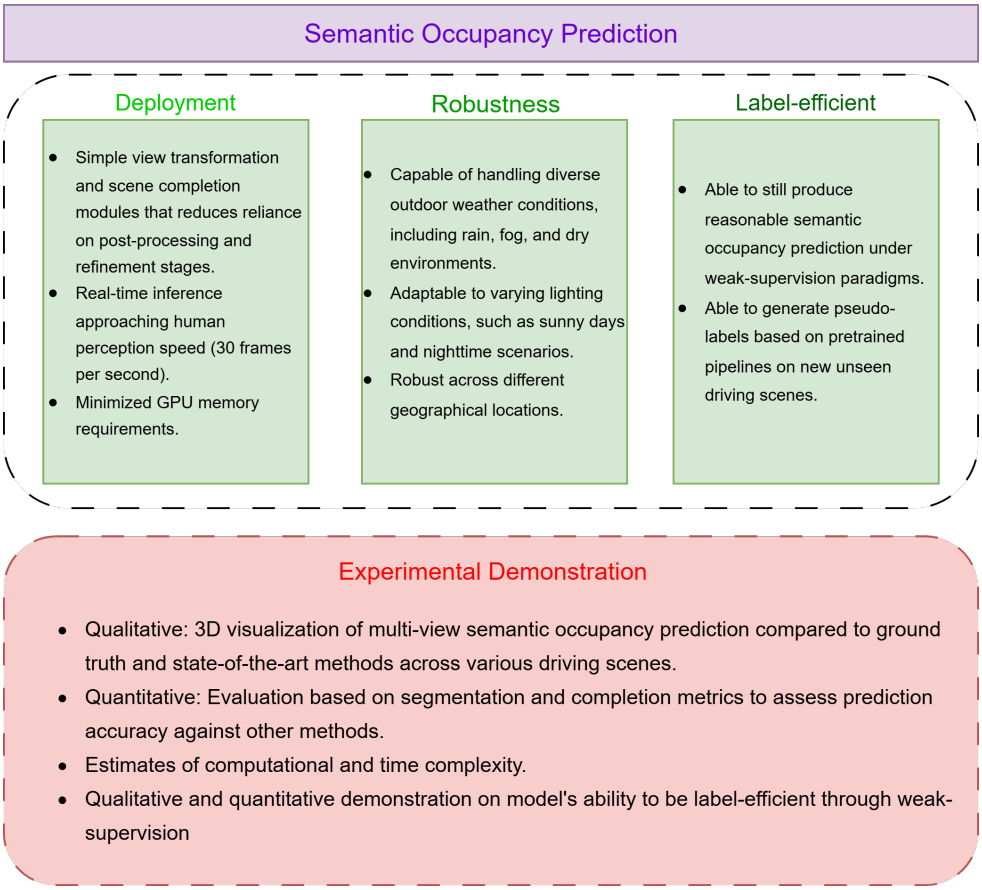


Figure 1.2: **Summary of our contributions.** The green boxes represent the main goals we aim to improve in 3D semantic occupancy prediction. We then analyze our approach using the metrics or criteria listed in the red box.

1.2 Problem statement and contributions

While significant progress has been made in 3D semantic occupancy perception for AVs, its practical deployment is hindered by several critical challenges. As illustrated in figure 1.2, these challenges fall into three domains: deployment-readiness, robustness, and label-efficiency. This thesis formally addresses these problems and propose our contributions as solutions.

1. **Deployment and first contribution**. Current methods for 3D semantic occupancy prediction often rely on computationally intensive architectures that process dense 3D grids. This approach leads to high latency and memory consumption, making it difficult to achieve the real-time inference speeds required for safe operation on the resource-constrained hardware found in autonomous vehicles.

   We define real-time performance as an inference rate of 20-30 frames per second (FPS). This target is grounded in the capture rates of LiDAR sensors, such as the 20 Hz LiDAR in the nuScenes dataset [1], preventing the model from inferring on a "stale" world representation. While higher frame rates might seem beneficial, they offer diminishing returns if they significantly surpass the sensor's frequency.

   To address this problem, our first contribution is **Real-time 3D semantic occupancy prediction for autonomous vehicles using memory-efficient sparse convolution**. Within, we introduce a novel, unified architecture that processes 3D information directly through geometric correspondences. By operating on sparse point representations instead of dense grids, our approach eliminates intermediate projection steps and reduces compu-

tational complexity, enabling real-time performance on standard hardware and making the system practical for deployment.

2. **Robustness, Label-efficiency and second contribution**. Beyond deployment, 3D semantic occupancy models also face a dual challenge. First, they must be robust, maintaining high performance across diverse weather, lighting, and geographical conditions where many existing models fail to generalize. Second, they must be label-efficient, as their dependency on large-scale, densely annotated 3D datasets creates a significant bottleneck for scaling to new environments.

   To address these interconnected challenges, our second contribution is **MinkOcc: Towards real-time label-efficient semantic occupancy prediction**. Building on our first contribution, MinkOcc introduces a multi-modal, semi-supervised training procedure. By warm-starting with a small set of dense, ground-truth 3D annotations and then continuing supervision with accumulated LiDAR sweeps and 2D pseudo-labels from vision models, MinkOcc reduces manual labeling requirements by 90%. This use of sensor-rich, weak supervision not only makes the system scalable but also enhances its robustness, enabling real-world deployment.

1.3 Thesis Roadmap

In this thesis, we focus solely on spatial understanding without temporal integration, addressing the fundamental challenge of reconstructing 3D scene representation from single-timestamp sensor observations. The remainder of this thesis is organized as follows:

Chapter 2 provides essential background knowledge. It details the building

blocks of a 3D semantic occupancy pipeline: feature extraction for camera and LiDAR sensors, view transformation approaches for cameras or sparse-to-dense scene completion for LiDARs, spatial fusion methods for combining multi-sensor information, and 3D occupancy head for decoding. It also examines the current state of self-driving datasets with its 3D annotation extensions, as well as seminal methods in 3D semantic occupancy prediction. Lastly, we introduce generalization and deployment concerns of semantic occupancy prediction which forms the foundation of our contributions.

Chapter 3 presents our first contribution: a real-time 3D semantic occupancy prediction system for autonomous vehicles using memory-efficient sparse convolution. This chapter details our novel unified architecture that simplifies 2D-to-3D projections and introduces an efficient scene densification approach. We present comprehensive experimental results demonstrating the effectiveness of our system across various scenarios.

Chapter 4 builds upon our first work by introducing MinkOcc: a label-efficient semantic occupancy prediction for AV. This chapter presents how we leverage rich 2D semantic and 3D geometric cues from cameras and LiDARs to act as weak supervision signals. In particular, we use LiDAR accumulation for occupancy completion and 2D pseudo labels with differential neural rendering for semantic segmentation. We also included detailed experimental results that validate MinkOcc's ability to perform semantic occupancy prediction across diverse driving scenarios with minimal training data.

Chapter 5 provides an in-depth analysis and evaluation of our approach with respect to the three main goals set forth in figure 1.2: deployment, robustness and label-efficiency.

Chapter 6 concludes the thesis and proposes directions for future work.

Chapter 2

Background

Semantic occupancy perception focuses on understanding both the geometric structure and semantic properties of a 3D scene by discretizing the continuous 3D space into a grid of voxels. This process maps the real-world space $W \in \mathbb{R}^3$ to a voxelized representation $V \in \{c_0, \ldots, c_n\}^{X \times Y \times Z}$, where X, Y, Z denote the grid dimensions, and each voxel is assigned one of $n + 1$ semantic classes c_0 through c_n (typically including an "empty" class to indicate free 3D space) [2]. Converting irregular 3D data into a structured 3D grid format enables the application of standard deep learning architectures for scene understanding. In addition, a voxel-based representation balances spatial granularity with computational efficiency, making it scalable for large 3D environments. Similar structured approaches such as bird's-eye view (BEV) representations have been widely adopted for autonomous driving tasks. Semantic occupancy grids extend the concept by encoding both the presence and semantic meaning of objects within a 3D space, preserving the height axis to provide a richer understanding of the scene. This thesis focuses on leveraging the 3D semantic voxel grid representation to enable detailed 3D scene understanding.

With reference to figure 2.1, 3D semantic occupancy prediction follows a structured pipeline consisting of four key stages. The process begins with sensor fea-

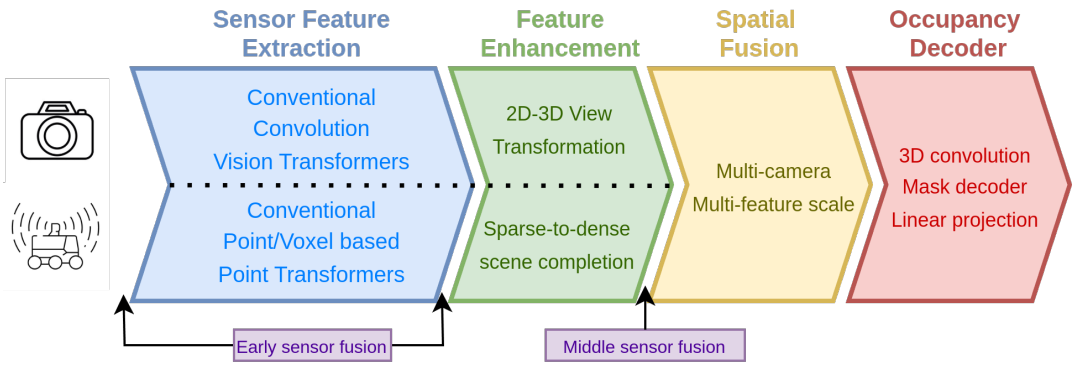


Figure 2.1: Typical pipeline of 3D semantic occupancy prediction models.

ture extraction, where raw input data from cameras and/or LiDARs is processed to obtain rich feature representations capturing both low-level patterns and high-level semantic information. These features then undergo view transformation or sparse-to-dense refinement, aligning them into a dense 3D space with a common coordinate system. The third stage, spatial fusion, integrates these transformed features into a coherent 3D representation. Finally, a 3D occupancy head processes the fused features to backpropagate training losses for both occupancy and semantic labels. For multi-modal approaches, additional fusion steps are incorporated into the pipeline. Early sensor fusion can be applied during feature extraction to combine complementary data streams from different sensors. In the spatial fusion stage, a mid-level sensor fusion step can integrate multi-modal features to enhance the final 3D representation. In the following sections, we examine each component of this pipeline in detail.

2.1 Feature Extraction

2.1.1 Camera Feature Extraction

Camera data is represented as a tensor $I \in \mathbb{R}^{H \times W \times 3}$, where H and W denote the height and width of the image in pixels, with the third dimension corresponds to the RGB color intensity channels. The primary challenge in camera feature

extraction is transforming these raw pixel values into meaningful representations that are robust to variations in scale, rotation, and illumination [3].

Conventional Feature Extraction

Traditional computer vision techniques focus on designing hand-crafted features that capture essential image properties while maintaining invariance to common geometric and photometric transformations. These methods identify distinctive local patterns such as corners (sudden intensity changes in multiple directions), edges (strong gradients), or blobs (regions distinct from their surroundings). The design of these features requires mathematical modeling of the visual attributes to be captured and the desired invariance properties.

One of the most influential methods in this domain is the Scale-Invariant Feature Transform (SIFT) [4], which addresses the challenge of scale variance in images. SIFT constructs a scale-space representation by convolving the input image I with Gaussian filters G_σ:

$$L(x, y, \sigma) = G_\sigma(x, y) * I(x, y). \tag{2.1}$$

Here, $L(x, y, \sigma)$ represents the blurred image at scale σ, obtained by convolving $I(x, y)$ with a Gaussian filter $G_\sigma(x, y)$. The convolution $*$ is computed by shifting the Gaussian kernel across the image, multiplying filter weights with corresponding pixel intensities, and summing the results. The scale parameter σ controls the level of smoothing, where larger values result in coarser representations [4]. Padding is typically applied to maintain the output dimensions.

To detect scale-specific features, SIFT computes the Difference-of-Gaussian (DoG) between adjacent scales:

$$D(x, y, \sigma) = L(x, y, k_S\sigma) - L(x, y, \sigma), \tag{2.2}$$

where k_S is a scaling factor defining the separation between consecutive scales. The DoG operation enhances features distinct at a given scale, facilitating stable feature detection for tasks such as object recognition [5].

Modern Approaches: From Convolutions to Transformers

Modern convolutional neural networks (CNNs) automate hierarchical feature extraction, learning multi-scale representations similar to SIFT but with learnable parameters [6]. The fundamental building block is the convolution layer, which generalizes SIFT's filtering operations. For an input tensor $X \in \mathbb{R}^{H\times W\times C_{\text{in}}}$, a convolution layer computes:

$$Y_c(h, w) = b_c + \sum_{i=1}^{C_{\text{in}}} \sum_{m=1}^{k} \sum_{n=1}^{k} W_{c,i,m,n} \cdot X_{i,h+m,w+n}. \tag{2.3}$$

Here, $Y_c(h, w)$ represents the output at spatial position (h, w) of channel c, while $W_{c,i,m,n}$ are the learnable convolution filter weights applied to input channel i. The bias term b_c is added to each element in channel c of the output. The indices m and n iterate over the kernel size k, enabling local neighborhood processing. The number of input channels C_{in} determines how multiple channels are combined in the output [6]. Unlike predefined Gaussian kernels used in SIFT, CNNs learn optimal filters W directly from data, enhancing adaptability to complex feature representations.

CNNs exhibit hierarchical structures, capturing low-level features (e.g., edges) in early layers and progressively more abstract patterns (e.g., textures, object components) in deeper layers. The receptive field expands as layers increase, allowing

CNNs to integrate global image context [7]. Many architectures use pooling or strided convolutions to downsample feature maps while increasing channel depth, achieving translational invariance and reducing computational costs.

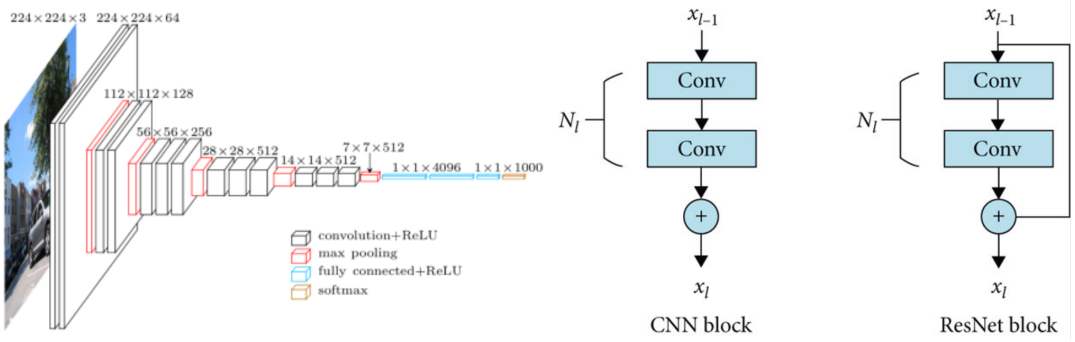


Figure 2.2: **Left Image**: Illustration of a Convolutional Neural Network (CNN) for 2D RGB image classification, focusing on convolutional layers, ReLU activation, and pooling operations for feature extraction. **Right Image**: A ResNet block incorporating residual connections to address vanishing gradients and enable deep network training through identity mapping. (Adapted from [8], [9].)

Residual Networks (ResNet) [9] introduce residual connections to mitigate vanishing gradient issues, allowing deep networks to maintain effective backpropagation. A residual block is expressed as:

$$F(x) = H(x) + x, \tag{2.4}$$

where $H(x)$ represents the transformation applied through convolutional layers, batch normalization, and ReLU activations, while x is the input. The skip connection allows x to bypass transformations, ensuring stable gradient propagation.

Despite CNNs' success in feature extraction, their reliance on localized receptive fields limits their ability to capture long-range dependencies. This constraint is particularly relevant in tasks requiring global spatial relationships, such as recognizing objects spanning distant image regions. Vision Transformers (ViTs) [10] address this limitation by adapting the transformer architecture, originally de-

signed for text, to the vision domain. Within ViTs, the patching mechanism transforms an image into a sequence of inputs. The image is first divided into a grid of fixed-size, non-overlapping patches. Each patch is then flattened and linearly projected into an embedding. This process converts the 2D image into a 1D sequence of patch embeddings, to which positional embeddings are added to retain spatial information. This sequence can subsequently be processed by the transformer's self-attention mechanism, which models dependencies across the entire image. The self-attention mechanism is defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{2.5}$$

where Q, K, and V are learned query, key, and value matrices derived from input image features, and d_k is the scaling factor for normalization. The dot product $QK^\top$ computes similarity scores between elements, followed by a softmax function to generate attention weights, which aggregate values from V [11]. This process allows every feature to be influenced by all other features in a single operation, capturing global relationships.

However, ViTs are computationally expensive, with self-attention operations scaling quadratically with the number of input patches. They also lack the spatial inductive biases inherent to CNNs, such as translation invariance, requiring large-scale pretraining on relevant datasets to achieve comparable performance [12]. These factors make ViTs more challenging to deploy in resource-constrained scenarios, such as autonomous driving.

Each feature extraction approach presents distinct advantages and trade-offs. SIFT provides interpretable, hand-crafted features with strong scale-invariance but lacks adaptability to diverse datasets. CNNs automatically learn hierarchical

features, balancing efficiency and robustness, though constrained by local receptive fields. Transformers offer global context modeling and superior feature representations but demand extensive computational resources and large-scale data. The selection of an appropriate camera feature extraction method depends on the constraints imposed by dataset size, computational limitations, and real-world application requirements.

2.1.2 LiDAR Feature Extraction

LiDAR data is typically represented as an unordered set of 3D points $P = \{p_i\}_{i=1}^{n}$, where each point $p_i \in \mathbb{R}^d$ contains spatial coordinates (x, y, z) and possibly other attributes (e.g., intensity). Unlike 2D images, point clouds are not arranged on a regular grid, making it challenging to extract features that capture both local geometric properties and higher-level semantics. Geometric attributes derived from local neighborhoods often serve as an effective starting point for understanding the structure of a scene.

Conventional Feature Extraction

To extract conventional geometric features, a neighborhood $N(p)$ is defined for each point p. This neighborhood can be determined using k-nearest neighbors or a fixed-radius search [13]. Once the neighborhood is identified, the centroid $\bar{p}$ is computed as:

$$\bar{p} = \frac{1}{|N(p)|} \sum_{q \in N(p)} q, \tag{2.6}$$

where q represents any point in $N(p)$ and $|N(p)|$ is the number of points in this neighborhood. The local geometric structure around p is characterized by the

spread of points in $N(p)$, captured using the covariance matrix C:

$$C = \frac{1}{|N(p)|} \sum_{q \in N(p)} \left(q - \bar{p}\right)\left(q - \bar{p}\right)^T. \tag{2.7}$$

By performing an eigen decomposition of C, the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ and corresponding eigenvectors v_1, v_2, v_3 are obtained [13]. Typically, the eigenvalues are ordered as $\lambda_1 \leq \lambda_2 \leq \lambda_3$, representing the variance along each eigenvector.

A key outcome of this analysis is the local surface normal, approximated by the eigenvector associated with the smallest eigenvalue, v_1. Since the direction of minimal variance is perpendicular to the dominant plane in the neighborhood, v_1 provides a good estimate of the normal vector. Beyond normals, eigenvalue ratios reveal additional shape information. A common measure of *planarity* is:

$$\text{Planarity} = \frac{\lambda_2 - \lambda_1}{\lambda_3},$$

where higher values indicate flatter surfaces [14]. These geometric cues, along with variations such as Fast Point Feature Histograms (FPFH), characterize local shape in a rotation-invariant manner, which is crucial for tasks like point cloud registration and object recognition [15].

Although geometric features are often used in LiDAR-based perception pipelines, they have limitations. Searching for neighbors (k-nearest or fixed-radius) can be computationally expensive for large LiDAR scans. Covariance-based features can be sensitive to noise, leading to unstable estimates of normals and shape descriptors. Moreover, geometric attributes do not provide high-level semantic information, such as object category or contextual cues. Since modern autonomous driving systems rely on semantic understanding (e.g., detecting cars, pedestrians, road signs), purely geometric approaches are often replaced by deep learning methods

that capture more nuanced contextual relationships [16].

Learning Approaches

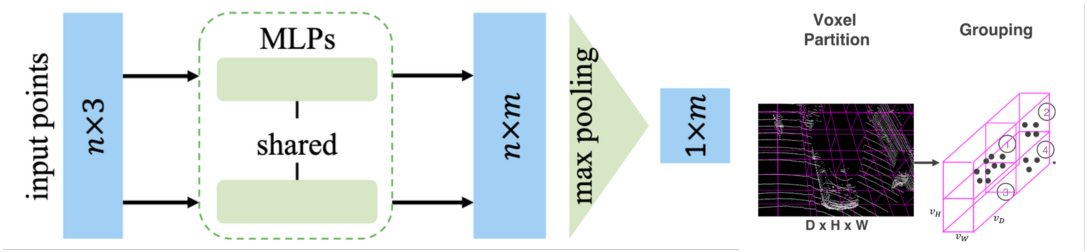


Figure 2.3: **Left Image**: Simplified architecture of PointNet. T-Net modules are commonly included before MLPs. For segmentation, the max-pooled global descriptor is concatenated with point features. **Right Image**: VoxelNet partitions LiDAR point clouds into voxel grids for 3D convolutions. (Adapted from [17], [18].)

PointNet [17] introduces a learning-based paradigm that processes raw point clouds without relying on hand-crafted geometric features. A key component is the T-Net module, which learns geometric transformations to achieve invariance to input point cloud rotations and translations, ensuring consistent feature extraction. The network uses multi-layer perceptrons (MLPs) applied to individual points, combined with symmetric max or average pooling functions to aggregate these learned point-wise features into a global descriptor. At each layer l, the model updates the feature of a point p by looking at its neighbors q, taking their relative positions $(q - p)$ and their previous-level features $f_{l-1}(q)$ as inputs:

$$f_l(p) = \max_{q \in N(p)} h_l\Big([\, q - p;\, f_{l-1}(q)\Big]), \tag{2.8}$$

where h_l is an MLP learned from data. Applying a symmetric function like max over $N(p)$ ensures permutation invariance, allowing the network to learn meaningful representations directly from raw 3D coordinates. Despite its simplicity, PointNet captures both local patterns (through the MLP's non-linear transforms) and a global context (via pooling), making it effective for 3D classification and

segmentation tasks.

A limitation of PointNet arises from its global feature aggregation. While this captures high-level context, it may overlook fine-grained spatial relationships. VoxelNet [18] addresses this by voxelizing the 3D space into a dense grid of small cubes (voxels). Each voxel aggregates points within its boundaries and uses Voxel Feature Encoding (VFE) layers to combine point-wise features with locally aggregated features, capturing finer spatial details.

After aggregating features within each voxel, VoxelNet arranges the voxel-wise features into a structured 3D tensor, suitable for processing by 3D convolutional neural networks (CNNs). These CNNs capture spatial relationships across voxels, similar to how 2D CNNs process images. The CNN-based hierarchical feature extraction enables VoxelNet to learn both local geometric structures (via voxel-level MLPs) and broader spatial patterns (via 3D convolutions). VoxelNet enables structured representation, allowing standard convolutional architectures to process LiDAR data efficiently. However, it introduces a trade-off: low-resolution voxel grids may lose fine details (e.g., thin objects or distant points), while high-resolution grids increase memory and computation costs. Many subsequent works try to optimize voxelization (e.g., using sparse convolutions or adaptive voxel sizes) to balance efficiency and accuracy [19].

Similar to Vision Transformers, recent methods in LiDAR feature extraction alleviate the locality constraints of point-based and voxel-based approaches by incorporating attention mechanisms. Point Transformers extend self-attention to point cloud data, capturing both local and global dependencies without relying on predefined neighborhoods [20]. A point's feature $f(p)$ is updated by attending to its neighbors:

$$f(p) = \sum_{q \in N(p)} \text{softmax}\Big(\phi(p, q)\Big) \cdot h(q), \tag{2.9}$$

where $\phi(p, q)$ is a learnable similarity function, and $h(q)$ represents the feature of neighbor q. By assigning attention weights via softmax, the network emphasizes relevant neighbors based on learned importance [20]. This enables Point Transformers to model long-range dependencies effectively. However, once again, self-attention incurs high computational overhead, limiting scalability for large-scale LiDAR data.

In summary, LiDAR data presents challenges due to its unstructured nature. Early methods relied on geometric features such as surface normals and shape metrics, but these were computationally expensive and lacked semantic richness. Deep learning approaches such as PointNet, VoxelNet, and Point Transformers address these issues by learning representations directly from raw point clouds, improving efficiency and accuracy in autonomous driving applications.

2.2 Camera-Centric Semantic Occupancy

Vision-centric approaches often require transforming 2D image features into a unified 3D representation to facilitate downstream tasks. The view transformation step bridges the gap between the 2D feature space of camera images and the 3D voxel grid representation necessary for semantic occupancy prediction.

At this stage, the extracted 2D features, denoted as $F_{\text{2D}}(u, v)$, undergo a mapping process that projects them into a 3D voxel grid. This process leverages camera geometry, intrinsic parameters, and depth information—either explicitly provided or inferred using learned models. By aligning the features spatially in 3D space, the transformation facilitates coherent integration of information across multiple

views. The resulting 3D voxel grid representation has predefined dimensions and voxel size to align with the requirements of subsequent processing steps. Nevertheless, while this transformation enables structured representation of the scene, it inherently introduces certain distortions due to the limitations of projection. These distortions are influenced by occlusions, inaccuracies in depth estimation, and variations in camera alignment. As such, 2D-to-3D view transformation has been an ongoing research area in the field of 3D semantic occupancy prediction as well.

Figure 2.4 illustrates three main methods of view transformation, highlighting the mapping from 2D features to a 3D voxel grid.

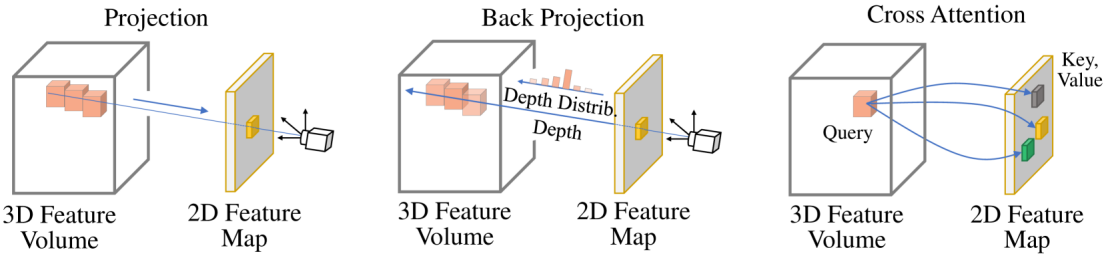


Figure 2.4: **Left**: Projection/Forward projection transforms fixed 3D locations to corresponding 2D image pixels/features to sample and extrapolate back to 3D.
Middle: Back projection estimates probabilistic distance rays from 2D image pixels/features.
Right: Cross-attention transformation queries 2D image pixels/features for each 3D location. (Source from [2].)

2.2.1 View Transformation

The foundation of view transformation begins with the pinhole camera model, which defines how 3D points in the physical world are mapped onto a 2D image plane. In this model, a point $\left(X, Y, Z\right)$ in world coordinates is projected onto the image plane as:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R \mid t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix},$$

where K is the intrinsic matrix, containing parameters such as focal lengths (f_x, f_y), the principal point $\Big(c_x, c_y\Big)$, and skew. The extrinsic matrix $[R \mid t]$ encodes the rotation and translation between the world and camera frames. The scalar s is a depth-dependent factor that normalizes the homogeneous coordinates $\Big(u, v, 1\Big)^\top$ in the image plane.

Forward Projection

Forward projection starts by defining 3D anchors either via a discretized voxel grid or LiDAR point clouds. Then, it projects these anchors onto the image plane using the pinhole camera model. Bilinear interpolation is then applied to sample features from the 2D image feature map and assign them back to the 3D anchors.

In the discretized voxel-grid approach, the 3D space is divided into a regular grid to represent occupancy states and semantic classes . The grid is normally defined in an ego-vehicle-centered coordinate system, where the X-axis points forward, the Y-axis points left, and the Z-axis points upward. For example, a typical configuration might span a 50-meter radius in the X-Y plane and 8 meters in the Z-axis (height), with a resolution of 0.4 meters per voxel. This results in a grid of approximately $200 \times 200 \times 20$ voxels. The center of each voxel, represented by $\Big(x, y, z\Big)$, is projected onto the image plane using (2.2.1) to compute 2D coordinates $\Big(u, v\Big)$. Since these coordinates often do not align with discrete image pixels, bilinear interpolation is used to sample features from the 2D image feature map. This interpolation combines features from the four nearest pixels to $\Big(u, v\Big)$,

weighted by their proximity to the projection point. The sampled features are then assigned back to their corresponding voxel, enabling a filled 3D voxel feature representation. Forward projection underpins many approaches [21–23] focusing on efficient feature sampling and occupancy prediction. A limitation of this is the ambiguity caused by multiple voxels along the same line of sight projecting to the same $\left(u, v\right)$ coordinates, which can lead to inaccurate feature representations despite the high density of voxel grids.

LiDAR-assisted forward projection approach primarily complements vision-centric semantic occupancy prediction by serving as an auxiliary channel for accurate 2D-to-3D transformation. A sensor commonly used in many autonomous vehicle platforms, LiDAR explicitly measures depth, enabling its integration with camera data for multi-modal processing. Similar to the voxel-based method, the approach transforms each LiDAR point $\left(X_l, Y_l, Z_l\right)$ into the camera frame and projects it onto the image plane via (2.2.1), followed by bilinear interpolation to sample features. Several methods [24, 25] use this simple multi-modal strategy to project camera features into 3D space and achieve competitive accuracy in various tasks. However, the sparse distribution of LiDAR points compared to the dense voxel grid creates a trade-off: while LiDAR ensures precise depth information for accurate feature retrieval, it limits the amount of 2D camera information that are brought into 3D space for occupancy prediction. Additionally, relying on LiDAR sensors makes this method inherently multi-modal, requiring both camera and LiDAR data.

Cross Attention

Cross attention represents an evolution of forward projection methods, introducing learnable parameters into the view transformation process. While conventional

voxel or point-based forward projection directly maps predetermined 3D locations to 2D image coordinates, cross attention adds a learnable query-key-value mechanism to this geometric foundation.

The process begins similarly to normal forward projection, where a 3D grid of points is established in the scene. However, instead of directly sampling features from the projected 2D locations such as bilinear interpolation, each 3D grid point now serves as a query in the attention mechanism. The mathematical formulation for this cross attention can be expressed as:

$$F_{3D}(q) = \sum_{ij} A_{ij} W_{ij} F_{2D}(p + \Delta p_{ij}) \tag{2.10}$$

where q represents the 3D query position, and p is its projected 2D location using standard perspective projection. The term $F_{2D}(p)$ denotes the extracted 2D feature at location p. Indices i and j enumerate different spatial sampling points on the image plane. W_{ij} is the learnable weights, while A_{ij} represents attention scores that determine the contribution of different sampled locations. The term Δp_{ij} denotes learnable position offsets from the projected point p, allowing the model to adjust sampling positions dynamically.

Since it is not practical to sample every single offset in an image for each projected 3D point, many methods adopt deformable cross attention [26–30], where each query interacts with a selected subset of features around its projected location rather than attending to the entire feature map. This approach significantly reduces computational complexity while maintaining the benefits of learnable feature sampling [31].

The key advantage of cross attention infused approach is that it combines the geometric structure of forward projection with the flexibility of learned feature

sampling. The network can learn optimal sampling patterns around each projected point, potentially compensating for projection errors or capturing more relevant contextual information. However, the computational complexity of cross attention operations, even with deformable variants, remains a significant consideration.

Backward Projection

While forward projection begins with fixed 3D locations and projects them onto the image plane, backward projection reverses this process by starting with image pixels and lifting them into 3D space through probabilistic depth estimation. Mathematically, this can be expressed as:

$$F_{3D}(x, y, z) = F_{2D}(u, v) \otimes P(d \mid u, v) \tag{2.11}$$

where F_{3D} represents the 3D feature volume at coordinates (x, y, z), F_{2D} represents 2D image features at pixel coordinates (u, v), $P(d \mid u, v)$ denotes the predicted discrete depth distribution along the camera viewing ray, and $\otimes$ indicates an outer product operation that distributes the 2D features across the depth dimension according to the predicted probabilities. The viewing ray originates at the camera's optical center and extends outward through the pixel (u, v) on the image plane, representing all possible 3D points along that direction in the scene. This is similar to the concept of raymarching [32]. Therefore, backward projection treats depth as a probabilistic distribution along each pixel's viewing ray, allowing the model to express uncertainty in scenarios with ambiguous depth perception, such as partially occluded objects or regions with complex geometry.

When working with multiple camera views, forward projection methods can struggle when incorrect depth estimates prevent proper feature integration at

shared 3D locations across different camera viewpoints. In contrast, backward projection's probabilistic nature enables more robust information fusion by incorporating depth uncertainty into the process. This fully differentiable approach facilitates end-to-end training of the entire 3D semantic occupancy pipeline, as demonstrated by methods like Lift, Splat, Shoot [32], and has subsequently become widely adopted [33–36].

Nevertheless, the depth estimation network within backward projection still requires supervision during training. This supervision typically comes from LiDAR. The LiDAR points are projected onto the image plane using forward projection to create supervised training signals for the depth estimation network. In terms of computational efficiency, backward projection can be more efficient than dense voxel grid approaches as it focuses processing on actual image content rather than predetermined 3D locations.

2.3 LiDAR-Centric Semantic Occupancy

Following feature extraction from LiDAR point clouds in section 2.1.2, LiDAR-centric semantic occupancy prediction faces a fundamental challenge: transforming these sparse feature representations into a dense occupancy grid. The sparsity issue inherent in LiDAR data manifests in three key ways. First, point density decreases quadratically with distance from the sensor - a physical limitation governed by the angular resolution of the LiDAR beam. As the laser beam diverges with distance, the spacing between adjacent points grows proportionally to the square of the distance, resulting in significantly sparser sampling in distant regions. Second, occlusions behind objects create information gaps. Lastly, the limited vertical resolution from LiDAR sensor design leads to irregular sampling

patterns [2]. These limitations necessitate scene completion strategies to infer occupancy in unobserved regions.

2.3.1 Scene Completion

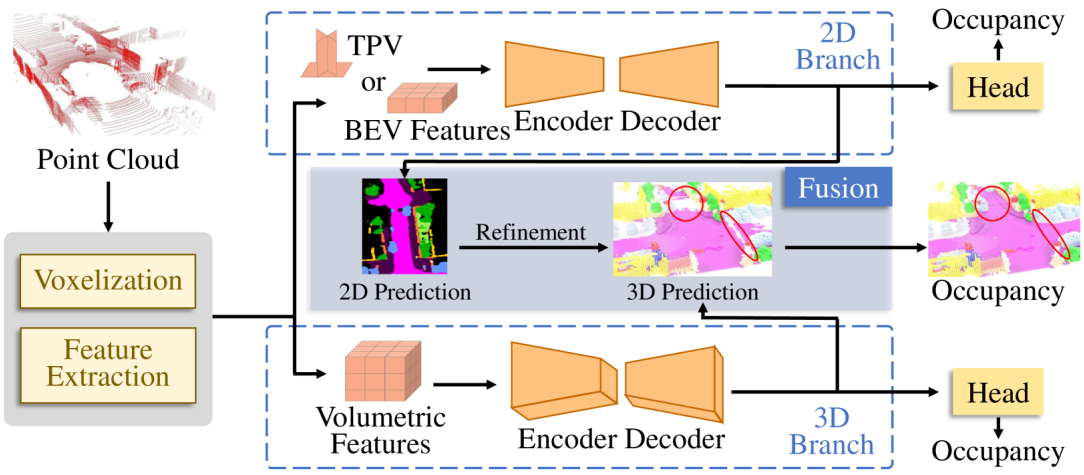


Figure 2.5: Illustration of LiDAR-centric semantic occupancy prediction. Scene completion can be done purely on the 3D branch using sparse 3D volumetric features, or utilizing multi-projection fusion (e.g., TPV and BEV) to leverage 2D convolutional networks, followed by fusion with densified 3D features to enhance scene understanding. (Source from [2].)

Scene completion can be approached through two primary paradigms. Traditional methods employ local feature propagation, defining neighborhoods around observed points and spreading information based on geometric and feature similarity. While computationally efficient, these approaches struggle with large-scale outdoor scene completion with complex scene structures.

Referring to figure 2.5, scene completion can also be accomplished via modern learning-based approaches. These typically leverage multi-scale processing networks such as U-Net [37] or cascaded residual blocks [38] with dedicated enhancement modules like Atrous Spatial Pyramid Pooling (ASPP) [39] to capture both fine geometric details and broader geometrical scene context. Sparse convolutional networks [19,40–42] have also emerged as an efficient solution. Processing only non-zero points regardless of spatial distance, these networks significantly ac-

celerate 3D convolution operations while allowing distant points to interact within the same convolutional kernel. This interaction helps establish relationships between spatially separated features despite the quadratic density falloff inherent in LiDAR data. Additionally, these specialized modules are trained end-to-end, enabling the model to learn optimal feature propagation strategies for accurate scene completion [43].

Recent works have also shown significant improvements in scene completion through multi-projection fusion [44,45], which begins by projecting sparse 3D features into different complementary viewpoints. For example, the Bird's Eye View (BEV) projection creates a top-down representation by collapsing height information into channels, which is particularly effective for understanding road-level geometry and object relationships in the driving plane. The Tri-Perspective View (TPV) approach [44] projects features into three orthogonal planes - front, side, and top views - enabling better capture of vertical structures and object heights while maintaining computational efficiency. Each projected view is processed using 2D convolutional networks to extract view-specific features, which are then lifted back to 3D space and fused with the original 3D features. This multi-perspective approach leverages the computational efficiency of 2D convolutions while maintaining crucial 3D geometric relationships, albeit having a much more complex network architecture.

2.4 Spatial Fusion

Spatial fusion ensures consistent geometric relationships and spatial context across different input sources. A key example is in scene completion under LiDAR-centric semantic occupancy, where 2D convolutional features from different projections

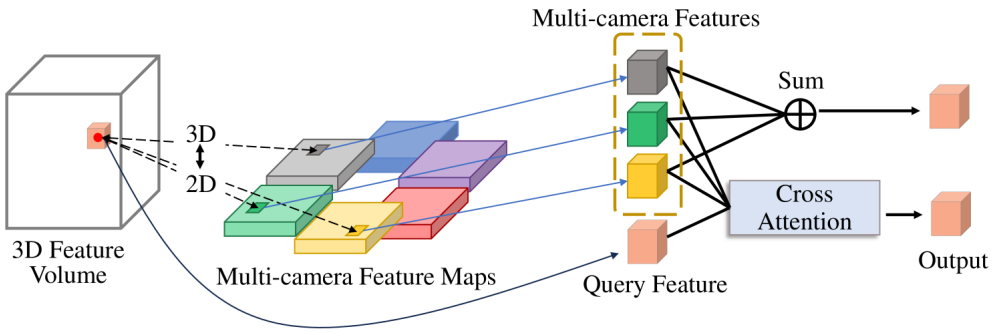


Figure 2.6: Illustration of camera-centric spatial fusion methods, demonstrating how pixels/features from multiple camera viewpoints projected and integrated to address overlapping regions in the camera fields of view which points to the same 3D space. (Source from [2])

(TPV, BEV) are lifted back to 3D space and fused with volumetric 3D features. With reference to figure 2.6, another example is in camera-centric semantic occupancy where spatial fusion is performed across features from multiple camera viewpoints, addressing regions where camera fields of view overlap. This fusion mechanism must account for varying camera perspectives and potential redundancies in the captured information.

Two primary approaches exist for spatial fusion. The first involves feature aggregation, where features from different sources are combined through averaging, summation or concatenation operations [46–49]. Given N different feature sources and a 3D position q, the basic aggregation can be expressed as:

$$F(q) = \frac{1}{N} \sum_{i=1}^{N} F_i(q) \tag{2.12}$$

where $F_i(q)$ represents the feature from source i at position q. This can be extended with learned weights w_i to create an adaptive fusion [25]:

$$F(q) = \sum_{i=1}^{N} w_i \cdot F_i(q) \tag{2.13}$$

where w_i is computed through network layers to adaptively weight different

feature sources based on their importance or reliability.

Cross attention provides another mechanism for feature fusion by computing weighted relationships between source features F_{src} and target features F_{tgt} [23,50, 51] through a multi-head attention operation:

$$F_{\text{fused}} = \text{MultiHead}(F_{\text{tgt}}, F_{\text{src}}) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2.14}$$

where:

$$\text{head}_i = \text{softmax}\left(\frac{F_{\text{tgt}}W_i^Q(F_{\text{src}}W_i^K)^T}{\sqrt{d}}\right)F_{\text{src}}W_i^V \tag{2.15}$$

With reference to equation 2.14 and 2.15, multi-head attention mechanism sub-divides the attention operation into h parallel heads, with each head focusing on distinct aspects of the feature relationships. Each head processes a lower-dimensional projection of the input features using the learned projection matrices W_i^Q, W_i^K, and W_i^V. Then, the outputs from all heads are concatenated together (Concat) and passed through a final linear projection W^O to produce the fused features. This concatenation combines the different types of attention patterns learned by each head into a single comprehensive feature representation. The feature dimension d appears in the scaling factor $\sqrt{d}$ to prevent the dot product within each head from growing too large in magnitude, especially for high-dimensional features. In camera multi-view spatial fusion scenarios, F_{tgt} represents features from one camera view, while F_{src} contains features from other overlapping camera views. The attention weights determine how these different types of features should be combined based on their spatial and semantic relationships.

The choice between aggregation and attention-based approaches often depends

on the specific requirements of the fusion scenario and computational constraints.

2.4.1 Sensor Fusion

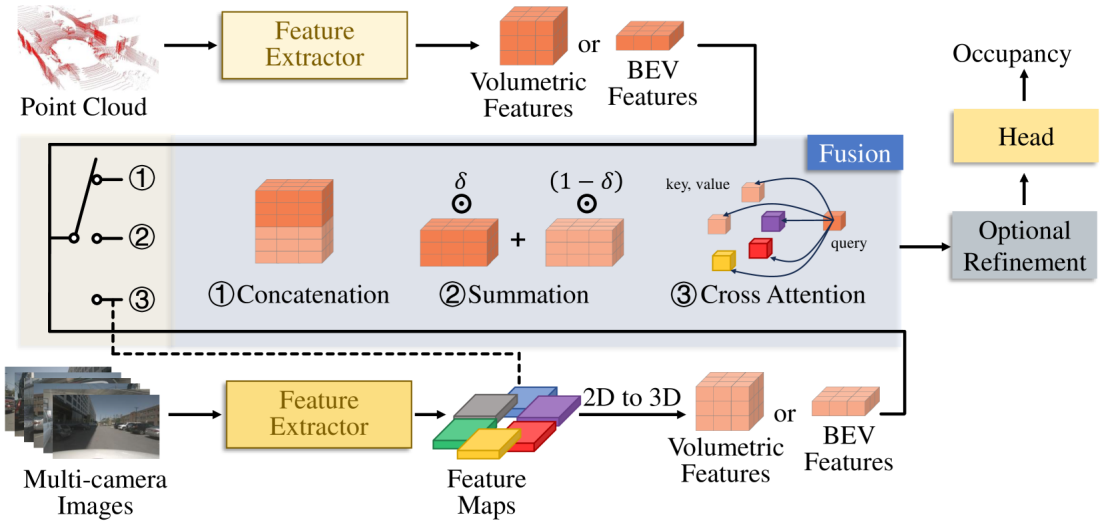


Figure 2.7: Illustration of sensor fusion in multi-modal semantic occupancy pipelines, depicting how features from LiDAR and cameras are integrated. (Source from [2])

Sensor fusion in multi-modal semantic occupancy pipelines can be seen as another case of spatial fusion, except now the different input sources are the different sensors (e.g., LiDAR and cameras). Similar to the goal of spatial fusion, sensor fusion aims to combine features from these different sources while preserving their geometric and contextual information in 3D space.

With reference to figure 2.1, for multi-modal semantic occupancy pipelines, sensor fusion can occur at one of two stages: early fusion during feature extraction or middle fusion after camera 2D-to-3D view transformation enhancement. These fusion approaches employ similar technical strategies to spatial fusion: either feature aggregation through weighted summation (with fixed or learned weights adapting to sensor reliability) or cross-attention mechanisms that compute dynamic weighted relationships between source and target features from different sensors.

Early fusion [29, 30] typically employs a projection-based approach to allow direct feature concatenation in the 2D image space. This has the advantage of allowing the network to learn joint feature representations from raw sensor data, potentially capturing fundamental cross-modal correlations that might be lost in separate processing streams. This approach is computationally simple since features are processed together through most of the network. However, early fusion can suffer from issues due to LiDAR projections being sprse and occluded. It can also be challenging to implement early fusion effectively because features from different sensors are still in their native formats with varying scales, densities, and representations.

Mid-sensor fusion [49, 52] is applied after feature extraction and enhancement, where both sensor modalities have been transformed into a common 3D representation. This makes the fusion process similar to spatial fusion. Mid-level fusion allows sensor features to be extracted and enhanced through individual specialized processing streams optimized for their unique characteristics, better preserving modality-specific information and allowing for more flexible architecture design. However, it requires maintaining separate processing streams in the network, which can increase computational overhead and memory requirements.

2.5 3D Occupancy Head

The final component in 3D semantic occupancy prediction involves transforming spatially fused 3D features into occupancy and semantic predictions for each voxel in the 3D space. Several architectural approaches have emerged for this task: 3D convolutional heads, mask decoder heads, and linear projection heads.

2.5.1 Convolutional Heads

Convolutional heads employ a series of 3D convolution operations to process the fused features, often incorporating multi-scale feature aggregation through skip connections in a U-Net style architecture. The discussed processes of spatial fusion, scene completion in LiDAR-centric approaches [43], and view transformation in camera-centric approaches [51], may all be encapsulated within this overarching U-Net structure. 3D convolution starts off by progressively downsampling the input features through an encoder path while capturing increasingly abstract representations. Then, these features are upsampled through a decoder path with skip connections aggregating corresponding encoder features to preserve fine-grained spatial details [37]. While effective at capturing spatial relationships, the computational cost of 3D convolutions can be substantial for high-resolution voxel grids [2].

2.5.2 Mask Decoder Heads

Inspired by advances in 2D instance segmentation approaches like Mask2Former [53], mask decoder heads reformulate semantic occupancy prediction as a set prediction problem. Rather than directly classifying each voxel, these approaches transform the input 3D feature volume into per-voxel embeddings ($E_v \in \mathbb{R}^{H \times W \times D \times d}$), where each voxel is encoded into a d-dimensional feature vector capturing local geometric and semantic characteristics. The network also maintains a set of K learnable query embeddings ($Q \in \mathbb{R}^{K \times d}$), one for each semantic category, which act as prototype vectors in the embedding space. The interaction between voxel embeddings and semantic queries is computed through a scaled dot-product attention mechanism:

$$M_{k,v} = \text{softmax}\left(\frac{Q_k \cdot E_v}{\sqrt{d}}\right) \tag{2.16}$$

Here, $M_{k,v}$ represents the probability that voxel v belongs to semantic category k, with the scaling factor ($\sqrt{d}$) included to prevent numerical instability in high-dimensional embedding spaces. By incorporating an empty class query, the network handles both occupancy and semantic predictions within a unified framework [54].

2.5.3 Linear Projection Heads

Linear projection heads offer a lightweight alternative by applying channel-wise MLPs to the feature volume [23, 55]. For each voxel position, a simple linear projection predicts both occupancy probability and semantic class probabilities:

$$[p_{\text{occ}}, p_{\text{sem}}] = \text{MLP}(F_{x,y,z}) \tag{2.17}$$

While simpler than the other two deocder heads, linear projection heads can be effective when the input 3D features already capture sufficient spatial and semantic context through earlier stages of the network.

2.6 Training Objectives

Training typically involves a combination of occupancy and semantic loss terms:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{occ}}\mathcal{L}_{\text{occ}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} \tag{2.18}$$

For geometric supervision of occupancy, Binary Cross-Entropy (BCE) loss is predominantly used [56]:

$$\mathcal{L}_{\text{occ}} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{2.19}$$

Here, y_i is the ground truth binary occupancy state (1 for occupied, 0 for empty), and $\hat{y}_i$ is the network's predicted probability of occupancy for each voxel i. The loss encourages confident predictions by heavily penalizing cases where the network assigns low probabilities to occupied voxels or high probabilities to empty ones. Alternative geometric losses include scale-invariant logarithmic loss [57], which offers robustness to scale variations, and soft-IoU loss [58], which better optimizes intersection-over-union metrics.

For semantic supervision, Weighted Cross-Entropy (CE) loss is standard [59]:

$$\mathcal{L}_{\text{sem}} = -\frac{1}{N}\sum_{i=1}^{N} \omega_{c_i} \log(p_{c_i}) \tag{2.20}$$

where p_{c_i} is the predicted probability for the correct class c_i of voxel i, and ω_{c_i} is a weighting factor that assigns higher importance to rare classes. This loss encourages accurate semantic classification while accounting for class imbalance through the weights ω. Some methods incorporate additional semantic losses such as Lovász-Softmax [60] for better handling of imbalanced classes or Focal loss [61] to address the imbalance between easy and hard samples.

For mask decoder approaches, an additional Hungarian matching loss [53] may be included to establish optimal correspondence between predicted and ground truth segments. Some methods also incorporate auxiliary losses, such as feature consistency between different views or regularization terms, to encourage spatially smooth predictions.

2.7 Evaluation Metrics

In the domain of 3D semantic occupancy prediction, the primary objective is to determine whether a voxel in 3D space is occupied, and to classify it accurately based on its semantic content. Evaluating these predictions requires metrics that assess both occupancy accuracy and semantic classification performance.

For occupancy, the evaluation metric for this task is Intersection-over-Union (IoU), which quantifies the accuracy of the predicted occupancy map compared to the ground truth. IoU measures the ratio of the overlapping volume between the predicted and actual occupied voxels to the total volume of their union:

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{2.21}$$

where:

- TP (True Positives): The number of correctly predicted occupied voxels.
- FP (False Positives): The number of voxels incorrectly predicted as occupied.
- FN (False Negatives): The number of occupied voxels that were not predicted as such.

In addition to IoU, Precision and Recall are often employed to provide a more comprehensive evaluation of the occupancy prediction performance.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.22}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.23}$$

Precision evaluates how many of the predicted occupied voxels are actually

correct, while recall measures the fraction of occupied voxels correctly identified. Furthermore, for tasks involving multiple semantic classes, the mean Intersection-over-Union (mIoU) is often used. mIoU is computed by calculating the IoU for each class independently, then averaging across all classes:

$$\text{mIoU} = \frac{1}{C} \sum_{i=1}^{C} \frac{TP_i}{TP_i + FP_i + FN_i} \tag{2.24}$$

where C is the total number of semantic classes. This metric ensures that performance is evaluated across all categories, balancing the performance on both major and minor classes.

Voxel-based metrics provide a straightforward and simple way to measure the performance of 3D occupancy prediction by comparing voxel-by-voxel predictions with ground truth. However, these metrics consider voxels independently, without taking into account their relative positions along the sensor's view direction [54]. This makes the metric susceptible to exploitation, particularly in methods trained on 3D annotations with occlusion visible masks or weak 2D supervision where it must perform some form of semantic scene completion. For example, voxel predictions that deviate by a single voxel grid from the ground truth can result in disproportionate penalties or, conversely, unintentional performance gains. Additionally, models can predict thicker volumes behind visible structures, effectively increasing the number of predicted voxels of the same class, often boosting performance in the current voxel-based metric.

To address these challenges, the Ray-level Intersection-over-Union (RayIoU) metric is introduced [54]. RayIoU evaluates 3D occupancy predictions by simulating LiDAR behavior through ray casting. Rather than directly comparing voxels, RayIoU projects query rays into the 3D occupancy volume from the ego vehicle

center and retrieves the depth (distance from ego vehicle to the voxel grid) and semantic class label around the vicinity of the first occupied voxel each ray intersects. This is done for both the prediction and ground truth voxel grid.

Similar to mIoU, the formula for RayIoU is defined as:

$$\text{RayIoU} = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FP_i + FN_i} \tag{2.25}$$

where:

- C is the total number of semantic classes.
- TP_i is the **number of rays** correctly predicting the class label for class c within an acceptable L1 error threshold between predicted ray depth and actual ray depth.
- FP_i is the number of rays that predict class c when the ground truth is another class.
- FN_i is the number of rays that predict another class when the ground truth is c.

RayIoU provides a more reliable and meaningful evaluation of 3D semantic occupancy predictions. It mitigates inconsistencies by focusing solely on the first intersection along a ray, ensuring that minor misalignment in depth do not disproportionately affect the score. Additionally, RayIoU prevents performance inflation by ignoring predictions beyond the first surface hit, countering the thick surface issue. Finally, the metric also casts rays from different spatial and temporal positions, thereby still able to evaluate the model's ability to infer occupancy of previously unobserved or occluded areas. Overall, RayIoU offers a more robust

and accurate evaluation metric specifically tailored for 3D semantic occupancy prediction in dynamic environments like autonomous driving.

2.8 Semantic Occupancy Datasets

The landscape of AV datasets has evolved significantly over the past decade, with each dataset offering unique contributions to the field. These datasets typically comprise of carefully selected on-road driving scenes, varying in duration from tens of minutes to hundreds of hours, each capturing diverse environmental conditions and scenarios. The comprehensiveness of these datasets can be evaluated through several key metrics, including the number of point cloud scans, image frames and annotated semantic classes.

To support semantic occupancy prediction, numerous research groups have undertaken initiatives to augment existing meta AV datasets with dense 3D annotations. For instance, SSCBench [62] incorporates data from KITTI-360 [63], nuScenes [1], and Waymo [64], focusing specifically on camera inputs. SurroundOcc [51], OpenOccupancy [65], and Occ3D [66] builds upon the nuScenes and Waymo dataset. Notably, Occ3D has expanded the scope of 3D occupancy labels substantially, providing approximately 35,000 annotated frames based on nuScenes and 200,000 frames from Waymo data. These datasets represent a careful balance between scale and annotation quality, making them particularly valuable for developing and evaluating new approaches in 3D semantic occupancy prediction.

2.8.1 nuScenes

nuScenes dataset [1] encompasses 1000 driving scenes from two cities (Boston and Singapore), with 850 scenes allocated for training and validation, and 150 for

Table 2.1: Summary of Modern Autonomous Driving Datasets and their 3D Occupancy Extensions. *Classes within brackets are the semantic classes available in Occ. Ext.

Dataset	Year	Scans	Image	Classes*	Occ. Ext.
Waymo [64]	2020	230k	12M	23 (16)	Occ3D-Waymo [66]
Nuscenes [1]	2019	390k	1.4M	32 (17)	Multiple [62, 65, 66]
Argoverse [67]	2022	150k	2.7M	30 (17)	OCFBench [68]
Lyft L5 [69]	2019	46k	240k	9 (2)	Cam4DOcc [70]
SemanticKITTI [71]	2019	43k	15k	28 (19)	SemanticKITTI [71]
Kitti-360 [63]	2022	320k	80k	19 (19)	SSC-beanch [62]

testing. Each of these scenes is captured over a 20-second duration, ensuring a comprehensive range of scenarios, including complex urban intersections, highway driving, pedestrian interactions, and various weather conditions. nuScenes provides a comprehensive sensor suite, comprising six cameras, one LIDAR, and five radars. Notably, the camera images feature a resolution of 1600 x 900 pixels and forms a 360 degree surround view with six cameras. In terms of annotations, it contains 2D bounding boxes for cameras and 3D bounding boxes and semantic segmentation labels for LiDARs.

3D semantic occupancy labels extension of the nuScenes dataset is recently published on 2023 CVPR 3D Occupancy Prediction Challenge [66]. The dataset is manually annotated based off nuScene's LIDAR semantic segmentation by accumulating multiple frames over time whilst cleaning temporal artifacts, forming dense 3D semantic annotation for each frame. The voxel grid is set at 200 x 200 x 16 size at a resolution of 0.4meters for 1 grid length, surrounding a 80m x 80m x 6m (width, length, and height respectively) zone around the ego vehicle, Each voxel grid can hold one of 17 semantic labels where 0 - 16 represents semantic class labels similar to nuScenes-lidarseg annotation [1] and 17 represents the *free* occupancy class. On top of that, it also adds explicit occlusion masks for each voxel grid both in terms of camera visibility and LiDAR visibility. In total, this

adds up to 28,130 training frames and 6,019 validation frames with LiDAR scans, semantic segmentation, and voxelized semantic occupancy labels.

2.9 Seminal Methods

Table 2.2 presents pioneering 3D and BEV semantic occupancy prediction models, highlighting their methods, sensor inputs, datasets, and performance. Approaches like MonoScene [72], LSS [32] and BEVFusion [73] first introduced 2D-to-3D view transformation. BEVFormer [22] and TPVFormer [23] advanced multi-modal fusion and attention mechanisms. OpenOccupancy [65] and Occ3D [66] set new standards with dense 3D labels. FB-OCC [74], VAMPIRE [75], FastOcc [48] focused on individual challenges within 3D semantic occupancy prediction and achieved significant breakthroughs.

Table 2.2: List of seminal 3D / BEV semantic occupancy prediction models. In **Method**, VC stands for vision-centric, 2D-to-3D stands for vision-centric view transformation, Fusion stands for spatial or sensor fusion. In **Sensors**, C stands for camera, L stands for LIDAR. **Details** provide important model specifications.

Name	**Method**	**Sensors**	**Details**	**Dataset**	**Metrics (mIoU %)**
MonoScene [72]	VC → 2D-to-3D → 3D Decoder	C	First semantic occupancy paper; Monocular; Proposed backward-like projection	SemKITTI	SemKITTI: 11.08
LSS [32]	VC → 2D-to-3D → Fusion → Flatten → 2D BEV Decoder	C	Pioneered backward projection; Popularized multi-view BEV mapping	nuScenes	Car: 32.06, Drivable: 72.94
BEVFusion [73]	Multi → 2D-to-3D → Fusion → Flatten → 2D BEV Decoder	C + L	Pioneered forward projection; Popularized multi-modal inputs	nuScenes	Lane: 53.7, Drivable: 85.5
TPVFormer [23]	VC → 2D-to-3D → Fusion → Tri-Perspective → MLP Decoder	C	Weak supervision via LiDAR semantic segmentation; Cross attention applied	SemKITTI / nuScenes	SemKITTI: 11.26 / Occ3D-nus: 27.83
OpenOccupancy [65]	Multi → 2D-to-3D → Fusion → 3D Decoder	C + L	First dense 3D labels benchmark for occupancy	nuScenes	Occ3D-nus: 38.69
Occ3D [66]	Multi → 2D-to-3D → Fusion → 3D Decoder	C + L	Most complete dense 3D label; Added occlusion annotations	nuScenes	Occ3D-nus: 28.53
FB-OCC [74]	VC → 2D-to-3D → Fusion → 3D Decoder	C	Combines forward and backward projection; Achieved highest mIoU	nuScenes	Occ3D-nus: 54.19
VAMPIRE [75]	VC → 2D-to-3D → Fusion → Diff. Render → 2D Decoder	C	Weak supervision using 2D and BEV labels; Focus on simple modules	nuScenes	Occ3D-nus: 28.33
FastOcc [48]	VC → 2D-to-3D → Fusion → Flatten → 2D Decoder	C	Optimized for real-time deployment and efficiency	nuScenes	Occ3D-nus: 40.75

2.10 Deploying Semantic Occupancy Models

Efficient deployment of 3D semantic occupancy prediction models is essential for real-time autonomous driving applications [2]. Achieving real-time performance on embedded on-board devices requires lightweight architectures and efficient data processing pipelines, as these models must handle large-scale 3D voxel grids and multi-view inputs under strict latency constraints. Optimizing deployment efficiency involves balancing computational cost, model size, and accuracy.

One strategy is to optimize the main 3D backbone with sparse convolution networks [43, 54], which exploit the spatial sparsity inherent in 3D outdoor scenes. Unlike dense 3D convolutions, sparse convolutions process only non-empty voxels, significantly reducing computational overhead while preserving scene details. This approach is particularly effective in autonomous driving scenarios where large portions of the environment, such as the sky or distant regions, are empty. Sparse convolutional backbones, such as those implemented with the Minkowski Engine [40], enable efficient feature extraction and scene completion without sacrificing accuracy.

Beyond the backbone, other components of the 3D occupancy prediction pipeline can be optimized for efficiency. View transformation modules can adopt lightweight strategies by combining depth-aware projections with LiDAR-guided supervision [34]. 2D BEV decoders, as demonstrated by FastOcc [48], collapse 3D features into efficient 2D representations, using fast interpolation to recover height information with minimal overhead. Additionally, employing lightweight image backbones (e.g., ResNet-18 [9] or EfficientNetV2 [76]) further reduce overhead computation.

Hardware-aware optimizations also play a crucial role. Leveraging inference frameworks like TensorRT [48] for model quantization and mixed-precision compu-

tation can significantly reduce latency on embedded devices. Pruning redundant network layers and adopting knowledge distillation techniques can also further compress model size without compromising performance.

Overall, deployment efficiency in 3D semantic occupancy prediction requires a multi-faceted approach, combining algorithmic innovations such as sparse convolutions with hardware-aware optimizations.

2.11 Generalized Semantic Occupancy Models

3D semantic occupancy prediction's scalability is hindered by the high cost of dense 3D annotations. For instance, OpenOccupancy [65] requires approximately 4000 hours of manual labeling for just 34,000 frames [2]. This difficulty arises from the need to manually label occluded regions with limited sensor data, account for dynamic object movements that create spatial-temporal "tubes," and ensure temporal consistency of semantic classes across frames [62]. As such, this motivates research into weak supervision techniques.

Weak supervision methods leverage sensor data from calibrated cameras and LiDAR to reduce dependency on fully labeled datasets. Calibrated camera and LiDAR sensors offer complementary information: cameras provide 2D semantic details, while LiDAR supplies accurate 3D structures, both in a time-synchronized manner. Using these multi-modal signals, models can potentially learn 3D representations efficiently without full 3D labels. A critical enabler for this is differentiable rendering, which bridges 2D supervision with 3D learning [77].

2.11.1 Differentiable Rendering

Differentiable rendering enables end-to-end learning of 3D scene representations from 2D observations by making the 3D-to-2D projection process differentiable. This allows models to propagate gradients from 2D losses back to 3D features, facilitating weakly supervised learning from 2D semantic maps or depth images [78].

The differentiable rendering pipeline typically includes:

- **3D Scene Representation**: Learnable structures such as meshes, voxel grids, neural implicit fields, point clouds or 3D Gaussians.
- **Projection to 2D**: Differentiable operations, including volume rendering or point-based splatting, that simulate the camera's view.
- **Loss Computation**: Comparing rendered 2D outputs with ground-truth semantic or depth maps to generate gradients for backpropagation.

Integrating differentiable rendering with weak supervision from cameras and LiDAR has the potential to significantly reduce reliance on fully annotated 3D datasets. Indeed, many weakly supervised semantic occupancy models has integrated some form of differentiable rendering in their approach [46, 79–82].

2.12 Summary

To summarize, semantic occupancy perception represents a framework for analyzing three-dimensional scenes by discretizing continuous space into a voxel grid. The method assigns each voxel both occupancy status and semantic class information to enable comprehensive scene understanding.

The methodology consists of four primary components. The first component extracts features from sensor inputs such as cameras, LiDARs or a combination of both.

The second component transforms and processes these features into three-dimensional space. Camera-based methods perform view transformation to derive 3D information from 2D images. LiDAR-based approaches focus on converting sparse point measurements into dense occupancy representations through scene completion algorithms.

The third component performs feature fusion while maintaining spatial consistency. This process may combine data from multiple cameras, view points, or sensor types. Implementation methods include direct feature aggregation such as summation, averaging and concatenation, as well as attention-based mechanisms.

The fourth component implements an occupancy prediction head that determines both occupancy state and semantic classification for each voxel utilizing various architectural designs. The training process optimizes both occupancy detection and semantic classification objectives. Additionally, advancements in AV dataset and semantic occupancy research provide essential data and benchmarks for training and evaluating semantic occupancy prediction models.

In terms of optimizing semantic occupancy model's deployment capabilities, it requires balancing computational cost, model size, and accuracy. Methods such as lightweight architectures, streamlined pipelines, and hardware-aware strategies like compression and quantization can potentially be directions for achieving real-time performance.

Generalized semantic occupancy prediction seeks to reduce or eliminate reliance on annotated AV datasets, enabling models to handle diverse driving environments

and object classes. This approach leverages sensor data from calibrated cameras and LiDAR. Specifically, differentiable rendering facilitates the use of 2D camera data by bridging 3D predictions with 2D supervision, enabling gradient-based learning from signals such as 2D semantic maps or depth images. Overall, adopting weak supervision techniques not only reduces annotation costs but also enhances model generalization across diverse AV conditions, driving progress toward scalable and adaptable 3D semantic occupancy prediction models.

Chapter 3

Real-time 3D semantic occupancy prediction for autonomous vehicles using memory-efficient sparse convolution

As part of this integrated thesis, we now present our first paper on this work, which is published in the Proceedings of the 2024 IEEE International Conference on Intelligent Vehicles. The paper is followed by a brief discussion.

Real-time 3D semantic occupancy prediction for autonomous vehicles using memory-efficient sparse convolution

Samuel Sze and Lars Kunze[1]

***Abstract*— In autonomous vehicles, understanding the surrounding 3D environment of the ego vehicle in real-time is essential. A compact way to represent scenes while encoding geometric distances and semantic object information is via 3D semantic occupancy maps. State of the art 3D mapping methods leverage transformers with cross-attention mechanisms to elevate 2D vision-centric camera features into the 3D domain. However, these methods encounter significant challenges in real-time applications due to their high computational demands during inference. This limitation is particularly problematic in autonomous vehicles, where GPU resources must be shared with other tasks such as localization and planning. In this paper, we introduce an approach that extracts features from front-view 2D camera images and LiDAR scans, then employs a sparse convolution network (Minkowski Engine), for 3D semantic occupancy prediction. Given that outdoor scenes in autonomous driving scenarios are inherently sparse, the utilization of sparse convolution is particularly apt. By jointly solving the problems of 3D scene completion of sparse scenes and 3D semantic segmentation, we provide a more efficient learning framework suitable for real-time applications in autonomous vehicles. We also demonstrate competitive accuracy on the nuScenes dataset.**

I. INTRODUCTION

The rise of Autonomous Vehicles (AV) heralds a transformative era in transportation, moving us to potentially more efficient and safer alternatives. At the heart of these autonomous systems is a software stack composed of perception, localization, planning, and control. Among these, perception is crucial as it enables the vehicle to interpret and understand its surroundings, serving as the foundation for all subsequent vehicle decision-making. Typically, AV's deploy a suite of sensors such as cameras and LiDARs to perceive their surroundings. Cameras operate in 2D perspective image space, capturing essential color and texture details but lacking 3D spatial information. In contrast, sensors like LiDAR inherently provide 3D coordinates, albeit often with lower spatial resolution. For a holistic 3D scene understanding, both pixel-level information from cameras and sparse spatial information from LiDARs should be leveraged.

Traditional scene understanding uses 3D object detection methods that take in sensor information from cameras and LiDARs and project 3D bounding boxes across objects of interest such as vehicles, pedestrians, and traffic lights. However, while 3D object detection provides information about object instances, it often fails to capture complex object geometries and background information such as drivable space and buildings [1], [2]. An alternative approach to scene understanding is constructing 3D semantic occupancy maps, which partition the environment into a structured grid map with a predefined resolution, allowing each grid cell to be assigned a semantic label.

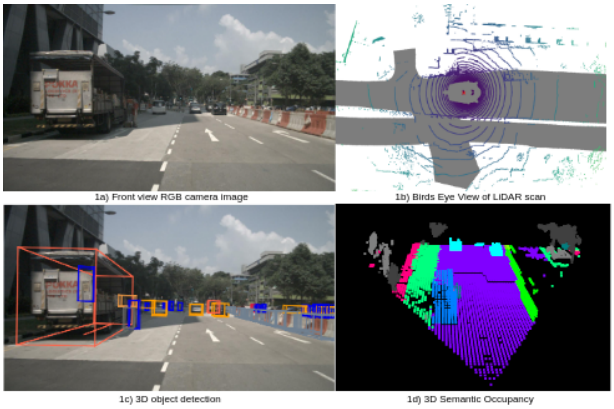


Fig. 1. Scene understanding methods using camera and LiDAR in nuScenes. Top two images shows perceived sensor output. Bottom two images shows scene predictions made from sensor output.

Nevertheless, 3D semantic occupancy prediction of outdoor scenes proves to be difficult due to the scenes' sparsity. Large areas such as the sky and occluded space captured by the camera often contain no relevant information. Most autonomous vehicles use rotary beam LiDARs which are inherently sparse, leading to overall sparse LiDAR data. To accurately interpret and fill in the missing parts of the environment, these algorithms must incorporate elements of 3D scene completion in addition to 3D semantic segmentation.

In this work, we focus on applying the existing sparse 3D convolution engine, Minkowski Engine [3], to a fused sensor representation of 2D camera and 3D LiDAR data for 3D semantic occupancy prediction. While the Minkowski Engine itself is a mature method, our key contribution lies in the innovative application of this sparse convolution framework to the problem of 3D semantic occupancy prediction. The fused sensor data enables sparse convolution to effectively handle outdoor scenes, allowing for accurate scene completion and semantic segmentation. Thereby, our main contributions are listed as the following:

- We propose the design of a novel sparse 3D convolution (Minkowski Engine [3]) model to perform 3D semantic occupancy prediction that jointly solves the problem of scene completion and semantic segmentation.
- We evaluate the model's 3D scene completion and semantic segmentation performance, achieving compet-

[1]Samuel Sze and Lars Kunze is with the Cognitive Robotics Group, Oxford Robotics Institute, Department of Engineering Science, University of Oxford: (samuels,lars)@robots.ox.ac.uk

itive accuracy against other algorithms on nuScenes dataset [1].
- We conduct time and memory usage evaluations to ensure model's real-time inference capabilities are close to human perception rates of 20 - 30 frames per second (FPS).

II. Related Work

A. Camera View Transformation

Camera view transformation techniques can be broadly bifurcated into two categories: extending 2D camera representations into a 3D domain or initializing a 3D domain and distilling 2D information onto it. The former aims to "lift" the camera image from perspective view to a 3D view, whereas the latter aims to "pull" camera information onto a predefined 3D space. Past lifting techniques includes probabilistic depth predictions for each camera pixel [4]–[8], as well as Multi-layer Perceptron (MLP) methods [9], [10]. A current popular pulling technique is using cross-attention based transformers where queries relayed from a discretized 3D space draws out keys and queries from a 2D perspective image [11]–[15].

Simple-BEV [16] suggests that the accuracy of Bird's Eye View (BEV) semantic prediction and similarly, 3D semantic occupancy prediction, often hinges more on factors like batch size, image resolution, and sensor fusion rather than sophisticated view transformation algorithms. Moreover, it is important to acknowledge that transforming views from 2D to 3D is inherently an ill-posed problem due to the absence of depth information in 2D images. This limitation highlights the necessity of employing deterministic methods that rely on additional sensors such as LiDARs. Arguing against the use of tools like LiDAR due to their cost overlooks their essential role in providing a fast and accurate method for view transformation.

B. 3D Semantic Scene Completion

Developments in 3D semantic scene completion of outdoor self-driving scenes is mainly spurred by well-curated datasets. With Semantic-KITTI [17], many LiDAR-based semantic scene completion methods appeared [18]–[20]. Given LiDAR as the only input, many of these methods do not require view transformation, simplifying the problem. However, most mainstream methods are built upon indoor LiDAR dataset such as NYUv2 [21] and ScanNet [22], and do not translate well to outdoor scenarios due to the sparsity of LiDAR points. Specific methods built upon Semantic-KITTI employed innovative solutions to densify the 3D scene. Notably, S3CNet [18] also uses Sparse convolution to balance computational budget while achieving state-of-the-art results. Nevertheless, LiDAR data lacks color and texture information, limiting its ability to discern objects with similar shapes but different visual features, such as distinguishing a street pole from a traffic light. Hence, LiDAR-only methods still require some level of feature engineering such as spherical projection to a range image, creating truncated signed distance function (TSDF) volumes, and estimating surface normal. These feature engineering inevitably requires additional tuning while introducing some level of noise.

More recently, multi-modal datasets such as nuScenes [23] and its extension, Occ3D-nuScenes [1], have enabled the development of vision-centric 3D semantic scene completion, primarily using cameras as the model input. Vision-centric methods, initially explored by Tesla [24] and further researched in various research studies [11], [25]–[27], employs cross-attention to geometrically relate 2D camera features into a 3D voxel grid, which is then used for BEV predictions or 3D semantic occupancy predictions. Efforts like deformable attention [14] and coarse-to-fine strategies [1] have been implemented to reduce the computational burden. Yet, a significant downside remains in the inherent dimensionality of establishing a dense 3D voxel grid, where attention mechanisms and transformers struggle to manage efficiently. Additionally, relying solely on camera data requires the model to simultaneously address the task of identifying and denoising distorted occluded areas on top of scene completion and semantic segmentation, posing a complex and challenging training task.

C. Sparse Convolution

For 3D and higher dimension data, it is often inefficient to parse such data through traditional dense 3D convolutions as most grid cells are empty to begin with. In the context of 3D semantic occupancy prediction, sparse convolution operates on spatially sparse 3D data. It only considers 3D points that are specified, often ones which contains LiDAR or camera information, whilst discarding meaningless empty 3D points. Sparse Convolution Networks, like Minkowski Engine [3], have been effectively utilized in indoor scenes [28] and for single object analysis [29]. However, their application in outdoor driving scenes, particularly with the availability of multi-modal outdoor datasets, remains an area for exploration. Given their proficiency in selectively processing meaningful 3D points, these networks are well-suited for the large and sparse areas typical of LiDAR and camera data capturing outdoor environments. Adapting sparse convolution networks to outdoor 3D semantic occupancy prediction presents a promising avenue for addressing the computational challenges currently faced in this field.

III. Problem Formulation

Formally, we define the 3D semantic occupancy prediction within the context of one front-view camera positioned to view directly ahead of an autonomous vehicle. Given a 2D camera image of dimensions $W \times H$, where each pixel i_{ij} contains RGB values (R, G, B) , the objective is to use view transformation to project, and sparse convolution to densify and segment image features into a 3D voxel space V with dimensions $X \times Y \times Z$ in real-time. We define real-time to be at least 20 frames per second (20 FPS), with the goal of reaching 30 FPS to match human perception rate. Each voxel v_{xyz} in this 3D space should encapsulate semantic and spatial information based on camera and LiDAR. The

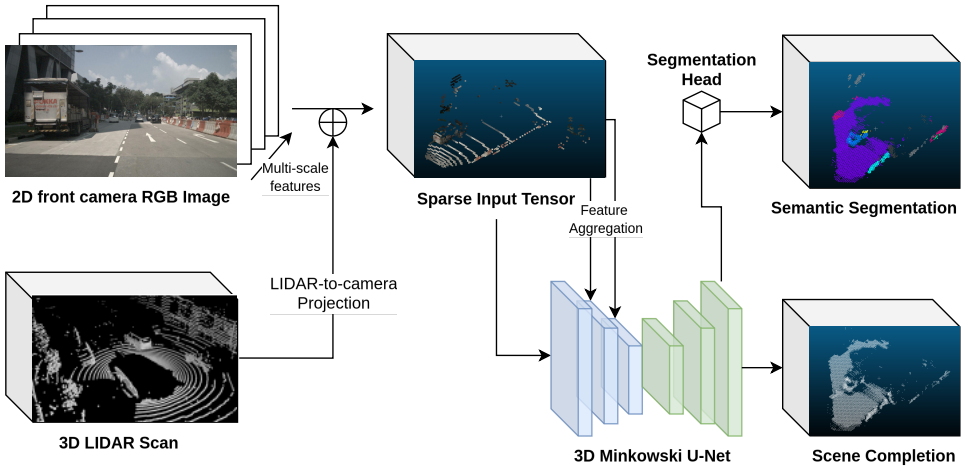


Fig. 2. Overview of System Pipeline

expected outcome is a labeled 3D voxel grid, where each voxel is affiliated with one of L class labels, denoted as $L = \{l_1, l_2, ..., l_C\}$, with C representing the aggregate number of distinct classes.

IV. Method

We utilize the Occ3D-nuScenes dataset [1], an extension of the nuScenes dataset, which is designed for 3D semantic occupancy prediction and benchmarking. It comprises of 28,130 Front camera RGB training images and 6019 validation images, as well as their corresponding voxelized ground truth semantic occupancy grid. The input LIDAR scans are obtained from the original nuScenes [23] dataset.

A. System Pipeline

The system pipeline is shown in Figure 2. Using a calibrated extrinsic and intrinsic matrix, we first project LIDAR points onto the RGB image space. Each projected point is fused with RGB color information through bilinear interpolation. Subsequently, the fused data is unprojected back into the 3D space. We also perform feature extraction using EfficientNetV2 [30] at layer 3,4 and 6 to extract higher level 2D camera features and lift them to the 3D space using the same projection method but scaled accordingly to accommodate for a reduction in spatial dimensions. The deterministic nature of the LiDAR-to-camera projection, assuming the sensors are adeptly calibrated, offers an immediate and precise depth-location correspondence - a key attribute for real-time tasks in autonomous vehicles.

Initial representation now has attributes of spatial coordinates (x, y, z) and feature values $([feats] + LiDARintensity)$. Spatial coordinates is voxelized by a resolution of 0.4m, discretizing the pointcloud into a 3D tensor, T_{sparse}. T_{sparse} has dimensions of 200 x 200 x 16 grid units in tensor shape, covering a physical boundary of [-40m, -40m, -1m] to [40m, 40m, 5.4m]. T_{sparse} is later converted to Coordinate List format (COO) with only non-empty grids included during sparse convolution operations. We leverage the Minkowski Engine generative completion algorithm [31] to densify the sparse tensor: $T_{\text{dense}} = \mathcal{M}(T_{\text{sparse}})$, where $\mathcal{M}$ represents the generative scene completion operation. Scene completion is guided by Occ3D-nuScenes [1] dataset ground truth labels with the grid resolution to match T_{sparse}'s dimension.

We carry out semantic segmentation on T_{dense}, directly supervised by ground truth semantic labels. Both the semantic segmentation and scene completion step is trained as a multi-task problem, with the semantic segmentation having an additional network of smaller size, predicting outputs over 18 semantic categories. The final 3D semantic occupancy prediction has 200 x 200 x 16 voxel grid with 0.4m resolution.

B. Minkowski Engine

Minkowski Engine processes sparse, high-dimensional data by focusing on non-zero data points and allowing adaptability in the kernel's shape and size. In other words, it follows a generalized convolution [3]. First, a sparse tensor is defined as a combination of a hash-table of coordinates and their corresponding features. Specifically, this is denoted as $u = [C_{n\times d}, x_{n\times m}]$, where $C_{n\times d}$ represents the coordinates in COO format and $x_{n\times m}$ corresponds to the feature vectors. The convolution operation within this framework is given by:

$$x^u_{\text{out}} = \sum_{i \in \mathcal{N}^D(u, C_{\text{in}})} W_i \cdot x^{u+i}_{\text{in}} \quad \text{for} \quad u \in C_{\text{out}} \tag{1}$$

In equation 1, x^u_{out} is the output feature at point u. $\mathcal{N}^D(u, C_{\text{in}})$ denotes the set of offsets defining the neighborhood of input coordinates around u that plays a part in the sparse convolution layer. W_i are the learnable weights of the convolution kernel, and x^{u+i}_{in} is the input feature at the position offset by i from u.

C. Network Architecture

With reference to figure 3, the proposed 3D semantic occupancy prediction model utilizes 2 U-Net-like [32] like encoder-decoder networks sequentially. Both U-Net shares similar structure, with the scene completion U-Net having

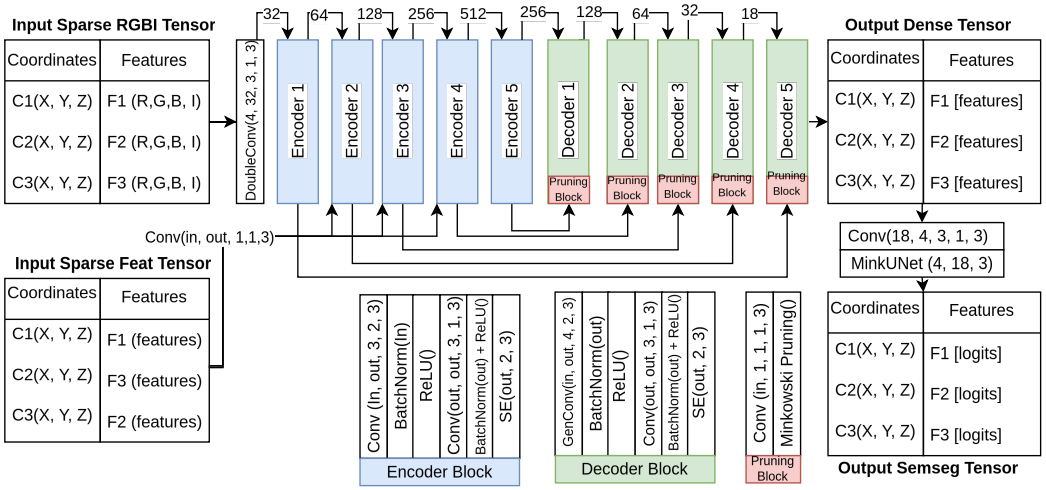


Fig. 3. Minkowski Engine Sparse Convolution Network Architecture. Scene Completion U-Net is shown in detail. MinkUNet is the Semantic Segmentation U-Net.

deeper layers and specialized modules such as Squeeze and Excite [33], Generative Convolution Transpose [31] and Sparse Pruning. An encoder block contains a double convolution Layer and a squeeze and excite (SE) layer. The convolution layer features batch normalization (BN) and ReLU activation to standardize the variable density of sparse data. The SE layer dynamically recalibrates channel-wise features through global average pooling, enhancing the model's focus on key features across the input sparse data. For certain encoder blocks, image features extracted from EfficientNetV2 [30] are also concatenated before further downsampling.

The decoder module for scene completion is designed for generative tasks. The process begins with generative transposed convolution capable of generating new coordinates with the outer product of the weight kernel and the input coordinates. A SE layer is attached after generation. At each decoder layer, a skip connection from the corresponding encoder is also added. Following that, a 3D binary classification convolution is applied to the sparse tensor which assigns a decision value to prune. The decision threshold is trainable with ground truth supervision. Redundant features are pruned using Minkowski Engine's built in pruning module. Overall, the decoder generates, upsamples, classifies, and then prunes voxels to iteratively create accurate 3D representation at each layer depth.

Upon reaching the final decoder layer, an additional semantic segmentation U-Net head is attached for the sole purpose of semantic segmentation of the densified sparse tensor. This U-Net features sparse convolution layers and skip connections, with bottleneck channel length of 256.

D. Loss Function

A multi-task loss function is defined to train the 3D Semantic Occupancy prediction model. The two terms are balanced by a Lambda (λ) constant empirically set to 0.5. Experiments conducted on different λ constants is represented in the ablation study.

$$\mathcal{L}_{\text{loss}} = \mathcal{L}_{\text{completion}} + \lambda \mathcal{L}_{\text{segmentation}} \tag{2}$$

$\mathcal{L}_{\text{completion}}$ is a voxel based binary cross entropy with logits loss ($\sum_{i,j,k} \text{BCE}(p_{ijk}, y_{ijk})$). It is computed over the volumetric occupancy on each decoder layer. This loss also tunes the decision threshold used for pruning generated voxels.

$$\mathcal{L}_{\text{segmentation}}(z, y) = -\frac{1-\beta}{1-\beta^{n_y}} \log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right) \tag{3}$$

In normal semantic segmentation settings, multi-class cross entropy loss is sufficient with z being predicted logits and y being the ground truth labels. However, in Occ3D-nuScenes dataset, the class imbalance is extreme. For example, *bicycle* and *motorcycle* class is 10^4 times lower in frequency than *drivable space* and *free* class [34]. Therefore, a class-balanced loss is applied [35], where an effective number β is used to reweight the cross entropy loss function, giving more weight to classes with fewer effective samples. Effective number is set to $\beta = 0.9$. Effective samples β^{n_y} is calculated by the normalized number of voxels representing each class over all ground truth voxels over the entire training dataset. Validity of the class balanced loss is evaluated in the ablation study.

V. Experiments

A. Experimental Setup

The training was configured with the Adam optimizer at a learning rate of 1e-4, and a Cosine-Annealing learning rate schedule. The model was trained till convergence, with a batch size of 10. The hardware setup for the training included 1 NVIDIA RTX 4090 GPU and 24GB DDR4 RAM. Training VRAM usage is at 4.3 GB.

Data augmentation is performed on the input and ground truth dataset, which includes image features and LiDAR intensity value perturbation to simulate sensor noise (noise

TABLE I

QUANTITATIVE RESULTS OF 16 SEMANTIC CLASSES ON OCC3D-NUSCENES VALIDATION DATASET.

	mIoU	Barrier	Bicycle	Bus	Car	Cons. Veh.	Motorcycle	Pedestrian	Traf. Cone	Trailer	Truck	Driv. Sur.	Other Flat	Sidewalk	Terrain	Manmade	Vegetation
MonoScene [25] (%)	6.06	7.23	4.26	4.93	9.38	5.67	3.98	3.01	5.90	4.45	7.17	14.91	6.32	7.92	7.43	1.01	7.65
OccFormer [36] (%)	21.93	30.29	12.32	34.40	39.17	14.44	16.45	17.22	9.27	13.90	26.36	50.99	30.96	34.66	22.73	6.76	6.97
BEVFormer [11] (%)	23.70	36.77	11.70	29.87	38.92	10.29	22.05	16.21	14.69	26.44	23.13	48.19	33.10	29.80	17.64	19.01	13.75
TPVFormer [27] (%)	27.83	38.90	13.67	40.78	45.90	17.34	19.99	18.85	14.30	26.69	34.17	55.65	35.47	37.55	30.70	19.40	16.78
CTF-Occ [1] (%)	28.53	39.33	20.56	38.29	42.24	16.93	24.52	22.72	21.05	22.98	31.11	53.33	33.84	37.98	33.23	20.79	18.0
NVOCC [37] (%)	54.19	57.98	46.40	52.36	63.07	35.68	48.81	42.98	41.75	60.82	49.56	87.29	58.29	65.93	63.30	64.28	53.76
Ours (%)	36.03	40.12	0.00	51.01	63.10	12.52	3.60	30.16	7.69	32.71	32.17	87.19	37.42	41.34	44.46	74.24	84.68

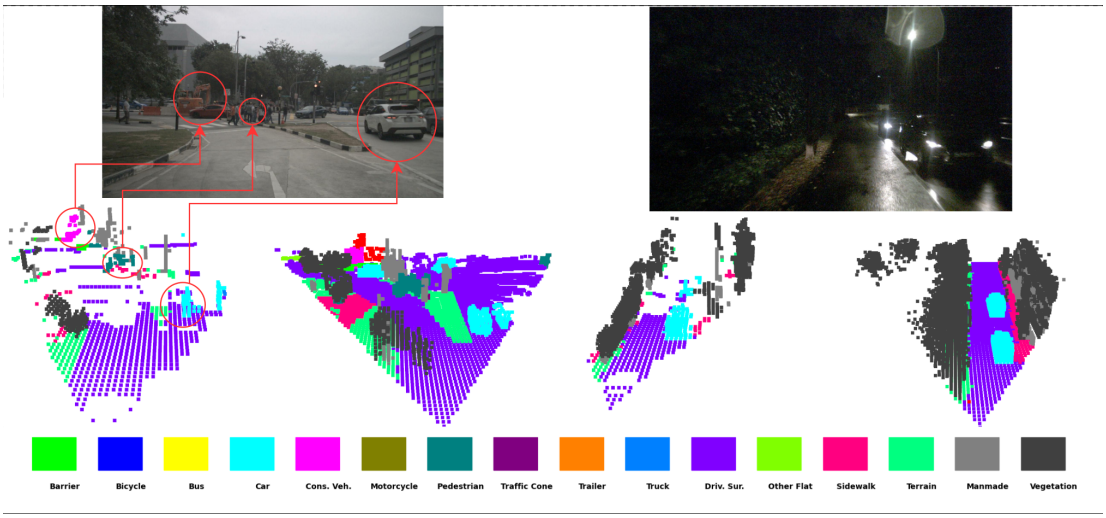


Fig. 4. Qualitative predictions on Occ3D-nuScenes validation dataset. Semantic occupancy grid is viewed in 3rd person perspective angle. Each group of images consists of occupancy prediction image on the left, ground truth label on the right, and front camera image on top.

strength of 5% on normalized values), random translation (-4 to 4 voxel grids) and ground truth voxel masking (10%).

B. Scene Completion

We evaluate the 3D scene completion performance using binary voxel completion intersect over union (IoU), precision, recall, and F1 score to be compared with other scene completion methods. Referring to Table II, P represents the set of voxels predicted by the model, GT the set of voxels in the dense occupancy ground truth.

TABLE II

EVALUATION METRICS FOR 3D SCENE COMPLETION TASK.

Metric	Equation	Better
IoU	$\frac{\lvert P \cap GT \rvert}{\lvert P \cup GT \rvert}$	Higher
Precision	$\frac{\lvert P \cap GT \rvert}{\lvert P \rvert}$	Higher
Recall	$\frac{\lvert P \cap GT \rvert}{\lvert GT \rvert}$	Higher
F1 Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$	Higher

We compare these metrics with other scene completion methods on the Occ3D-nuScenes validation dataset. In particular, TransformerFusion [38] and Atlas [39], which originally targets the task of 3D reconstruction of indoor scenes from RGB-D images. SurroundOcc [26], MonoScene [25] and CTF-Occ [1] focuses on the same outdoor driving scene completion task. Values from these methods are achieved through dense ground truth supervision for fair comparison, as reported by [26]. Referring to Table III, our proposed method achieves an scene completion IoU of 0.533, outperforming the current state-of-art by 16%. We also show a large improvement in precision, recall and f1, indicating we densify much more of the sparse input scene while being accurate to the ground truth.

TABLE III

SCENE COMPLETION EVALUATION ON OCC3D-NUSCENES VALIDATION SET.

Model	IoU	Precision	Recall	F1 Score
TransformerFusion [38]	-	0.375	0.591	0.453
Atlas [39]	-	0.407	0.546	0.458
SurroundOcc [26]	0.347	0.414	0.602	0.483
MonoScene [25]	0.342	-	-	-
CTF-Occ [1]	0.377	-	-	-
Ours	**0.533**	**0.775**	**0.632**	**0.690**

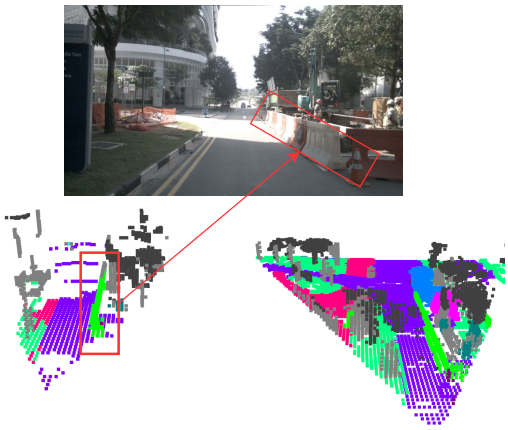


Fig. 5. This example shows the model's ability to predict dynamic obstacles such as temporary road barriers installed for construction on drivable surface.

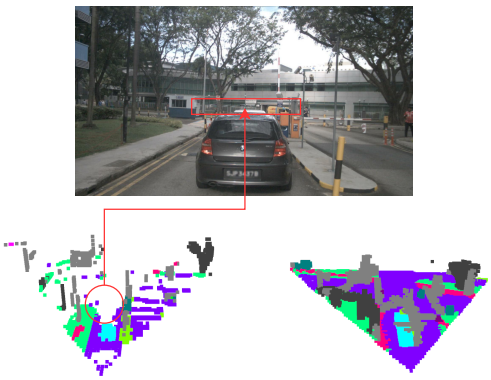


Fig. 6. This example shows the model's inability to hallucinate occluded areas caused by blocking cars. This is the expected performance as the model should not imagine areas without a proper occlusion handling method.

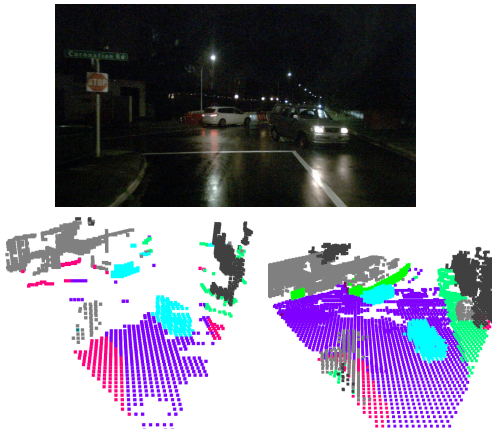


Fig. 7. This example shows a night time intersection with multiple vehicles. The model is able to detect both vehicles with their geometries. However, night time prediction of drivable surface is poor and further distances are empty.

C. Semantic Segmentation

In semantic segmentation, mIoU (mean Intersection over Union) is calculated using the following formula: $\text{mIoU} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i+FP_i+FN_i}$, where N represents the total number of semantic classes, TP_i denotes True Positives for class i, FP_i represents False Positives for class i, and FN_i corresponds to False Negatives for class i. We monitored the semantic segmentation performance through mean Intersection over Union (mIoU) and semantic IoU for each semantic class, and compare it with baseline BEVFormer [11], several popular architectures [1], [27], [36] and NVOCC [37], current state-of-the-art. We do not compare the *void* and *free* class. Referring to Table I, we observe a 8 - 10 % improvement in mIoU over BEVFormer and CTF-Occ, as well as improvements in most semantic class IoU. Compared to NVOCC, we are competitive in *car*, *drivable surface*, *manmade* and *vegetation*. However, *bicycle*, *motorcycle* and *traffic cones* presents a significant challenge to the model's prediction. These small objects not only appear a lot less in Occ3D-nuScene's dataset due to class imbalance, but also potentially smoothed in the voxelization and convolution processes, erasing or blending their features with its surroundings. Methods to mitigate this problem will be discussed in section VI.

Figure 4 shows qualitative inferences of the model. Left group of day-time images demonstrates the model's ability to classify scene objects of varying distance and shapes into distinct semantic classes. Right group of images shows model's fidelity in low light environments: it is able to discern dark colored vehicles at night, as well as predicting obstacles on the left side of the drivable surface. However, the model does show a decrease in maximum prediction distance. More qualitative results are shown in Figure 5, 6, 7.

D. Real-time Inference

Inference are performed on RTX 4090 with six multi-camera images for the other models at 1600x900 image resolution. Since our model focuses on monocular camera, we use a batch size of 6 for fair comparison. We compare the inference time and GPU memory usage in Table IV, showing a 6 - 10 times increase in inference time, translating to a potential frame rate of 30 FPS and an achievable frame rate of 20 FPS while utilizing only 1.2GB of GPU memory.

TABLE IV
MODEL TIME EFFICIENCY AND GPU MEMORY USAGE COMPARISON

Model	Inference Time (s)	Memory Usage (MB)
BEVformer [11]	0.31	4500
MonoScene [25]	0.87	20300
SurroundOcc [26]	0.34	5900
CTF-Occ [1]	0.38	18000
Ours	**0.03 - 0.05**	**1200**

E. Ablation Study

Referring to Table V, we conducted ablation studies by removing the class balanced cross entropy loss and use standard focal loss, removing the squeeze and excite layer, and removing the higher level feature input tensor. "Ours" refers to the current model with all three components included. The results are validated against Occ3D-nuScenes validation dataset. The semantic segmentation mIoU and completion IoU are reported. We observed that removing the class balance loss results in a severe drop in both completion IoU and mIoU. This validates the problem of class imbalance in Occ3D-nuScenes dataset, and class balance loss correctly addresses it by leading the model to be sensitive to all classes in both scene completion and semantic segmentation. The incorporation of Squeeze-and-Excite layers leads to slight improvements in both mIoU and completion IoU. This can be attributed to the use of global average pooling, which enhances the model's adaptability to varying outdoor scenes. Lastly, concatenating higher level image features with the initial $RGB + Intensity$ features achieved no significant performance improvements. This outcome could suggest that the initial feature set was likely sufficient for the task. It could also indicate the feature extractor, EfficientNetV2 [30], might have limited capacity to produce meaningful outdoor scene features.

TABLE V
ABLATION STUDY ON THREE CORE MODEL COMPONENTS.

Description	Semseg mIoU	Completion IoU
Ours	0.3603	**0.5329**
Ours w/o CBLoss	0.3240	0.3911
Ours w/o SELayer	0.3360	0.4983
Ours w/o features	**0.3622**	0.5110

Referring to table VI, using a lambda value of 0.5 yields the best results, with a minor yet notable improvement in Semantic Segmentation mIoU to 0.3555 and a more significant enhancement in Completion IoU to 0.5243.

TABLE VI
LAMBDA CONSTANT EFFECT ON 3D SEMANTIC OCCUPANCY PREDICTION.

λ	Semseg mIoU	Completion IoU
0.3	0.3534	0.5089
0.5	**0.3555**	**0.5243**
0.7	0.3466	0.5115

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a Minkowski engine-based sparse convolution model capable of semantic scene completion on diverse outdoor autonomous vehicle driving scenes using LiDAR and a front-view camera. In particular, we demonstrated that the model is able to perform 3D semantic occupancy prediction with significantly fewer computational resources compared to many current methods, achieving real-time inference while maintaining comparable accuracy. More broadly, the sparse convolution approach we propose for AVs offers a pathway to enhance the efficiency of real-time robotics perception.

To address the challenge of detecting and segmenting small objects, incorporating an object detection preprocessing step on the RGB images can be considered. By utilizing a pretrained lightweight object detection module, the 2D bounding boxes can be projected into the 3D space using the same camera-to-LiDAR transformation matrices, subsequently creating 3D ROI (Region of Interest) masks. These masks complement the input sparse tensor into the current network, providing guidance towards small objects during training and improving their segmentation accuracy.

Future work involves several aspects. Firstly, we aim to extend the current model to a multi-view camera setup for 360-degree occupancy prediction of the ego vehicle. Secondly, performing scene completion at distances beyond 20 meters proves challenging due to the sparsity of LiDAR points, specifically when using a 32-beam setup as in the nuScenes dataset [23]. Incorporating probabilistic techniques to fuse distant camera features and radars into the 3D space may help achieve better accuracy in these regions. Thirdly, employing self-supervision by creating pseudo-dense occupancy ground truth via monocular 2D camera-based depth estimation and semantic segmentation can help expand the training dataset, allowing the current model to become more robust in complex driving environments.

ACKNOWLEDGMENT

This project was supported by the EPSRC project RAILS (EP/W011344/1) and the Oxford Robotics Institute's research project RobotCycle.

REFERENCES

[1] X. Tian, T. Jiang, L. Yun, Y. Wang, Y. Wang, and H. Zhao, "Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving," *arXiv preprint arXiv:2304.14365*, 2023.

[2] X. Wang, Z. Zhu, W. Xu, Y. Zhang, Y. Wei, X. Chi, Y. Ye, D. Du, J. Lu, and X. Wang, "Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception," *arXiv preprint arXiv:2303.03991*, 2023.

[3] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.

[4] J. Philion and S. Fidler, "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*. Springer, 2020, pp. 194–210.

[5] C. Reading, A. Harakeh, J. Chae, and S. L. Waslander, "Categorical depth distribution network for monocular 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8555–8564.

[6] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. L. Rus, and S. Han, "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2774–2781.

[7] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall, "Fiery: Future instance prediction in bird's-eye view from surround monocular cameras," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 273–15 282.

[8] S. Schulter, M. Zhai, N. Jacobs, and M. Chandraker, "Learning to look around objects for top-view representations of outdoor scenes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 787–802.

[9] T. Roddick and R. Cipolla, "Predicting semantic map representations from images using pyramid occupancy networks," 2020.

[10] N. Hendy, C. Sloan, F. Tian, P. Duan, N. Charchut, Y. Xie, C. Wang, and J. Philbin, "Fishing net: Future inference of semantic heatmaps in grids," *arXiv preprint arXiv:2006.09917*, 2020.

[11] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *European conference on computer vision*. Springer, 2022, pp. 1–18.

[12] L. Chen, C. Sima, Y. Li, Z. Zheng, J. Xu, X. Geng, H. Li, C. He, J. Shi, Y. Qiao, *et al.*, "Persformer: 3d lane detection via perspective transformer and the openlane benchmark," in *European Conference on Computer Vision*. Springer, 2022, pp. 550–567.

[13] A. Saha, O. Mendez, C. Russell, and R. Bowden, "Translating images into maps," in *2022 International conference on robotics and automation (ICRA)*. IEEE, 2022, pp. 9200–9206.

[14] Y. Li, Z. Yu, C. Choy, C. Xiao, J. M. Alvarez, S. Fidler, C. Feng, and A. 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*, 2023, pp. 9087–9098.

[15] Z. Xia, Y. Liu, X. Li, X. Zhu, Y. Ma, Y. Li, Y. Hou, and Y. Qiao, "Scpnet: Semantic scene completion on point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 642–17 651.

[16] A. W. Harley, Z. Fang, J. Li, R. Ambrus, and K. Fragkiadaki, "Simple-bev: What really matters for multi-sensor bev perception?" in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2759–2765.

[17] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9297–9307.

[18] R. Cheng, C. Agia, Y. Ren, X. Li, and L. Bingbing, "S3cnet: A sparse semantic scene completion network for lidar point clouds," in *Conference on Robot Learning*. PMLR, 2021, pp. 2148–2161.

[19] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[20] S. Liu, Y. HU, Y. Zeng, Q. Tang, B. Jin, Y. Han, and X. Li, "See and think: Disentangling semantic scene completion," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[21] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[22] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[23] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[24] F. Richard, J. Schneider, D. Trines, and A. Wagner, "Tesla technical design report part i: Executive summary," *arXiv preprint hep-ph/0106314*, 2001.

[25] A.-Q. Cao and R. de Charette, "Monoscene: Monocular 3d semantic scene completion," 2022.

[26] Y. Wei, L. Zhao, W. Zheng, Z. Zhu, J. Zhou, and J. Lu, "Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving," 2023.

[27] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3d semantic occupancy prediction," 2023.

[28] S. Huang, M. Usvyatsov, and K. Schindler, "Indoor scene recognition in 3d," *IROS*, 2020.

[29] S. Alonso-Monsalve, L. H. Whitehead, A. Aurisano, and L. E. Sanchez, "Automated segmentation of computed tomography images with submanifold sparse convolutional networks," 2022.

[30] M. Tan and Q. V. Le, "Efficientnetv2: Smaller models and faster training," 2021.

[31] J. Gwak, C. B. Choy, and S. Savarese, "Generative sparse detection networks for 3d single-shot object detection," in *European conference on computer vision*, 2020.

[32] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[33] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," 2019.

[34] T. Vu, J.-H. Kim, M. Kim, S. Jung, and S.-G. Jeong, "Milo: Multi-task learning with localization ambiguity suppression for occupancy prediction," 2023.

[35] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," 2019.

[36] Y. Zhang, Z. Zhu, and D. Du, "Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction," 2023.

[37] Z. Li, Z. Yu, D. Austin, M. Fang, S. Lan, J. Kautz, and J. M. Alvarez, "FB-OCC: 3D occupancy prediction based on forward-backward view transformation," *arXiv:2307.01492*, 2023.

[38] A. Božič, P. Palafox, J. Thies, A. Dai, and M. Nießner, "Transformerfusion: Monocular rgb scene reconstruction using transformers," 2021.

[39] Z. Murez, T. van As, J. Bartolozzi, A. Sinha, V. Badrinarayanan, and A. Rabinovich, "Atlas: End-to-end 3d scene reconstruction from posed images," 2020.

3.1 Design Choice: Sparse Convolution

In 2D settings, such as images, data is often dense. Every pixel in an image typically contains some information such as color value. This justifies the application of standard convolution processes. However, when transitioning to the 3D domain, especially with data like point clouds or volumetric representations, the scenario differs. A vast majority of the voxels in a 3D volume might be empty or devoid of points. Applying convolutions on this largely inactive space would be computationally wasteful. For example, when LiDAR scans are transformed into a 3D voxel grid representation, the inherent sparsity becomes evident. This sparsity intensifies the need for specialized convolution methods, like sparse convolution, to process the data efficiently in autonomous driving scenarios without wasting computational resources on the large empty portions.

3.1.1 Minkowski Engine

Several architectures have emerged as pioneering implementations in this field [40–42]. Of which, Minkowski Engine's centers around designing efficient and differentiable sparse tensor, as well as dynamic coordinate generation. The engine's approach to use Coordinate List Format (COO) for sparse tensor representation facilitates both memory-efficient operation and computational flexibility. COO format are internally represented using a hash table structure that maps coordinates to features:

$$C = \begin{bmatrix} b_1 & x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots & \vdots \\ b_N & x_N & y_N & z_N \end{bmatrix}, \quad F = \begin{bmatrix} f_1^T \\ \vdots \\ f_N^T \end{bmatrix}, \tag{3.1}$$

where N corresponds to the number of non-empty voxels, b_i represents the

batch index, and f_i represents the feature vector associated with the i-th coordinate. A key strength of the Minkowski engine lies in its ability to automatically handle coordinate generation and management during network operations. When performing convolutions, the engine determines valid output coordinates by considering the input sparsity pattern and the convolution kernel. This process maintains sparsity while ensuring all necessary computations are performed.

During convolution, the neighborhood computation is defined as:

$$N^D(u, C_{\text{in}}) = \{i \mid u + i \in C_{\text{in}}, i \in N^D\} \tag{3.2}$$

For each output coordinate u, which can be thought of as a center point, the computation identifies contributing points in a principled way. N^D represents the shape of the convolution kernel — for instance, in a standard $3 \times 3 \times 3$ kernel in 3D space, it encompasses all possible offsets within one step in each direction, such as $(-1, -1, -1)$, $(-1, -1, 0)$, and so forth. The condition $u + i \in C_{\text{in}}$ ensures that offset from center point u by i only consider coordinates that contain actual input data. The selective approach enables Minkowski engine to perform convolutions efficiently by processing only locations with valid data while maintaining the mathematical integrity of the convolution operation.

Furthermore, Minkowski engine's architecture allows for direct adaptation of traditional 2D architectural designs into higher-dimensional networks. This capability stems from its generalized convolution implementation, which treats dimensionality as a parameter rather than a fixed constraint. By maintaining sparsity throughout the network operations and implementing efficient sparse matrix operations, the engine can process high-resolution inputs that would be otherwise be computationally prohibitive with dense representations. This capability is par-

ticularly valuable when processing large-scale 3D scenes or high-dimensional data where efficient memory utilization becomes crucial for practical applications.

3.1.2 Generative Minkowski Engine Operations

Minkowski Engine extends its core functionality through specialized transposed convolutions that can systematically modify the sparsity pattern of tensors. This generative process is critical for tasks that require expanding beyond observed surface points, such as scene completion. The process operates by expanding the support region of sparse tensors through a mathematical formulation:

$$\text{support}(T) = C \otimes [-K, \ldots, K]^3, \tag{3.3}$$

where C represents the input sparsity pattern and K denotes the kernel size. This outer product operation effectively creates a denser sparsity pattern by expanding each input point into a cubic region based on the kernel size, following the mathematical formulation:

$$T'[x, y, z] = \sum_{i,j,k \in \mathcal{N}(x,y,z)} W[x - i, y - j, z - k] T[i, j, k] \quad \text{for } (x, y, z) \in C', \tag{3.4}$$

where $C' = C \otimes [-K, \ldots, K]^3$, and $\mathcal{N}(x, y, z)$ defines the neighborhood of contributing points. This formulation allows multiple regions to overlap with each other, creating a richer representation space. The process is particularly valuable in 3D detection tasks where object centers often lie in unobserved regions behind surface points.

After the expansion phase, the engine employs a pruning mechanism to maintain efficiency. This process uses a probability function $P_s(\cdot)$ to evaluate each

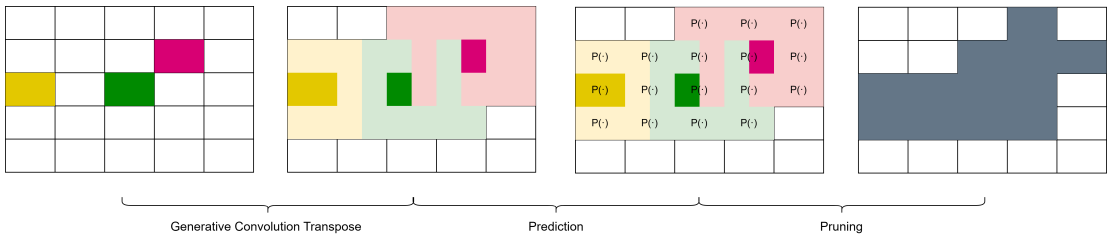


Figure 3.1: Illustration in 2D: Generative transposed convolution upsamples a low resolution sparse tensor into a high resolution sparse tensor. Then, some of the upsampled coordinates with sparsity predictions P(·) below threshold are pruned.

newly created voxel. Instead of relying on a fixed threshold, the pruning decision is determined by a learned binary classification threshold, which in our case compares the predicted occupancy against ground-truth voxel occupancy:

$$T' = \text{Prune}(T, \hat{b} = \text{Classify}(T; W_P)),$$

where $\hat{b}$ represents the binary classification output indicating whether a voxel is likely to contain meaningful information. This classification threshold is trained to minimize the discrepancy between predicted voxel occupancy and ground-truth data, ensuring that only voxels with high confidence in their occupancy status are retained.

3.2 Discussion

The aim of this work is to optimize 3D semantic occupancy prediction for real-time applications in autonomous vehicles. To achieve this, we propose an efficient sensor fusion technique that uses a simple LiDAR-to-camera forward projection to construct a sparse 3D voxel representation. By integrating front-view camera images and LiDAR scans in an early fusion fashion, our hope is to keep computation low at the cost of losing some higher level camera features.

A key contribution of our work is the introduction of a sparse convolution-

based 3D U-Net built on the Minkowski Engine. This design allows us to process sparse 3D data efficiently, handling many empty voxels typical in outdoor driving scenes. Unlike traditional dense 3D convolutions, our sparse convolution approach reduces unnecessary computation by focusing only on occupied voxels, which can drastically improves efficiency. Compared to other semantic occupancy methods, our model achieves up to a tenfold reduction in inference time, running at 20 to 30 frames per second while using only 1.2 GB of GPU memory. Despite the efficiency gains, we maintain competitive accuracy in scene completion and semantic segmentation on the nuScenes dataset. Notably, our approach improves occupancy scene completion IoU by 16 percent over prior state-of-the-art methods.

However, our approach is not without limitations. The early fusion of LiDAR and camera data, while efficient, results in the loss of high-level camera features that could otherwise enhance semantic segmentation. This limitation likely contributes to our model's difficulty in detecting small objects, such as bicycles and traffic cones, which is further compounded by class imbalance and voxelization artifacts that blur fine details. Nighttime and long-range predictions also suffer from performance degradation due to reduced camera visibility and sparser LiDAR returns. Additionally, we only conducted experiments using monocular front-view camera. Extending our approach to a multi-view camera setup is necessary for a comprehensive 360-degree understanding and fair comparison with many other semantic occupancy methods.

3.3 Immediate Improvements

Following the publication of the work presented in this chapter, we identified several limitations in our initial approach, particularly regarding the early fusion of

raw sensor data and the constraints of the current training pipeline. To address these deficits and lay the groundwork for our second contribution, we implemented a series of immediate improvements.

To streamline network training, improve dataloading, and facilitate open-source research in 3D semantic occupancy prediction, we migrated to MMDetection3D, part of the OpenMMLab framework [83]. It is an open-source toolbox optimized for multi-modality 3D detection. MMDetection3D provides natural integration with 2D detection methods, efficient training pipelines, and direct support for large-scale outdoor datasets such as nuScenes and Waymo, making it a robust platform for future developments.

Additionally, we refine our sensor fusion strategy by shifting the LiDAR-to-camera projection step to occur after the 2D feature extraction stage, similar to early fusion mentioned in figure 2.5. This modification allows the model to operate on high-level semantic features rather than raw RGB data. We employ EfficientViT-B1 [84], a lightweight Vision Transformer designed for real-time applications, to extract 2D camera features. Pretrained on the Cityscapes dataset [85], EfficientViT-B1 improves mIoU by 5% over the B0 variant while maintaining computational efficiency. Despite this additional processing step, the overall inference time remains around 25 FPS while utilizing 1.4GB of GPU memory.

3.4 Illustrative Results

Referring to figure 3.2, we selected challenging scenes to evaluate our model's robustness, including rainy and nighttime scenarios. Despite the increased difficulty, our method demonstrates resilience across various weather and lighting conditions. The final row provides a control comparison, showcasing results before and after

the improvements (refer to figure 5 from the paper for comparison). However, we observe that the current 3D decoder layers requires further refinement, as it occasionally results in over-pruning of flat regions such as *drivable surface* and *sidewalk*.

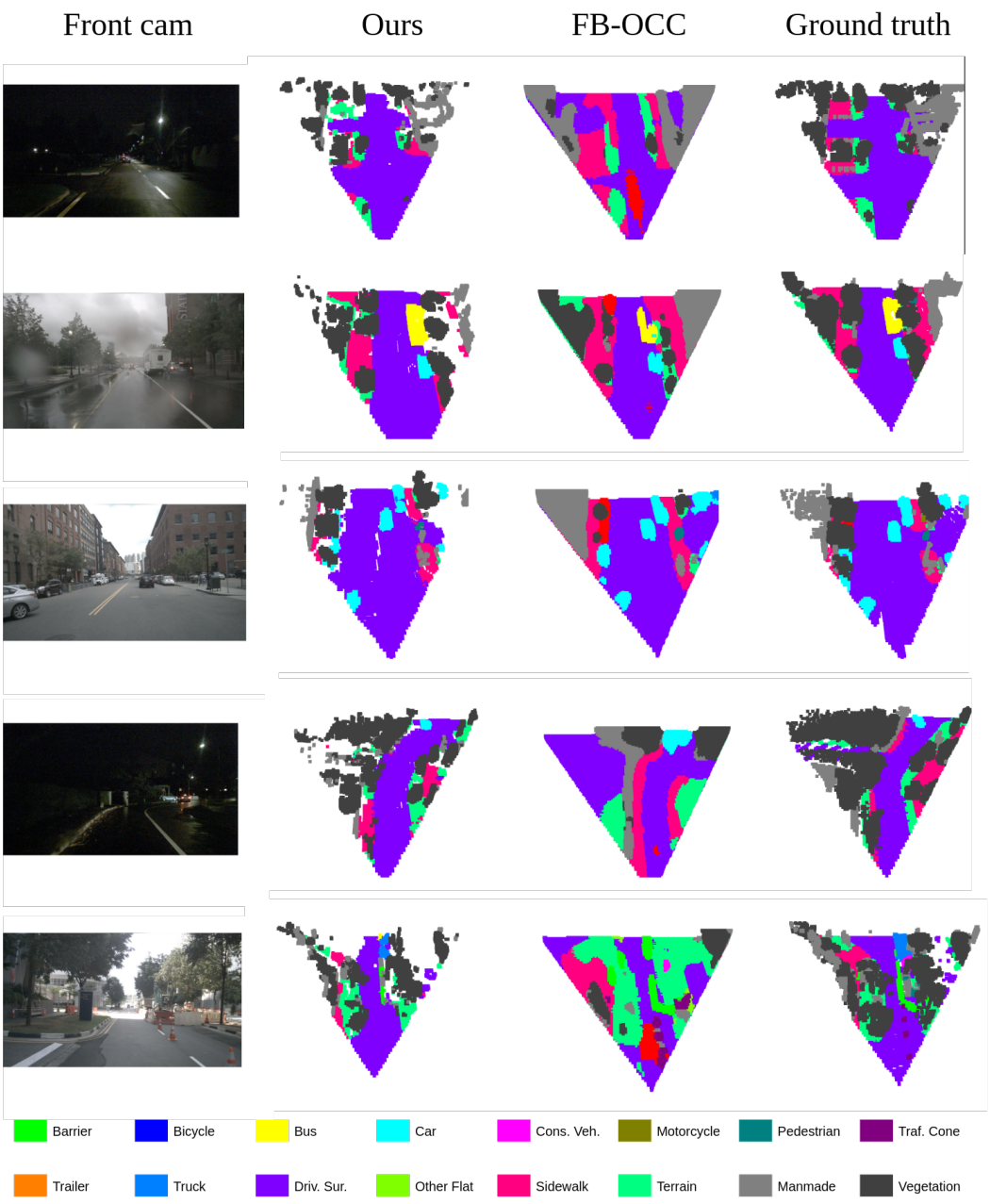


Figure 3.2: Additional qualitative results against Ground-truth and FB-OCC [74] on the Occ3D-nuScenes validation set with aforementioned improvements. **Better viewed in color and zoomed in.**

3.5 Further Work

While our original contribution and subsequent improvements demonstrate real-time inference on front-camera field-of-view semantic occupancy prediction, several avenues for further exploration remain. First, we plan to extend our method to a multi-view camera setup. Multi-view inputs will allow for a more comprehensive 360-degree understanding of the environment, which is crucial for autonomous driving where situational awareness from all directions is necessary. Additionally, multi-view camera data can potentially enhance the accuracy of 3D semantic occupancy predictions by providing minor overlapping perspectives, helping the model to learn some extent of semantic consistency.

Second, we found that the trade-off between computation efficiency and accuracy justifies transitioning from the current projection strategy with raw RGB projection to camera features projection. Additionally, both LiDAR and camera features should undergo independent feature extraction before sensor fusion. Although this change may introduce a potential increase in computational cost due to an additional LiDAR feature extractor module, it is crucial to maximize the amount of sensor information input at the fusion stage to enrich the 3D representation from the beginning. Overall, we still abide with early sensor fusion but delay the fusion stage until after each sensor modality has undergone feature extraction in its respective domain. Camera features are then lifted to 3D using LiDAR points as geometric anchors, followed by sensor fusion.

Further expanding on this, the sensor fusion step itself can be significantly enhanced by employing an attention mechanism as a more adaptive form of fusion. Our current approach uses a form of early fusion based on geometric projection, which rigidly combines features. In contrast, an attention-based fusion mechanism,

such as cross-attention, treats one modality (e.g., camera features) as the query and another (e.g., LiDAR features) as the key and value. The model can then learn to dynamically weight the importance of features from each sensor. For example, in a well-lit area with rich color, the model might learn to assign a higher weight to camera features for semantic understanding. Conversely, in a poorly lit or geometrically complex area, it might learn to rely more heavily on the precise depth information from LiDAR.

Chapter 4

MinkOcc: Towards real-time label-efficient semantic occupancy prediction

As part of this integrated thesis, we now present our second paper on this work, which is submitted for review at the 2025 IEEE International Conference on Intelligent Robots and Systems. The paper is followed by a brief discussion.

MinkOcc: Towards real-time label-efficient semantic occupancy prediction

Samuel Sze[1], Daniele De Martini[1] and Lars Kunze[1,2]

***Abstract*— Developing 3D semantic occupancy prediction models often relies on dense 3D annotations for supervised learning, a process that is both labor and resource-intensive, underscoring the need for label-efficient or even label-free approaches. To address this, we introduce MinkOcc, a multi-modal 3D semantic occupancy prediction framework for cameras and LiDARs that proposes a two-step semi-supervised training procedure. Here, a small dataset of explicitly 3D annotations warm-starts the training process; then, the supervision is continued by simpler-to-annotate accumulated LiDAR sweeps and images – semantically labelled through vision foundational models. MinkOcc effectively utilizes these sensor-rich supervisory cues and reduces reliance on manual labeling by 90% while maintaining competitive accuracy. In addition, the proposed model incorporates information from LiDAR and camera data through early fusion and leverages sparse convolution networks for real-time prediction. With its efficiency in both supervision and computation, we aim to extend MinkOcc beyond curated datasets, enabling broader real-world deployment of 3D semantic occupancy prediction in autonomous driving.**

I. Introduction

Semantic occupancy prediction is crucial for scene understanding in autonomous vehicles (AVs) [1], combining spatial and semantic information to enhance decision-making. By estimating the occupied state of each voxel in 3D space, it generalizes well to irregular objects, diverse vehicle shapes, and complex road structures. To support this, various 3D occupancy datasets have emerged, including Monoscene [2], SSCBench [3], Occ3D [4], and OpenScene [5], built upon established multi-modal self-driving datasets like nuScenes [6], Waymo [7], and SemanticKITTI [8]. These datasets span different sensor configurations, semantic taxonomies, and geographic regions, covering diverse environments such as highways and urban areas.

Figure 1 shows that training on these well-curated annotations significantly boosts model performance over weakly-supervised alternatives. However, large-scale dense annotations across driving scenes remain costly and impractical. For instance, OpenOccupancy [9] required approximately 4000 hours of manual labeling for 34,000 annotated frames [1]. This difficulty arises from the need to manually label occluded regions with limited sensor data, and account for dynamic object movements that create spatial-temporal "tubes" to ensure temporal consistency of semantic classes across frames [3].

[1]Samuel Sze, Daniele De Martini and Lars Kunze are with the Oxford Robotics Institute, Department of Engineering Science, University of Oxford: (samuels,lars,daniele)@robots.ox.ac.uk
[2]Lars Kunze is with the Bristol Robotics Laboratory, School of Engineering, University of the West of England.

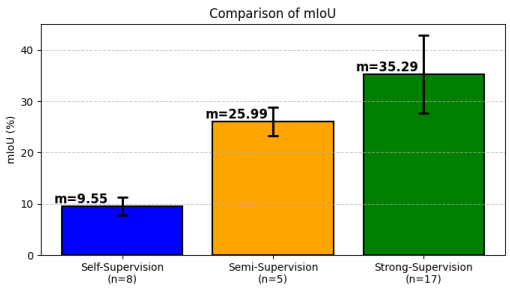


Fig. 1. mIoU of occupancy prediction methods using different supervision signals on Occ3D-nuScenes [4]. Strong-supervision yields the highest mIoU, but its labeling cost is prohibitive. Self-supervision avoids labels but suffers low accuracy. Semi-supervision, where our proposed MinkOcc belongs, offers a practical alternative to reduce labeling costs while maintaining accuracy.

While self-supervised learning could eliminate the need for labels, it remains challenging in outdoor environments due to heavy occlusions and limited overlapping viewpoints amid rapid ego-vehicle motion [10]–[12]. Semi-supervised learning offers a practical alternative, markedly reducing annotation dependency while retaining strong supervision where necessary. For example, VFG-SSC [13] achieves 85% of its fully supervised performance using only 10% of labeled data, and similarly facilitates adaptation across diverse AV datasets [1].

Building on previous work [14], we introduce MinkOcc, a real-time semi-supervised 3D semantic occupancy prediction model. Our model is built upon Minkowski Engine [15], a fully convolutional sparse network serving as the main backbone. For occupancy prediction, we use accumulated LiDAR sweeps as a surrogate for dense 3D ground truth, providing a weak supervision signal for scene completion. For semantic segmentation, we integrate Pulsar [16], an efficient differentiable neural rendering head that bridges multi-view 2D images with 3D voxel features. Additionally, we employ the vision-language models Grounding-DINO and Segment Anything (SAM) [17], [18] to generate 2D pseudo-ground-truth labels for semantic supervision. Thanks to its sparse design, MinkOcc achieves real-time inference speeds.

The main contributions of this work are:

- A fully sparse, multi-modal 3D semantic occupancy prediction model capable of real-time inference, demonstrating state-of-the-art performance.
- A semi-supervised strategy that leverages LiDAR-accumulated sweeps for occupancy completion, as well

as differentiable rendering and foundational models for semantic segmentation, thereby reducing reliance on dense and expensive 3D annotations.

II. Related Work

A. Multi-modal occupancy prediction

Multi-modal 3D semantic occupancy prediction not only leverages richer sensory input but also enhances robustness to sensor failures, such as poor camera visibility at night or reduced LiDAR accuracy in adverse weather [9]. In the camera branch, methods typically rely on forward projection via probabilistic depth estimation [19] or backward projection using either interpolation [20] or cross-attention [21]–[23]. In the LiDAR branch, feature extracting backbones such as VoxelNet [24] and PointPillars [25] process point clouds into structured representations for downstream processing.

Despite the upsides of multi-modality, [1] reports that LiDAR-centric methods outperform multi-modal or camera-centric approaches in SemanticKITTI [8] and SSCBench [3]. Indeed, while LiDAR offers the fastest and most straightforward way for accurate depth, it has often been overlooked in favour for more sophisticated attention methods. Many multi-modal methods predefine 3D voxelized coordinate grids [26]–[29] to query camera features, but these grids frequently include occluded or out-of-view regions, leading to wasted computation. Some works address this by densifying LiDAR points [30] or aggregating LiDAR sweeps over time [31] before sensor fusion. In contrast, our approach drastically simplifies the problem by densifying points after fusion, leveraging Minkowski Engine's generative capabilities [32] to handle the complexity of scene completion. As such, sensor fusion is performed through a simple projection transformation of LiDAR points, followed by sampling on the image plane to extract multi-scale camera features, aggregating to a sparse multi-modal feature volume.

B. Semi-supervision in occupancy prediction

Recent trends in semi-supervision use 2D semantic and depth images as ground truth or derive sparse ground truth by projecting LiDAR semantic segmentation labels onto images [12], [33], [34]. First of all, if LiDAR semantic segmentation ground truth is available, it may be more advantageous to utilize it directly in 3D space and create sparse semantic annotations by accumulating sweeps. In terms of 2D annotation ground truth, curating them across tens of thousands of frames still remains labor-intensive and time-consuming. Therefore, we leverage vision-language models [17], [18] to rapidly generate 2D pseudo-labels, each assigned a confidence score that reflects the model's certainty, allowing us to weigh labels accordingly and mitigate errors.

Drawing inspiration from VAMPIRE [33], we aim to construct a robust 3D feature representation of an AV driving scene in a label-efficient manner, which means it is important to have strong 3D supervision signals during the initial learning stage. Conventional semi-supervised methods that exploit 3D labels typically follow a self-training protocol [13], [35], where a model is first trained on a small labeled subset, then used to generate pseudo-labels for a larger unlabeled dataset, and subsequently refined using both the original and pseudo-labeled data. However, this approach requires three training phases and is also prone to model overfitting on the small dataset. Instead, we propose integrating weaker 2D pseudo-label signals into the framework to consolidate this multi-step process into a single training phase. In the initial warm-start phase, we leverage dense 3D annotations to guide learning. Once the warm-start iterations are completed, we drop the 3D semantic signal and continue training with 2D supervision.

C. Differentiable neural rendering

Differentiable rendering enables end-to-end learning of 3D scene representations from 2D image observations. It typically comprises of three stages: a 3D scene representation (e.g., NeRF [36], Gaussian Splatting [37], or occupancy-based models [38]), a projection method that maps 3D data to 2D via volume rendering, point splatting, or mesh rasterization, and neural shading that refines the 2D output. Many weakly supervised 3D semantic occupancy prediction approaches actively incorporate differentiable rendering [10]–[12], [34], [39]–[41]. RenderOcc [12] introduces a signed density field (SDF) with temporal auxiliary rays to improve multi-view consistency, while OccNeRF [10] extends NeRF with parameterized occupancy fields to render depth and semantic images. GSrender [34] and GaussTR [39] utilize Gaussian splatting for its real-time rendering capabilities.

However, NeRF-based methods rely on volumetric representations that, while enabling fine-grained scene reconstruction, are computationally expensive and require dense multi-view supervision, limiting their practicality in AV scenarios with sparse viewpoints. Gaussian splatting depends on anisotropic covariance matrices for shape adaptation, making optimization difficult in AV environments with only 2-6 views per scene. To address these limitations, we adopt Pulsar [16], a spherical differentiable renderer optimized for high-speed, CUDA-accelerated rendering. By representing 3D scenes as isotropic spheres, Pulsar enables efficient, geometrically consistent rendering of voxel grids while seamlessly integrating with PyTorch as a fully differentiable component.

III. Problem Formulation

We tackle 3D semantic occupancy prediction using six surround-view cameras and a top-mounted LiDAR sensor. The inputs include multi-view images $I = \{I_i\}_{i=1}^{N}$, where each $I_i \in \mathbb{R}^{W \times H \times 3}$ represents an image from the i-th camera, and a LiDAR point cloud $P \in \mathbb{R}^{N \times (3+f)}$ containing 3D coordinates and additional features. The output is a voxel grid $V \in \mathbb{R}^{X \times Y \times Z}$, where each voxel v_{xyz} is assigned a class label from C semantic categories. We adopt a semi-supervised approach, leveraging pseudo-labels from (1) 2D semantic segmentation of current-frame images, $y_{2D} = \{y_{2D,i}\}_{i=1}^{N}$, and (2) accumulated LiDAR sweeps over the past K frames, $P_{\text{acc}} = \{P_t\}_{t=-K}^{0}$.

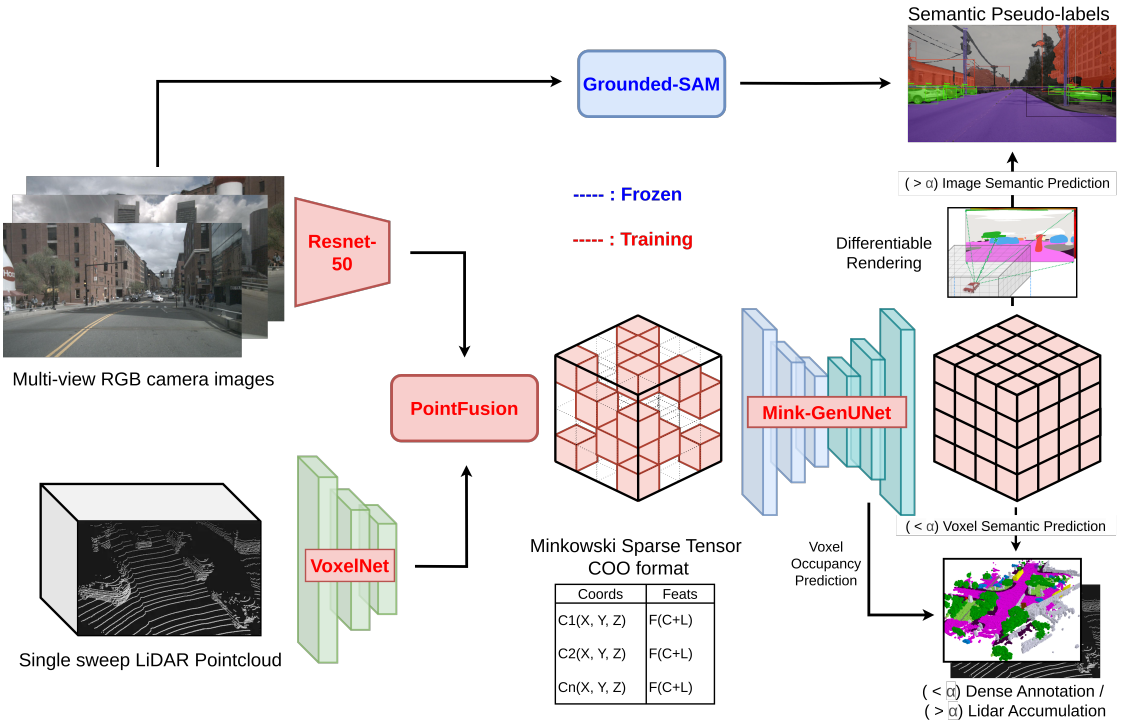


Fig. 2. **Overview of system pipeline**. Our model predicts dense, 3D semantic occupancy maps from LiDAR and camera information. It is trained in two steps. First, we warm-start the prediction model through $\alpha = 10\%$ of dense 3D semantic annotations from Occ3D-nuScenes; then, the voxel semantic prediction branch is turned off, and cheaper LiDAR accumulated sweeps and image semantic maps replace dense annotations. The supervision of the images is provided through a differentiable rendering approach, which projects the semantic information in the camera frames. 3D sparse and dense representations are converted to Coordinate List Format (COO) to ensure compatibility with Minkowski Engine [15].

IV. Method

A. System Pipeline and Training Procedure

Figure 2 shows the system pipeline. We extract 2D features from multi-view images I_i using ResNet-50 [42] and process LiDAR $P \in \mathbb{R}^{N\times(3+f)}$ via MVX-Net [43], an extension of VoxelNet [24]. We utilize PointFusion [44]'s dense fusion to project LiDAR points as spatial anchors onto corresponding image planes. Multi-scale image features are then sampled and adaptively fused across multiple views. The final representation is refined through a shared-weight MLP, producing a sparse yet contextually rich voxelized 3D feature grid.

Following Minkowski Engine's coordinate list format, we convert the 3D feature grid into sparse tensors to be passed into Mink-GenUNet [32], a Generative UNet [45] which performs sparse-to-dense occupancy completion. Pruning is applied at each upsampling layer to remove newly created coordinates using a probability function guided by binary classification against ground truth. Similar to UNet [45] architectures, we add skip connections after each generative upsampling to integrate feature details from the encoder.

Each voxel in the densified 3D feature grid is then classified through a softmax operation into one of the C semantic categories. We train our system in two separate phases requiring different annotation types. First, we warm-start our training using a small dataset of dense 3D annotations. These will directly supervise the 3D occupancy and semantic classification. In this phase, we denote with α the amount of ground truth data used as a percentage of the total annotated dataset. After this α warm-start, we switch to accumulated LiDAR sweeps P_{acc} for occupancy classification, which are cheaper to annotate but may contain incongruence due to misalignments or moving objects. For semantic classification, a differentiable rendering head performs 3D-to-2D rasterization at each camera position. We convert voxel features into points at their centers and model them as spheres with learnable radii and opacity. These spheres are then rendered into the 2D image space, where each pixel accumulates contributions from a predefined number of spheres in a back-to-front compositing order. This results in a 2D image feature, which can then be optimized using 2D pseudo labels y_{2D} generated from Gounding-Dino [17] and SAM [18].

B. Minkowski Convolution Blocks

Minkowski Engine employs a generalized convolution [15] as described in Equation 1, where x^u_{out} represents the output feature at point u.

$$x^u_{\text{out}} = \sum_{i\in\mathcal{N}^D(u,C_{\text{in}})} W_i \cdot x^{u+i}_{\text{in}} \quad \text{for} \quad u \in C_{\text{out}} \tag{1}$$

The neighborhood of input coordinates around u is defined by $\mathcal{N}^D(u, C_{\text{in}})$, which specifies the set of offsets contributing to the sparse convolution operation. The convolution is governed by the learnable weights W_i of the kernel, while x^{u+i}_{in} represents the input feature at the position offset by i

from u. Generalized convolution allows for arbitrary kernel shapes, enabling the direct adoption of 2D architectural design into higher-dimensional networks. Mink-GenUNet utilize two Minkowski convolution layers, each followed by batch normalization and ReLU activation. A residual connection adds the input back to the output after the second convolution. We also insert Minkowski Squeeze and Excite (SE) modules to recalibrate channel-wise features through global average pooling, enhancing focus on key features across sparse data [46]. To perform sparse-to-dense completion, Minkowski generative transposed convolution blocks are used to expand the sparse tensor's support region. This is determined by the outer product of the convolution kernel applied to the input sparsity pattern, expressed as: $\text{support}(T) = C \otimes [-K, \dots, K]^3$ where C is the input sparsity pattern and K is the kernel size.

C. Spherical Differentiable Rendering

3D voxels are represented as spheres S, where each sphere is parameterized by its voxel center in 3D Cartesian coordinates v_{xyz}, Mink-GenUNet feature vector $f_{xyz} \in \mathbb{R}^d$, radius $r_{xyz} \in \mathbb{R}$, and opacity $o_{xyz} \in \mathbb{R}$. Each scene is learned from a set of training images $I = \{I_i\}_{i=1}^N$ using their corresponding camera parameters, including rotation R_i, translation t_i, and intrinsic matrix K_i. Given these inputs, spherical differentiable rendering performs differentiable projection $Proj$ and neural shading NS.

The projection step computes a blending function w_{xyz} that determines how spheres contribute to each pixel based on their depth ordering, spatial proximity, and opacity. This is calculated by

$$w_{xyz} = \frac{o_{xyz} \cdot d_{xyz} \cdot \exp(o_{xyz} \cdot z_{xyz}\gamma)}{\exp(\varepsilon\gamma) + \sum_k o_k \cdot d_k \cdot \exp(o_k \cdot z_k\gamma)}, \quad (2)$$

where z_{xyz} represents the normalized depth of the sphere, $d_{xyz} = \min(1, ||\mathbf{d}_{xyz}||_2 / r_{xyz})$ accounts for the orthogonal distance of the ray to the sphere center, and γ controls the depth sensitivity of the weighting function. Small values of γ enforce sharper blending, while larger values allow for a smoother transition between overlapping spheres. The additional term ε ensures numerical stability, preventing division errors. Equation 2 also ensures that spheres closer to the camera and with higher opacity contribute more significantly to the final pixel value.

After projection, we aggregate the weighted contributions from all intersecting spheres on each pixel. Given the per-pixel blending weights w_{xyz}, the rendered feature value at each pixel is computed as a weighted sum: $F_{uv} = \sum w_{xyz} f_{xyz}$.. The shading function NS with its parameters Θ then transforms the accumulated features F, which encapsulates all camera viewpoints, into semantic prediction $\hat{y_{2D}} = NS(F; \Theta)$. Θ is essentially a convolutional UNet which performs semantic segmentation in our case.

Altogether, we form Equation 3. The overall optimization objective is to minimize the difference between predicted semantics $\hat{y_{2D}}$ and ground-truth pseudo-labels y_{2D}. As such, we refine both the spheres S and neural shader parameters Θ to reach their optimized levels S^*, Θ^*.

$$S^*, \Theta^* = \arg\min_{S,\Theta} \sum_i |y_{2D,i} - NS\left(Proj(S, R_i, t_i, K_i); \Theta\right)| \quad (3)$$

D. Loss Function

We use three losses to supervise our training. $\mathcal{L}_{\text{bce}}$ is a voxel-based binary cross-entropy loss applied for sparse-to-dense occupancy completion across the decoder layers. We adopt $\mathcal{L}_{\text{3D_ce}}$ as a dense 3D class-balanced cross-entropy loss against 3D annotation. Class balancing reweights each semantic category based on its frequency, calculated using an effective number of voxel samples per class with $\beta = 0.9$ over the entire dataset [47].

$$\mathcal{L}_{\text{3D_ce}}(\hat{y}, y) = -\frac{1-\beta}{1-\beta^{n_y}} \log\left(\frac{\exp(\hat{y}_y)}{\sum_{j=1}^{C} \exp(\hat{y}_j)}\right). \quad (4)$$

2D pseudo-labels are not perfect and can be prone to errors. Therefore, we use a soft cross-entropy loss weighted by the confidence score from Grounded-DINO's object detection. For overlapping detections on the same image pixel, we select the highest confidence score among them. The soft cross-entropy loss is given by:

$$\mathcal{L}_{\text{2D_ce}} = -\sum_{i \in \mathcal{I}} c_i \cdot y_{\text{2D},i} \log \hat{y}_{\text{2D},i}, \quad (5)$$

where $y_{\text{2D},i}$ is the one-hot pseudo-label from Grounded-DINO and SAM, $\hat{y}_{\text{2D},i}$ is the predicted softmax probability for pixel i, and c_i is the confidence score assigned to each pseudo-label. Unlabeled pixels (class 0) are ignored. This formulation allows high-confidence labels to contribute more significantly to the loss while low-confidence labels contribute less.

The total loss is shown in Equation 6, balanced by their respective hyperparameters λ_1 and λ_2:

$$\mathcal{L}_{\text{total}} = \begin{cases} \lambda_1 \mathcal{L}_{\text{3D_ce}} + \lambda_2 \mathcal{L}_{\text{bce}}, & \text{if warm-start phase} \\ \lambda_1 \mathcal{L}_{\text{2D_ce}} + \lambda_2 \mathcal{L}_{\text{bce}}, & \text{if after warm-start} \end{cases} \quad (6)$$

In our training, the warm-start phase employs dense 3D annotations for occupancy completion and semantic segmentation using $\mathcal{L}_{\text{bce}}$ and $\mathcal{L}_{\text{3D_ce}}$, respectively. After the warm-start, semantic supervision transitions to 2D pseudo-labels via $\mathcal{L}_{\text{2D_ce}}$, while occupancy completion continues with LiDAR accumulation pseudo ground truth using $\mathcal{L}_{\text{bce}}$. λ_1 and λ_2 are empirically set, with $\lambda_1 = 0.5$ and $\lambda_2 = 1$.

V. Experimental Setup

A. Datasets

We conduct our experiments on the Occ3D-nuScenes [4] dataset, which consists of 28,130 training scenes and 6,019 validation scenes. Each scene includes six RGB cameras forming a 360-degree surround view, calibrated LiDAR

TABLE I

QUANTITATIVE RESULTS ON OCC3D-NUSCENES VALIDATION DATASET. SELF-SUPERVISION MIOU EXCLUDES *others* AND *other flat* AS THEY ARE NOT WELL DEFINED IN 2D LABELS [10]. BEST RESULTS FOR EACH SEMANTIC CLASS ARE BOLDED.

Method	Sup.	RayIoU	mIoU	others	barrier	bicycle	bus	car	const. veh.	motorcycle	pedestrian	traffic cone	trailer	truck	driv. surf.	other flat	sidewalk	terrain	manmade	vegetation
OccNeRF [10]	Self	-	10.81	-	**0.83**	**0.82**	5.13	12.49	**3.50**	**0.23**	**3.10**	**1.84**	**0.52**	**3.90**	52.62	-	20.81	24.75	18.45	13.19
SelfOcc [11]	Self	-	9.03	-	0.00	0.00	0.00	10.03	0.00	0.00	0.00	0.00	0.00	**7.11**	52.96	-	**23.59**	**25.16**	11.97	4.61
MinkOcc	Self	**12.5**	**13.23**	-	0.00	0.00	**7.92**	**13.83**	0.00	0.00	0.00	0.00	0.00	5.75	**70.27**	-	17.60	20.08	**35.97**	**26.98**
RenderOcc [12]	Semi	19.50	23.93	5.69	27.56	14.36	19.91	20.56	11.96	12.42	12.14	14.34	20.81	18.94	68.85	33.35	42.01	43.94	17.36	22.61
VAMPIRE [33]	Semi	-	28.33	7.48	32.64	**16.15**	**36.73**	41.44	16.59	20.64	16.55	15.09	21.02	28.47	67.96	33.73	41.61	40.76	24.53	20.26
GSRender [34]	Semi	-	29.56	**8.42**	34.93	15.12	30.71	29.61	16.70	9.48	17.61	**16.58**	**23.92**	27.24	**77.94**	**39.21**	**51.69**	**54.61**	23.82	24.92
MinkOcc	Semi	**24.60**	**33.43**	1.58	**37.90**	10.24	34.95	**46.20**	**22.08**	**35.07**	**37.46**	15.06	15.59	**35.57**	73.13	37.01	46.84	48.23	**40.61**	**30.75**
Occ3D [4]	Strong	-	28.53	8.09	39.33	20.56	38.29	42.24	16.93	24.52	22.72	21.05	22.98	31.11	53.33	33.84	37.98	33.23	20.79	18.00
FastOcc [48]	Strong	-	40.75	12.86	46.58	29.93	46.07	54.09	23.74	31.10	30.68	28.52	33.08	39.69	83.33	44.65	53.90	55.46	42.61	36.50
FB-OCC [49]	Strong	33.5	42.06	14.30	49.71	**30.00**	46.62	51.54	**29.30**	29.13	29.35	**30.48**	34.97	39.36	83.07	**47.16**	**55.62**	**59.88**	44.89	39.58
MinkOcc	Strong	**40.90**	**44.85**	**20.02**	**56.45**	25.79	**55.60**	**57.13**	27.54	**40.22**	**39.15**	29.59	**41.39**	**46.29**	**85.78**	46.84	45.43	46.80	**50.42**	**48.03**

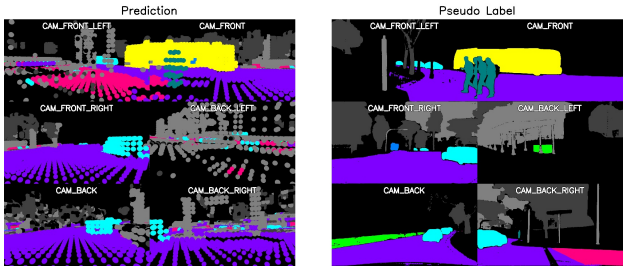


Fig. 3. Comparison of MinkOcc-semi's predicted 3D feature volume rendered in 2D against 2D pseudo-label. Results are from Occ3D-nuScenes validation set. **Better viewed in color and zoomed in**.

scans and voxelized 3D ground truth. The voxel grid has dimensions $200 \times 200 \times 16$, covering a physical space from $[-40\,\text{m}, -40\,\text{m}, -1\,\text{m}]$ to $[40\,\text{m}, 40\,\text{m}, 5.4\,\text{m}]$, with 18 semantic classes, including free space (class 17). We use the masks provided to occlude camera out-of-view voxels. For semi-supervised training, we set warm-start phase as $\alpha = 10\%$ of total scenes from Occ3D-nuScene. We use Grounding-DINO [17] to generate bounding box proposals from text prompts, followed by SAM [18] for per-pixel semantic segmentation, producing 2D pseudo labels. We select semantic classes that align better with 2D representations to improve segmentation quality in perspective space. Since our primary focus is 3D semantic occupancy, we subsequently remap the 2D labels to match the 3D annotations in Occ3D-nuScenes, ensuring consistency across tasks.

B. Training details

Training is configured with the Adam optimizer at a learning rate of 1e-4, and a Cosine-Annealing learning rate schedule. The model is trained for 30 epochs, with a batch size of 2. The hardware setup for the training included 1 NVIDIA RTX 4090. Data augmentation is performed on the camera, LiDAR and annotated 3D ground truth. For 2D data augmentation, we perform random image translation, scaling, rotation and RGB perturbation. For 3D data augmentation, we perform random flipping, rotation about the z-axis and ground truth voxel dropout. We also perform gradient clipping during warm-start phase α transition to stabilize training.

C. Evaluation Metrics

We evaluate our 3D semantic occupancy prediction using the traditional Mean Intersection over Union (mIoU) for voxel-level metrics as well as the Ray-level metric (RayIoU) proposed by SparseOcc [50]. mIoU is defined as $\frac{1}{|C|}\sum_{i \in C} \frac{TP_i}{TP_i+FP_i+FN_i}$ where C represents the semantic classes excluding the "free" class, and TP_i, FP_i, and FN_i denote True Positives, False Positives, and False Negatives for class i. However, imperfections in dense annotations can cause challenges in evaluating physically thin objects and occlusion. This results in disproportionately strict voxel-level metrics, where even a one-voxel deviation may reduce the IoU to zero. As such, to address this limitation, we also use RayIoU, which simulates a LiDAR beam by using query rays to sample the first voxel contact surface.

VI. EXPERIMENTS

A. Semi-supervised Semantic Occupancy Prediction

Referring to Table I, we compare our model's performance with state-of-the-art semi-supervised methods [12], [33], [34]. With a warm-start phase of $\alpha = 10\%$ on the Occ3D-nuScenes dataset, our model demonstrates strong performance across key dynamic object categories. It achieves competitive results in *cars*, *motorcycles*, *pedestrians*, and *trucks*, demonstrating the effectiveness of 2D supervision in compensating for limited 3D labels. Figure 3 compares our rendered 2D predictions with pseudo-labels from the Occ3D-nuScenes validation set, illustrating how the model refines semantic segmentation in perspective space. By training with 2D pseudo-labels from vision-language models, the model learns a semantically meaningful 3D feature representation, improving both 2D segmentation accuracy and its 3D semantic occupancy predictions.

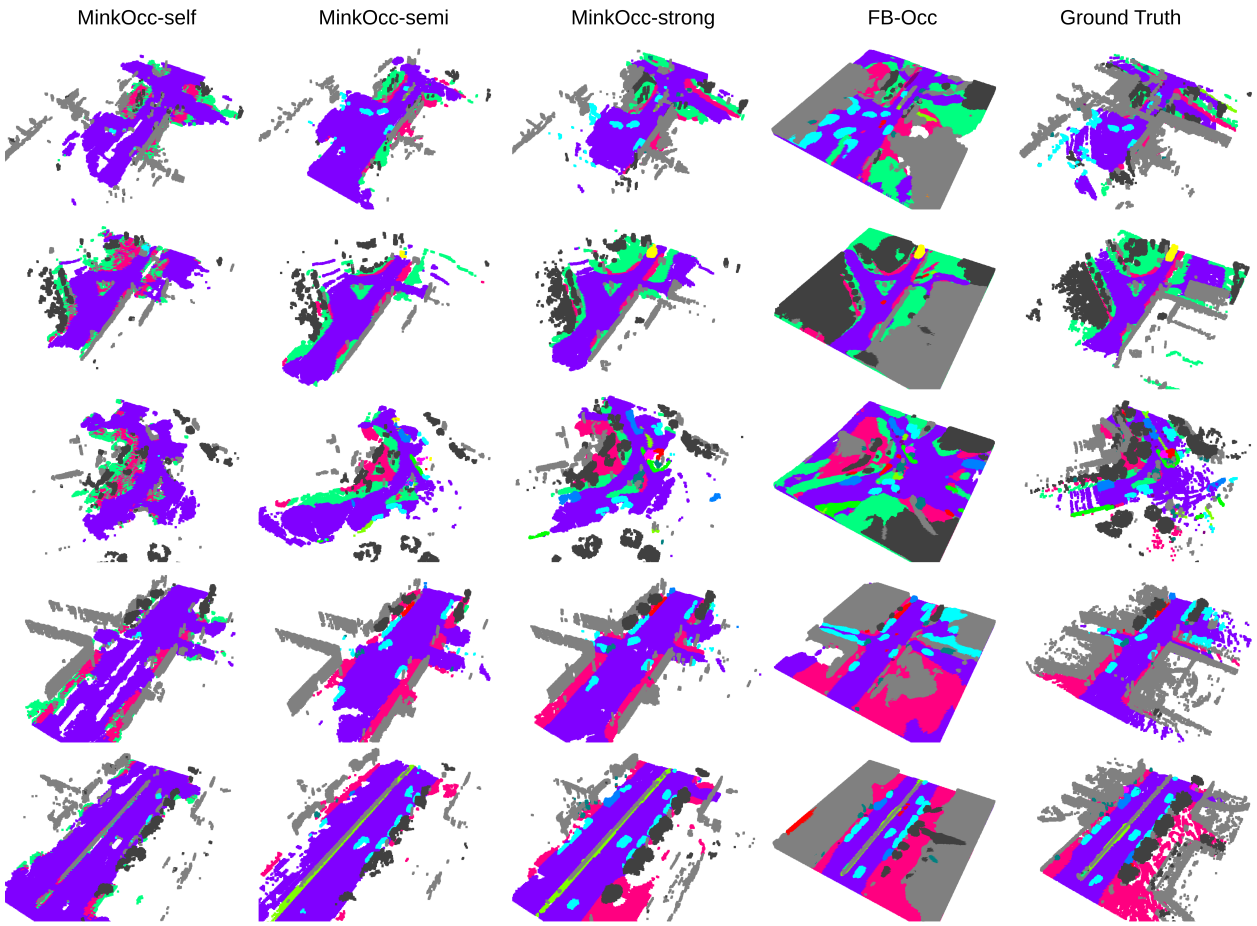


Fig. 4. Qualitative results on Occ3D-nuScenes validation across different MinkOcc warm-start phase percentage setup against FB-Occ [49] and Ground Truth. **Better viewed in color and zoomed in.**

Performance in other classes is slightly weaker, likely due to suboptimal pseudo-label generation. For example, in Figure 3, certain pseudo-label frames fail to annotate regions of *drivable surface* and *sidewalk*. Errors in pseudo-labels, which is reflected by the soft cross-entropy 2D loss, weakens supervision for these classes. Addressing this may require improving pseudo-label generation or refining the model's handling of uncertain supervision.

As shown in Figure 4, our method maintains strong geometrical accuracy, ensuring precise voxel alignment with real-world positions. This is evident in dense urban traffic scenes such as row 1, row 3, and row 4, where *cars*, *pedestrians*, and *trucks* retain their correct shapes instead of appearing stretched or merged, as seen in FB-Occ. The high RayIoU score further indicates that the model reconstructs 3D structures without hallucinating excessive thickness behind occlusions, making it more reliable in cluttered environments.

In semi-supervised settings, sparsity plays a crucial role in scene densification and semantic accuracy. When voxelized, LiDAR accumulation provides fewer points than dense 3D annotations, and at coarser voxel resolutions, multiple LiDAR points may merge into a single voxel. Additionally, the voxel-to-point-to-sphere transformation introduces information loss by collapsing each voxel into a single point before rendering, discarding some semantic details. Referring to Figure 3, these factors result in fewer rendered spheres for *pedestrians* and distant *cars*, limiting the model's ability to align predictions with pseudo-labels. To better understand these effects, we conduct an ablation study on voxel resolution and its impact on accuracy and inference speed. Further research is also needed to improve voxel-to-sphere transformation pipeline to avoid unnecessary information loss.

In terms of inference speed, referring to Table II, we achieve a potential inference speed of 44.69 milliseconds (ms) on MinkOcc-semi, representing a 2-times increase compared to the TensorRT-optimized FastOcc [48] which runs at 80 ms. This translates to 23 FPS while utilizing only 1.65 GB of GPU memory. Our video demo further demonstrates the model's capability of real-time inference.

TABLE II
EFFECT OF VOXEL RESOLUTION AND α ON MODEL PERFORMANCE

Voxel Resolution (m)	$\boldsymbol{\alpha}$ (%)	**Latency (ms)**	**mIoU** (%)
0.4	100	43.18	44.85
0.4	10	44.69	33.43
0.4	0	42.47	13.23
0.2	10	54.99	36.23
0.1	10	102.31	37.05

B. Ablation Study

To evaluate the impact of dense 3D supervision, we vary warm-start phase α which controls the fraction of training samples with dense 3D annotations. At $\alpha = 0\%$, MinkOcc operates in a fully self-supervised setting, relying solely on 2D pseudo-labels and LiDAR-accumulated scans. While this enables open-vocabulary classification via Grounded-SAM, we remap 2D labels back to 3D (Section V.B) for fair comparison with other methods. As shown in Table I and Figure 4, the model effectively segments larger road-layout-specific classes such as *drivable surface*, *manmade*, *sidewalk*, and *vegetation*. However, it struggles with smaller, less frequent objects, indicating that 2D semantic pseudo-labels alone is insufficient for self-supervised semantic occupancy prediction. Additional supervision from LiDAR, camera signals, or temporal information is needed.

At $\alpha = 100\%$, MinkOcc operates in a strongly supervised mode, with 2D pseudo-labels acting as auxiliary supervision alongside dense 3D annotation. MinkOcc-strong achieves better overall semantic accuracy over Occ3D, FB-Occ, and FastOcc while maintaining efficient inference. Similar to MinkOcc-semi, it reconstructs scenes with high geometric fidelity, avoiding generating artifacts that distort object shapes and road layouts.

We analyze the effect of voxel resolution on accuracy and inference speed. Table II highlights the trade-off between voxel size, latency, and mIoU. While higher resolution (smaller voxels) improves mIoU, the computational cost rises exponentially. The limited accuracy gains at 0.1m suggest that this voxel resolution exceeds the inherent spatial precision of LiDAR sensors in nuScenes [6], leading to an occupancy grid of many empty voxels. A moderate upsample in voxel resolution (e.g., from 0.4m to 0.2m) in a semi-supervised setting provides a good balance, improving performance with minimal impact on efficiency.

VII. CONCLUSION

In this work, we introduced MinkOcc, a real-time, label-efficient 3D semantic occupancy prediction model that leverages semi-supervision to reduce reliance on dense 3D annotations. Our approach effectively balances accuracy and computational efficiency by integrating sparse convolution networks with multi-modal sensor fusion and differentiable rendering. Experimental results on the Occ3D-nuScenes dataset demonstrate that MinkOcc-semi achieves competitive performance with significantly fewer annotations, particularly excelling in dynamic object segmentation while maintaining real-time inference speeds.

Our ablation study also examines voxel resolution, supervision quality, and computational cost in 3D semantic occupancy prediction. In our main experiments, we adopt a voxel resolution of 0.4 m to align with the resolution provided by dense 3D annotations. Notably, a finer resolution of 0.2 m yields better performance, suggesting that capturing finer spatial details can be advantageous under weak supervision. However, the benefits of higher voxel resolution are limited by the inherent sparsity of LiDAR-accumulated sweeps. Overall, while strongly supervised methods continue to set the benchmark, our results indicate that semi-supervised approaches hold promise for AV datasets with limited dense annotations.

Future work could improve 2D supervision quality by enhancing semantic pseudo-label accuracy through depth and image feature maps. Another key direction is to develop a denser rendering approach for voxel grids, either via direct voxel rendering [51] or a lossless voxel-to-sphere transformation, to mitigate information loss. Another direction would be applying MinkOcc to diverse AV datasets to further validate its robustness and generalization.

ACKNOWLEDGMENT

This project was supported by the EPSRC project RAILS (EP/W011344/1) and the Oxford Robotics Institute's research project RobotCycle.

REFERENCES

[1] H. Xu, J. Chen, S. Meng, Y. Wang, and L.-P. Chau, "A survey on occupancy perception for autonomous driving: The information fusion perspective," 2024.

[2] A.-Q. Cao and R. de Charette, "Monoscene: Monocular 3d semantic scene completion," 2022.

[3] Y. Li, S. Li, X. Liu, M. Gong, K. Li, N. Chen, Z. Wang, Z. Li, T. Jiang, F. Yu, Y. Wang, H. Zhao, Z. Yu, and C. Feng, "SSCBench: Monocular 3d semantic scene completion benchmark in street views," 2024. [Online]. Available: https://openreview.net/forum?id=8fJEOri51F

[4] X. Tian, T. Jiang, L. Yun, Y. Wang, Y. Wang, and H. Zhao, "Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving," *arXiv preprint arXiv:2304.14365*, 2023.

[5] O. Contributors, "Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving," https://github.com/OpenDriveLab/OpenScene, 2023.

[6] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[7] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[8] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9297–9307.

[9] X. Wang, Z. Zhu, W. Xu, Y. Zhang, Y. Wei, X. Chi, Y. Ye, D. Du, J. Lu, and X. Wang, "Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception," *arXiv preprint arXiv:2303.03991*, 2023.

[10] C. Zhang, J. Yan, Y. Wei, J. Li, L. Liu, Y. Tang, Y. Duan, and J. Lu, "Occnerf: Advancing 3d occupancy prediction in lidar-free environments," 2024. [Online]. Available: https://arxiv.org/abs/2312.09243

[11] Y. Huang, W. Zheng, B. Zhang, J. Zhou, and J. Lu, "Selfocc: Self-supervised vision-based 3d occupancy prediction," 2023. [Online]. Available: https://arxiv.org/abs/2311.12754

[12] M. Pan, J. Liu, R. Zhang, P. Huang, X. Li, B. Wang, H. Xie, L. Liu, and S. Zhang, "Renderocc: Vision-centric 3d occupancy prediction with 2d rendering supervision," 2024. [Online]. Available: https://arxiv.org/abs/2309.09502

[13] D.-H. Pham, D.-D. Nguyen, A. Pham, T. Ho, P. Nguyen, K. Nguyen, and R. Nguyen, "Semi-supervised 3d semantic scene completion with 2d vision foundation model guidance," 2025. [Online]. Available: https://arxiv.org/abs/2408.11559

[14] S. Sze and L. Kunze, "Real-time 3d semantic occupancy prediction for autonomous vehicles using memory-efficient sparse convolution," 2024. [Online]. Available: https://arxiv.org/abs/2403.08748

[15] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.

[16] C. Lassner and M. Zollhöfer, "Pulsar: Efficient sphere-based neural rendering," 2020. [Online]. Available: https://arxiv.org/abs/2004.07484

[17] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," 2024. [Online]. Available: https://arxiv.org/abs/2303.05499

[18] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023. [Online]. Available: https://arxiv.org/abs/2304.02643

[19] J. Philion and S. Fidler, "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*. Springer, 2020, pp. 194–210.

[20] A. W. Harley, Z. Fang, J. Li, R. Ambrus, and K. Fragkiadaki, "Simple-bev: What really matters for multi-sensor bev perception?" in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2759–2765.

[21] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *European conference on computer vision*. Springer, 2022, pp. 1–18.

[22] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. L. Rus, and S. Han, "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2774–2781.

[23] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3d semantic occupancy prediction," 2023.

[24] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[25] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," 2019. [Online]. Available: https://arxiv.org/abs/1812.05784

[26] J. Zhang, Y. Ding, and Z. Liu, "Occfusion: Depth estimation free multi-sensor fusion for 3d occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2403.05329

[27] Z. Ming, J. S. Berrio, M. Shan, and S. Worrall, "Occfusion: Multi-sensor fusion framework for 3d semantic occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2403.01644

[28] Z. Zhuang, Z. Wang, S. Chen, L. Liu, H. Luo, and M. Tan, "Robust 3d semantic occupancy prediction with calibration-free spatial transformation," 2024. [Online]. Available: https://arxiv.org/abs/2411.12177

[29] Z. Yang, Y. Dong, H. Wang, L. Ma, Z. Cui, Q. Liu, and H. Pei, "Daocc: 3d object detection assisted multi-sensor fusion for 3d occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2409.19972

[30] M. Seong, J. Kim, G. Bang, H. Jeong, and J. W. Choi, "Mr-occ: Efficient camera-lidar 3d semantic occupancy prediction using hierarchical multi-resolution voxel representation," 2024. [Online]. Available: https://arxiv.org/abs/2412.20480

[31] J. Zhang, Y. Ding, and Z. Liu, "Occloff: Learning optimized feature fusion for 3d occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2411.03696

[32] J. Gwak, C. B. Choy, and S. Savarese, "Generative sparse detection networks for 3d single-shot object detection," in *European conference on computer vision*, 2020.

[33] J. Xu, L. Peng, H. Cheng, L. Xia, Q. Zhou, D. Deng, W. Qian, W. Wang, and D. Cai, "Regulating intermediate 3d features for vision-centric autonomous driving," 2023. [Online]. Available: https://arxiv.org/abs/2312.11837

[34] Q. Sun, C. Shu, S. Zhou, Z. Yu, Y. Chen, D. Yang, and Y. Chun, "Gsrender: Deduplicated occupancy prediction via weakly supervised 3d gaussian splatting," 2024. [Online]. Available: https://arxiv.org/abs/2412.14579

[35] A. N. Ganesh, "Soccdpt: 3d semantic occupancy from dense prediction transformers trained under memory constraints," *Advances in Artificial Intelligence and Machine Learning*, vol. 04, no. 02, p. 2201–2212, 2024. [Online]. Available: http://dx.doi.org/10.54364/AAIML.2024.42126

[36] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," 2020. [Online]. Available: https://arxiv.org/abs/2003.08934

[37] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," 2023. [Online]. Available: https://arxiv.org/abs/2308.04079

[38] Z. Wang, "3d representation methods: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2410.06475

[39] H. Jiang, L. Liu, T. Cheng, X. Wang, T. Lin, Z. Su, W. Liu, and X. Wang, "Gausstr: Foundation model-aligned gaussian transformer for self-supervised 3d spatial understanding," 2024. [Online]. Available: https://arxiv.org/abs/2412.13193

[40] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Gaussianformer: Scene as gaussians for vision-based 3d semantic occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2405.17429

[41] Y. Huang, A. Thammatadatrakoon, W. Zheng, Y. Zhang, D. Du, and J. Lu, "Gaussianformer-2: Probabilistic gaussian superposition for efficient 3d occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2412.04384

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[43] V. A. Sindagi, Y. Zhou, and O. Tuzel, "Mvx-net: Multimodal voxelnet for 3d object detection," 2019. [Online]. Available: https://arxiv.org/abs/1904.01649

[44] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3d bounding box estimation," 2018. [Online]. Available: https://arxiv.org/abs/1711.10871

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[46] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," 2019. [Online]. Available: https://arxiv.org/abs/1709.01507

[47] T. Vu, J.-H. Kim, M. Kim, S. Jung, and S.-G. Jeong, "Milo: Multi-task learning with localization ambiguity suppression for occupancy prediction," 2023.

[48] J. Hou, X. Li, W. Guan, G. Zhang, D. Feng, Y. Du, X. Xue, and J. Pu, "Fastocc: Accelerating 3d occupancy prediction by fusing the 2d bird's-eye view and perspective view," 2024. [Online]. Available: https://arxiv.org/abs/2403.02710

[49] Z. Li, Z. Yu, D. Austin, M. Fang, S. Lan, J. Kautz, and J. M. Alvarez, "FB-OCC: 3D occupancy prediction based on forward-backward view transformation," *arXiv:2307.01492*, 2023.

[50] H. Liu, Y. Chen, H. Wang, Z. Yang, T. Li, J. Zeng, L. Chen, H. Li, and L. Wang, "Fully sparse 3d occupancy prediction," 2024. [Online]. Available: https://arxiv.org/abs/2312.17118

[51] C. Sun, J. Choe, C. Loop, W.-C. Ma, and Y.-C. F. Wang, "Sparse voxels rasterization: Real-time high-fidelity radiance field rendering," 2025. [Online]. Available: https://arxiv.org/abs/2412.04459

4.1 Design Choice: Spherical Differentiable Rendering

4.1.1 Point Rendering

Point rendering represents 3D scenes as collections of discrete points rather than continuous surfaces like meshes [86, 87]. Unlike mesh-based methods, point rendering does not require predefined topological structures, making it particularly suitable for dynamic and unstructured environments. This flexibility allows for efficient scene representation and enables adaptive sampling of geometry, where only the most relevant parts of the scene are stored and processed.

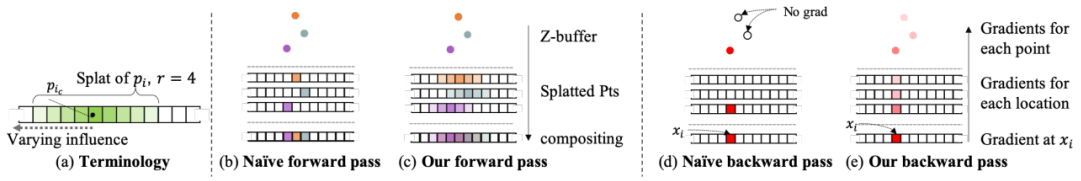


Figure 4.1: Synsin's [86] point rendering pipeline uses an influence region determined by α-compositing for forward pass. As such, in back-propagation, gradients flow to all of the points (Source from [86]).

Rendering a 3D point cloud requires projecting each point onto a 2D image plane using camera parameters such as intrinsic and extrinsic matrices. This process determines how each 3D point contributes to the final image from a given viewpoint. A challenge in this projection step is handling visibility, as multiple points may project onto the same pixel. To resolve depth ordering, traditional methods rely on a z-buffer, which retains only the closest point per pixel [86]. However, this hard selection introduces discontinuities and aliasing artifacts, making it unsuitable for gradient-based learning. To address this, modern differentiable renderers use soft visibility functions, which approximate the contribution of multiple points continuously, allowing for stable gradient propagation.

Once projected, the opacity and attributes of each point must be blended to produce the final pixel value. This is achieved through volumetric compositing, where contributions from multiple points along the same viewing ray are aggregated based on their depth and opacity. A widely used technique for this is α-compositing, which blends values by weighting them according to their opacity and depth [86, 88, 89]. While this is commonly used for color rendering, the same principles can be applied to project semantic features from 3D to 2D. Instead of color attributes, each point can store high-dimensional semantic features, which are then projected and blended into the 2D plane for semantic segmentation comparison.

Unlike methods that focus on rendering high-quality RGB images for novel view synthesis, we do not perform direct RGB synthesis. Instead, we use differentiable rendering as a bridge between 3D and 2D supervision. By projecting learned 3D semantic features onto the 2D image plane, the soft cross-entropy loss is computed against pseudo-labeled ground truth segmentation maps. This enables supervision of 3D semantic occupancy prediction without requiring dense 3D annotations.

4.1.2 Spherical Rendering

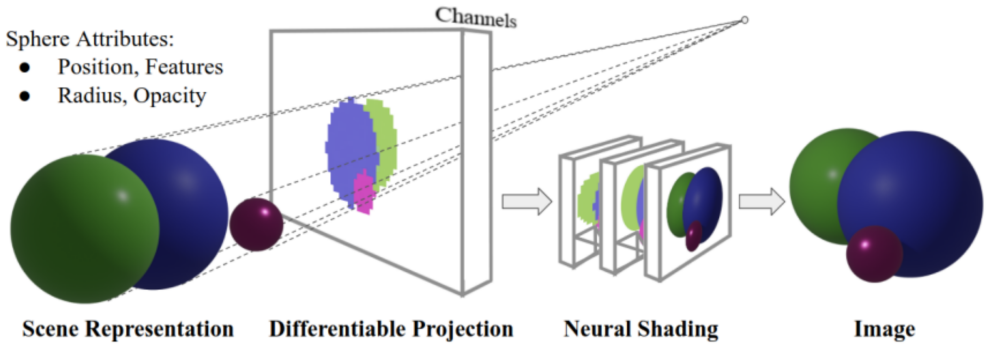


Figure 4.2: Pulsar's [78] spherical rendering pipeline. (Source from [78]).

Spherical rendering extends point-based methods by assigning a radius to each point, effectively treating them as spheres. This additional spatial structure pro-

vides a better approximation of object surfaces without requiring explicit connectivity. Each sphere is defined by its 3D position, radius, opacity, and attributes such as color or semantic feature vectors. When projected onto the image plane, each sphere forms a circular footprint, whose contribution to the final image is determined via blending functions [78].

Compared to standard point rendering, spherical rendering provides a more stable gradient signal by preventing aliasing effects associated with infinitely small points. Moreover, it mitigates sparsity issues common in raw point clouds by filling gaps more effectively. Unlike meshes, spheres inherently support dynamic topologies without requiring complex connectivity updates. In addition, many existing point-based renderers limit the number of points contributing to each pixel by selecting only the first few closest points [78]. This approach often leads to high-frequency artifacts in complex scenes due to the abrupt exclusion of relevant contributions. In spherical rendering, a theoretical unlimited number of spheres are able to contribute to the final pixel feature, controlled by a blending parameter γ. In reality however, a minimum contribution threshold should be set as an optional constraint to manage computational efficiency. In any case, γ allows only the most relevant spheres to be selected for gradient propagation, preventing instability while maintaining fidelity in feature projection.

For semantic occupancy prediction, spheres facilitate smooth and differentiable projections of learned 3D features onto the 2D image space, allowing supervision via soft cross-entropy loss. Since the goal is not to reconstruct highly detailed 3D world models or novel view synthesis, but to enable supervision of semantic occupancy, a balance between accuracy and computational efficiency is necessary. By using spherical primitives, gradients propagate effectively while keeping memory

usage and computational cost manageable.

4.1.3 Comparison to Neural Radiance Fields (NeRF)

Neural Radiance Fields (NeRF) represent a scene as a continuous function approximated by a neural network, mapping spatial coordinates and viewing directions to color and opacity [90]. Rendering in NeRF involves querying this function densely along camera rays and aggregating results via volumetric integration.

While NeRF provides high-fidelity view synthesis, its reliance on dense sampling leads to high computational costs and slow inference. Raymarching through a neural network-based scene representation requires thousands of function evaluations per pixel, making real-time performance difficult to achieve [87]. Furthermore, NeRF is primarily designed for rendering detailed RGB images, making it less straightforward to incorporate high-dimensional semantic feature projection, which is necessary for 3D semantic occupancy prediction.

In contrast, spherical rendering provides a discrete representation where each sphere directly contributes to the final projected feature map without requiring implicit function evaluations. This reduces computational requirements while still enabling the supervision of 3D occupancy grids using 2D semantic labels. Unlike NeRF, which requires per-scene optimization and fine-tuned view interpolation, spherical rendering enables direct 3D-to-2D feature projection across training iterations, making it more suitable for applications where differentiable rendering serves as a loss propagation mechanism rather than building a world model.

4.1.4 Comparison to Gaussian Splatting

Gaussian splatting extends point-based rendering by modeling each point as an anisotropic 3D Gaussian, characterized by a mean position, covariance matrix,

and feature vector [91]. Unlike simple point or spherical representations, Gaussian splatting introduces a covariance matrix that defines the spread of each point in space, allowing for elliptical footprints that can better capture fine details and structured surfaces. This anisotropic nature makes it particularly effective at representing complex textures and geometric variations with fewer primitives, as it can adapt its shape dynamically based on local surface orientation.

While Gaussian splatting provides higher accuracy in capturing fine details, its increased expressiveness comes with added training complexity. The optimization of covariance matrices, along with opacity and feature blending, requires additional regularization to prevent degenerate solutions and ensure stability during training. This makes the learning process more demanding compared to isotropic representations such as spherical rendering, where the shape remains uniform and requires fewer learned parameters.

In semantic occupancy prediction, Gaussian splatting do allow for more precise feature projection by adapting its footprint shape to local surface variations. However, this flexibility introduces additional parameters and requires complex regularization strategies. Since the objective is efficient loss propagation through differentiable rendering rather than high-fidelity reconstruction, a simpler and stable representation is preferable. Spherical rendering, with its isotropic structure, provides a better balance between computational efficiency and effective optimization.

4.2 Discussion

In our second contribution, we introduced MinkOcc, a real-time 3D semantic occupancy prediction model that significantly reduces reliance on dense 3D annotations.

By integrating multi-modal sensor fusion (camera + LiDAR) in an adaptive early-fusion pipeline [25], MinkOcc encodes raw sensor data into a sparse feature volume and employs a generative Minkowski UNet [40] for 3D scene completion. Training is guided by a semi-supervised approach that combines (1) dense 3D labels for a small fraction of scenes (warm start phase α) and (2) 2D pseudo-labels from vision-language models (Grounded-DINO [92] + Segment Anything [93]) for the rest of the dataset. This strategy reduces 3D annotation requirements by 90% while maintaining competitive mIoU scores against other semi-supervised approaches.

By leveraging efficient sparse operations and a spherical differentiable rendering head, MinkOcc achieves strong geometric fidelity even in cluttered and occluded urban environments while remaining computationally efficient for real-time deployment. Our experiments on Occ3D-nuScenes validate the effectiveness of this approach, demonstrating that even with the inherent inaccuracy of pseudo-labels, the model's multi-task training schedule encourages robust feature learning. Ablation studies on voxel resolution highlight an optimal trade-off between accuracy and efficiency, with MinkOcc achieving improved accuracy while maintaining moderate GPU memory usage at slightly higher voxel resolution. However, the benefits diminish exponentially as resolution continue to rise, indicating that even finer voxel grids exceed the spatial precision limits of LiDAR-accumulated sweeps.

4.2.1 Open-Vocabulary Mapping

Beyond performance and label efficiency, a further ambition of this work is to accommodate open-vocabulary semantics. On the 2D side, vision-language models such as Grounded-DINO and SAM allow us to rapidly generate pseudo-labels for any arbitrary text prompt, effectively enabling unlimited classes in principle.

However, since Occ3D-nuScenes (and most other autonomous-driving benchmarks) fixes an 18-class semantic taxonomy (including a "free" space class), we must *remap* the open-vocabulary 2D labels to a set of 3D categories. Table 4.1 illustrates these mappings. While this re-labeling step ensures that MinkOcc's output is compatible with existing voxel-level metrics, it also constrains the richness of the open-vocabulary supervision to standard dataset classes.

Table 4.1: Mapping 2D open-vocabulary labels used to prompt Grounded-SAM to the fixed 3D categories in Occ3D-nuScenes.

2D Pseudo-labels	**3D Occ3D-nuScenes Classes**
`background`	`void/other`
`barrier`	`barrier`
`bicycle`	`bicycle`
`bus`	`bus`
`sedan`	`car`
`crane`	`construction vehicle`
`motorcycle`	`motorcycle`
`people`	`pedestrian`
`cone`	`traffic cone`
`truck trailer`	`trailer`
`truck`	`truck`
`highway`	`drivable surface`
`NA`	`other flat`
`sidewalk`	`sidewalk`
`terrain`	`terrain`
`building`, `overhead bridge`, `billboard`, `pole`	`manmande`
`tree`	`vegetation`
`sky`	`free`

As shown in Table 4.1, certain 3D semantic categories, such as `trailer`, `other flat`, and `terrain`, are ill-defined in 2D, resulting in missing mappings `NA`. This limitation arises because some classes inherently depend on spatial or volumetric context that is difficult to infer from a single image. For instance, `other flat` lacks clear boundaries in perspective projections, often appearing visually similar to `sidewalk` and `highway`. The absence of direct 2D supervision for these categories makes it challenging for MinkOcc to learn fine-grained distinctions, potentially leading to ambiguities in its 3D semantic predictions.

This constraint underscores a fundamental challenge in using vision-language models for 3D occupancy prediction: while they enable flexible class definitions in 2D, they lack the volumetric context needed to fully capture the semantics of 3D space. To mitigate this, future work could explore hybrid strategies that supplement 2D semantic pseudo-labels with additional spatial priors. For example, incorporating inferred depth maps from monocular depth estimation networks or using sparse depth cues from LiDAR projections could provide additional geometric context.

In the following discussion chapter, we will present additional illustrative results that highlight MinkOcc's robustness in cluttered city scenes and examine the trade-off between label efficiency and accuracy.

Chapter 5

Discussion

The effectiveness of 3D semantic occupancy prediction is determined by its ability to balance deployment efficiency, robustness, and label efficiency—each of which poses unique challenges that must be addressed. As outlined in the introduction and figure 1.2, these challenges include computational constraints that hinder real-time deployment, environmental variability that impacts model robustness, and the reliance on expensive large-scale annotations that limit generalization. The chapter is structured into three primary sections, each dedicated to a core challenge. Through this structured analysis, we aim to determine the effectiveness of our approach in tackling these fundamental challenges and to provide insight into its remaining limitations and potential future research directions.

5.1 Deployment

5.1.1 Problem Statement

Deploying 3D semantic occupancy prediction models in real-world applications presents significant challenges due to the high computational demands of deep learning frameworks. These models must process large-scale multi-view sensor data efficiently while adhering to strict real-time performance constraints. How-

ever, adapting complex end-to-end deep learning architectures for deployment often leads to excessive memory consumption and difficulties in maintaining inference speed. If not addressed, these deployment inefficiencies can hinder the practical application of 3D perception models in autonomous driving.

5.1.2 Considerations and Criterion

To achieve efficient deployment, models must be optimized for both computational performance and memory efficiency while maintaining high predictive accuracy. One key strategy is leveraging sparse representations and efficient network architectures to minimize redundant computations. Additionally, view transformation techniques should be optimized to accelerate processing while preserving spatial and semantic fidelity. Lightweight network designs, such as pruning and quantization, can further enhance computational efficiency. Moreover, models must be adaptable to embedded on-board devices without significant loss in performance.

Key Criteria:

- **Inference Speed:** Must achieve real-time performance (around 20-30 FPS) to be viable for time-sensitive applications such as autonomous driving.
- **Computational Efficiency:** Models should be designed to minimize GPU/memory consumption while maintaining competitive accuracy.
- **End-to-End Optimization:** The entire inference pipeline should be optimized to reduce latency and reliance on extensive post-processing.
- **Scalability and Hardware Adaptability:** The approach should generalize across different deployment environments and be adaptable to resource-limited embedded devices.

5.1.3 Analysis on Approach

Our approach integrates several computational optimizations to enhance the efficiency and real-time feasibility of 3D semantic occupancy prediction:

- **Sparse Convolution (Minkowski Engine [40])**: Reduces computational overhead by only processing non-empty voxels, improving efficiency over dense 3D CNNs.
- **Early Camera-LiDAR Fusion**: Uses a simple 2D-to-3D transformation via forward projection and bilinear interpolation, avoiding expensive cross-attention or raymarching mechanisms.
- **Multi-task Learning**: Simultaneously predicts semantic segmentation and scene completion, reducing redundant computations.
- **Spherical Differentiable Rendering (Pulsar [78])**: The use of spherical primitives in our second contribution ensures efficient memory usage while maintaining accurate feature projection.

First Approach: Performance Metrics

Our first approach is designed for monocular front-camera input, differing from existing multi-view benchmarks. To ensure a fair comparison, inference speed is measured on the Occ3D-nuScenes [66] dataset with a batch size of 6 to match the number of cameras as in multi-view methods. The model is evaluated on an NVIDIA RTX 4090 GPU.

Computational Efficiency and Inference Speed:

- GPU Memory Usage: 1.2GB VRAM.
- Inference Speed: 33.1 seconds per frame, achieving 30 FPS.

Second Approach: Performance Metrics

The second approach prioritizes improved accuracy at the cost of a reduction in inference speed. Key changes include an optimized multi-modal fusion strategy that better processes LiDAR and camera features. In this analysis, we evaluate our model which is trained with strong supervision for a direct comparison. The model is evaluated on an NVIDIA RTX 4090 GPU.

Computational Efficiency and Inference Speed:

- GPU Memory Usage: 1.65GB VRAM
- Inference Speed: 43.5 milliseconds per frame, translating to 23 FPS.

Comparison of Approaches

Table 5.1 compares our approaches with state-of-the-art methods in terms of inference speed, GPU usage, and accuracy.

Table 5.1: Deployment Feasibility Comparison

Approach	**Inference Speed (s)**	**FPS**	**GPU Usage (GB)**	**mIoU (%)**
BEVFormer	0.31	3.2	4.5	23.70
SurroundOcc	0.34	2.9	5.9	38.69
FastOcc	0.22	4.5	-	40.75
FastOcc (TRT)	0.08	12.5	-	40.75
Ours (First Approach)	0.03	30	1.2	36.03
Ours (Second Approach)	0.04	23	1.6	44.85

Limitations

While our approach achieves real-time inference on high-end GPUs, it has not been extensively tested on embedded or low-power hardware. Future work includes developing a streamlined inference pipeline optimized for deployment on edge devices using pre-trained weights.

5.2 Robustness

5.2.1 Problem Statement

Robustness in 3D semantic occupancy prediction is crucial for ensuring consistent and reliable performance under diverse real-world conditions. Environmental factors such as changing lighting conditions, adverse weather (e.g., fog, rain), and varying geographical terrains introduce significant distribution shifts that can degrade model accuracy. If models fail to generalize across these variations, they may produce unreliable occupancy estimates.

5.2.2 Considerations and Criterion

To develop robust 3D semantic occupancy models, it is necessary to design architectures that can effectively generalize across different environmental conditions. One approach is to leverage multi-modal sensor fusion, where complementary information from different sensor types (e.g., radar, LiDAR, cameras) enhances resilience to environmental variations. Additionally, robust scene representations, such as learned domain-invariant features, can help mitigate the effects of distribution shifts. Architectural improvements, including self-supervised learning techniques and uncertainty-aware models, can further enhance robustness.

Key Criteria:

- **Generalization Across Environments:** Models should maintain performance across different geographical locations.
- **Weather Adaptability:** Must be resilient to environmental factors like fog, rain, and extreme lighting conditions.
- **Consistency Under Variability:** Predictions should remain stable across

different times of the day and dynamic scenarios.

- **Scene Representation Robustness:** Models should handle variations in input data quality without significant performance degradation.

5.2.3 Analysis on Approach

To make sure our model is robust in 3D semantic occupancy prediction, we integrate several key design choices:

- **Multi-modal Sensor Fusion:** We extract LiDAR and camera features independently to preserve the unique characteristics of each modality—LiDAR for precise depth, and cameras for semantic information. These features are then fused in 3D space using adaptive deep point fusion [25].

- **2D Pseudo-label Supervision with 3D Semantic Occupancy as Priority:** Our model employs a multi-task learning framework, where 2D pseudo-labels from image-based semantic segmentation act as auxiliary supervision. This improves semantic consistency between 2D perspective space and 3D voxel space, ensuring that the 3D voxel representation retains coherent semantic definitions across different viewpoints.

- **Training Augmentation Strategies:** To improve robustness against distribution shifts, we apply augmentations across both modalities. 2D image augmentations include rotation, scaling, and translation, while 3D LiDAR augmentations involve random flipping, rotation, and ground truth dropout.

To assess the robustness of our approach, we perform qualitative and quantitative analyses guided by the aforementioned key criteria, selecting diverse scenes

that capture geographical variations, weather conditions, sensor failures, and challenging environments. For each scene, we present the front camera image for visual context, the LiDAR point cloud for spatial reference, the predicted 3D semantic occupancy, and the corresponding 3D ground truth for comparison. We further quantify performance using per-scene mean Intersection over Union (mIoU). We use MinkOcc-strong from our second contribution trained with $\alpha = 100$ (strong supervision) and a voxel size of $0.4m$. Based on the Occ3D-nuScenes validation set, we select scenes that align with a combination of key criteria.

Table 5.2: Scene Condition Variations

Attribute	**Possible Values**
Location	Boston / Singapore
Time of Day	Night / Day
Weather	Sunny / Rainy / Cloudy
Sensor Condition	Camera degraded / LiDAR degraded / Both sensors operational

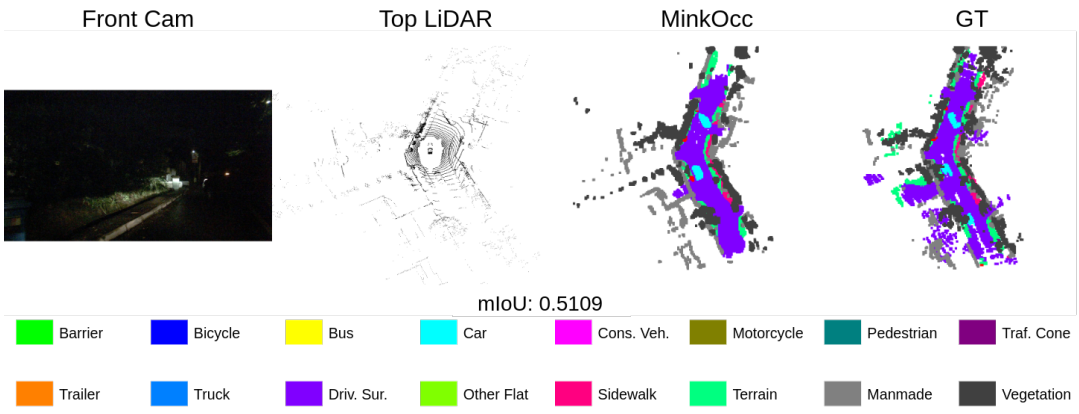


Figure 5.1: Singapore, night, cloudy weather, degraded camera. Scene includes a narrow alleyway with a reversing vehicle. The model accurately detects the reversing vehicle and road boundaries despite adverse sensor conditions. **Better viewed when zoomed in.**

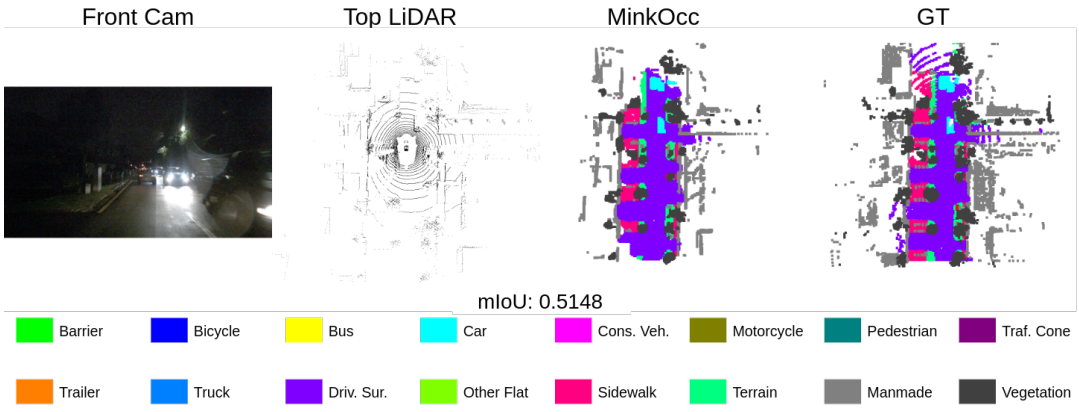


Figure 5.2: Singapore, night, cloudy weather, degraded camera. Oncoming traffic with intense headlights presents challenging lighting conditions. The model successfully identifies oncoming vehicles and road boundaries.

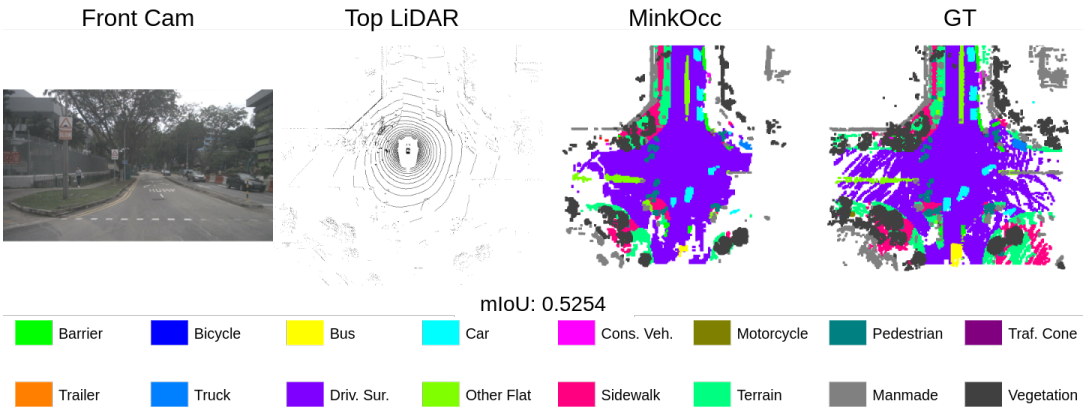


Figure 5.3: Singapore, daytime, overcast, both sensors operational. Scene depicts a busy junction with pedestrians and multiple turning vehicles. The model captures pedestrians, turning vehicles, and road dividers, though with minor inaccuracies.

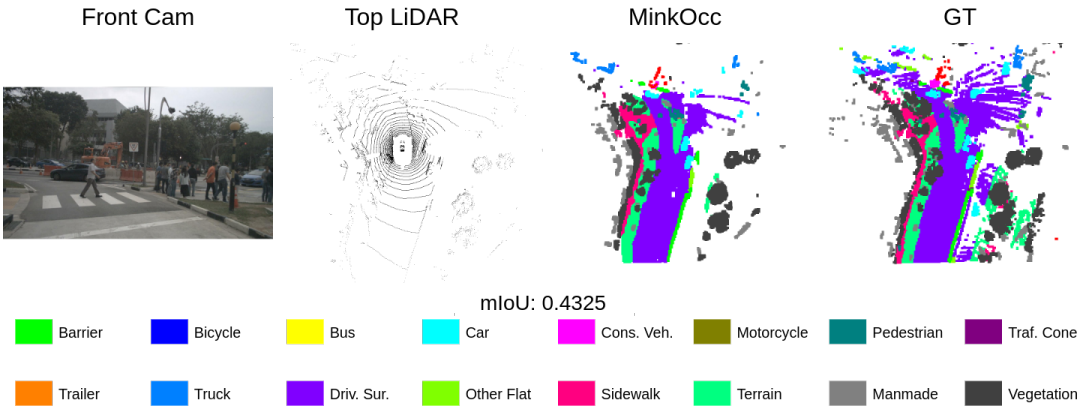


Figure 5.4: Singapore, daytime, overcast, both sensors operational. Scene features a left turn lane and traffic island with a zebra crossing. The model detects crossing pedestrians and road boundaries effectively, but struggles with distant barriers and construction vehicles.

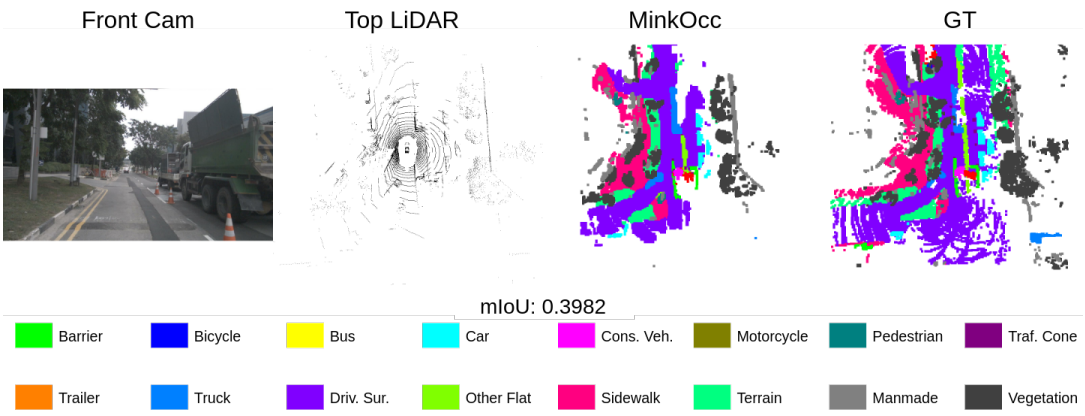


Figure 5.5: Singapore, daytime, overcast, both sensors operational. Scene exhibits temporary roadworks with marked traffic cones and a stationary truck. While the model identifies the truck accurately, it misses some traffic cones.

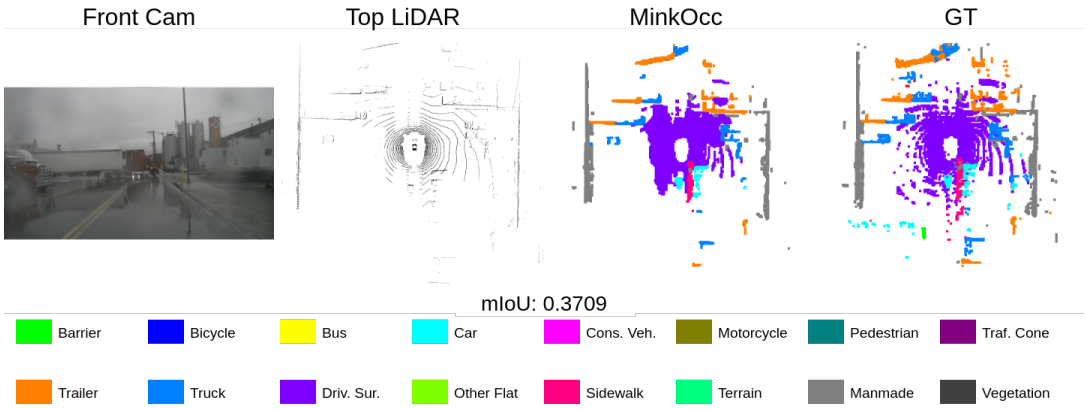


Figure 5.6: Boston, daytime, rainy conditions with a partially obscured camera and degraded LiDAR. Sparse ground-truth annotations. A reversing truck and trailer block the lane. The model detects nearby vehicles but overall scene completion remains sparse.

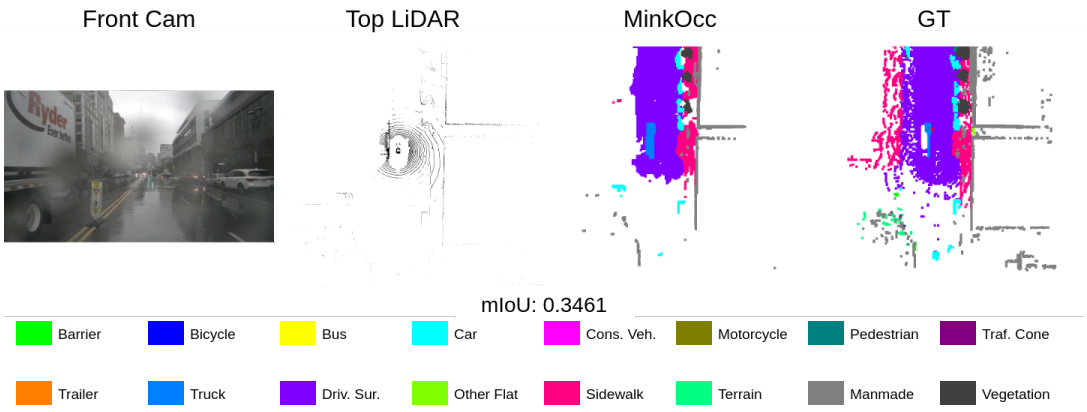


Figure 5.7: Boston, daytime, rainy conditions with water droplets on the camera and degraded LiDAR. Scene includes a vehicle close to the road divider, an oncoming truck, and parked cars. The system detects most vehicles but overall scene completion remains sparse.

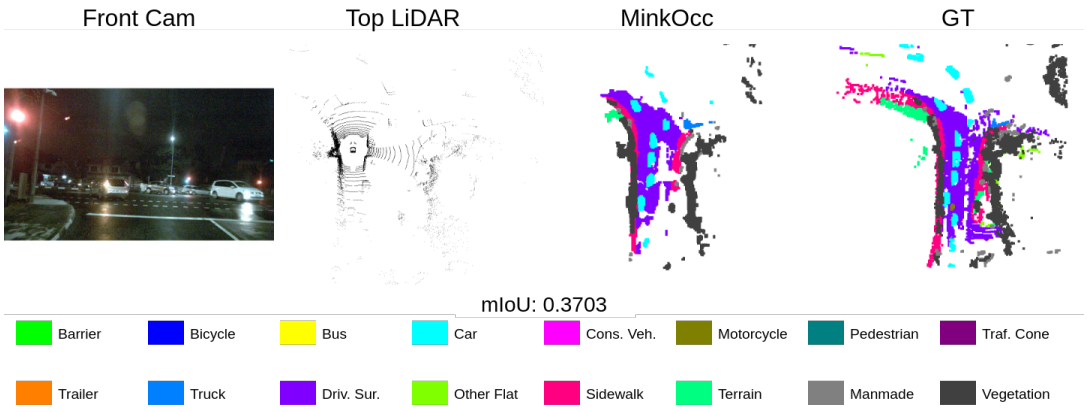


Figure 5.8: Singapore, nighttime, wet conditions. Camera is functional with slight distortion; LiDAR suffers from heavy occlusions. Scene shows a busy intersection with multiple turning vehicles. The model detects nearby vehicles but misses distant traffic and drivable surfaces.

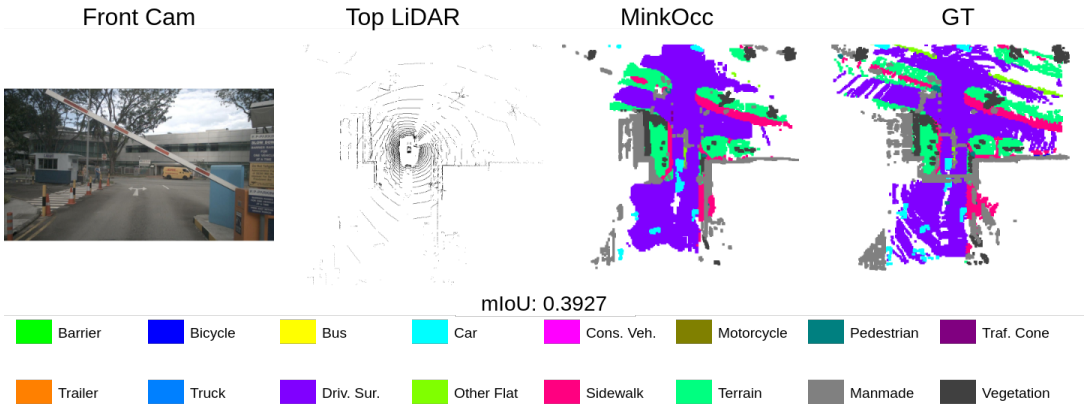


Figure 5.9: Singapore, daytime, overcast, both sensors operational. Scene features a gantry entrance with a pedestrian and surrounding vehicles. The system detects all key actors and boundaries accurately.

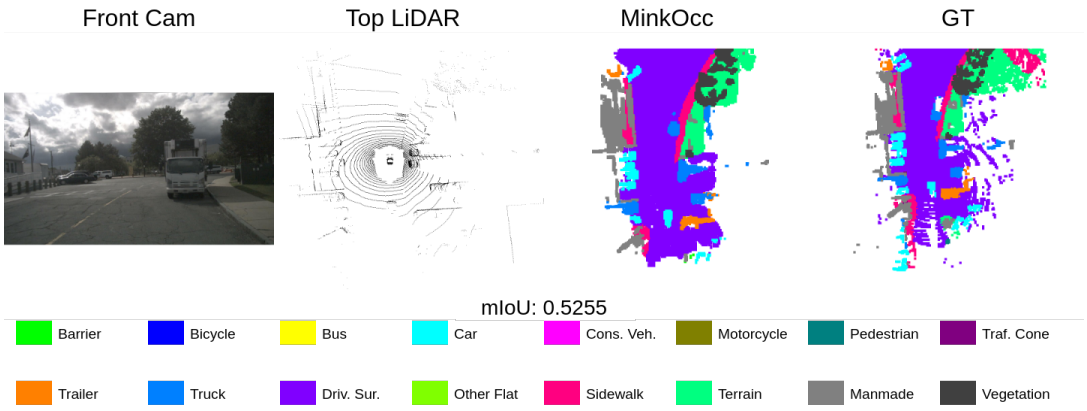


Figure 5.10: Boston, daytime, overcast, both sensors operational. A stationary truck partially blocks the lane, necessitating the ego-vehicle to enter the oncoming lane. The model identifies the truck, road boundaries, and surrounding vehicles.

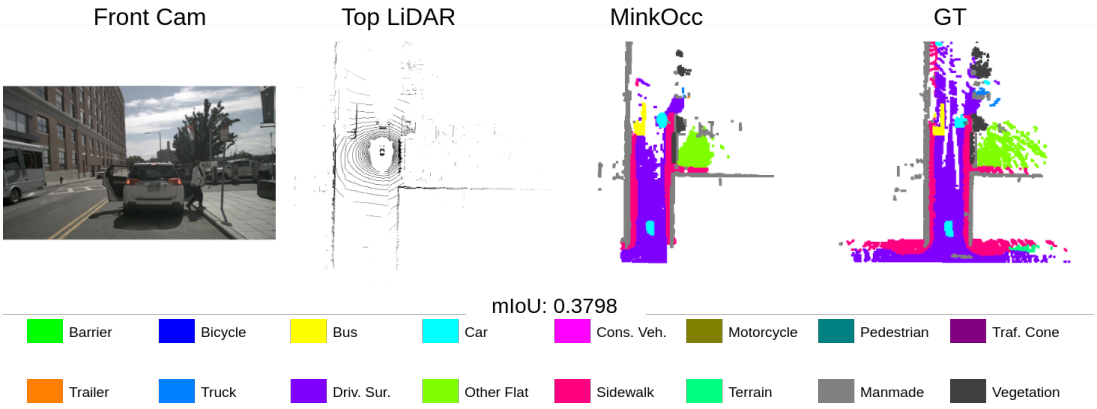


Figure 5.11: Boston, daytime, sunny, both sensors operational. Scene shows passengers exiting a parked car on a one-lane road with an oncoming bus. The model detects the bus and parked car but struggles with pedestrians and drivable surface completion.

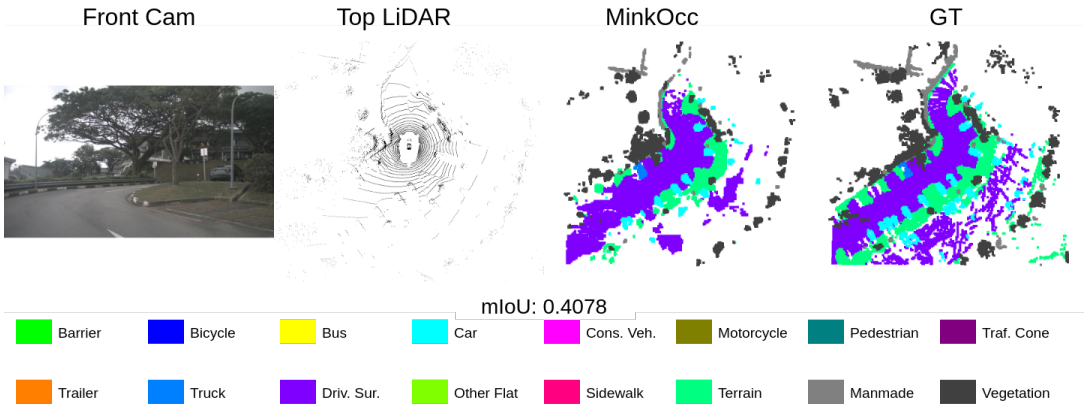


Figure 5.12: Singapore, daytime, overcast, both sensors operational. Scene highlights an uphill carpark with winding roads. The model detects parked cars and drivable surfaces downhill but lacks resolution for uphill geometry.

Limitations

Despite demonstrating resilience under various environmental and sensor conditions (figures 5.1–5.12), our approach exhibits several notable limitations. First, small or distant objects often remain challenging to detect, as they are prone to under-segmentation or misclassification. This issue is exacerbated by occlusions in dense urban areas and degraded sensor data (e.g., raindrops on camera lenses or reduced LiDAR returns in adverse weather). Additionally, when both camera and LiDAR signals are degraded—such as in heavy rain or night-time scenes with minimal illumination—our model's confidence in semantic occupancy can drop significantly, indicating limited robustness to extreme sensor failures.

A second limitation arises from the fact that our experiments predominantly rely on the Occ3D-nuScenes dataset, which, while diverse, does not fully represent global variations in road infrastructure, signage, or extreme weather. Consequently, the model's ability to generalize to entirely different geographies (e.g., heavy snowfall, unpaved roads) has not been evaluated. Future work could address these gaps by incorporating larger, more geographically varied datasets.

5.3 Label-efficient

5.3.1 Problem Statement

A significant challenge in 3D semantic occupancy prediction is the heavy reliance on large-scale, high-quality annotated data. Manually labeling 3D scenes is not only expensive but also labor-intensive, making it impractical for real-world applications. Without addressing label efficiency, the deployment of 3D perception systems remains constrained, as models struggle to adapt without extensive annotation efforts.

5.3.2 Considerations and Criterion

To overcome this, label-efficient methods must be developed. One promising direction is weak supervision, where models learn from partial ground truth labels or noisy pseudo-labels while still achieving competitive performance. Self-supervised learning can potentially leverage large-scale, unlabeled datasets to extract meaningful representations without explicit human annotations. Semi-supervised learning provides another viable approach, where models are trained on a combination of labeled and unlabeled data. A key consideration in label-efficient approaches is to maintain a reasonable level of prediction accuracy while reducing annotation costs.

Key Criteria:

- **Reduction in Label Dependence:** Models should achieve strong performance with minimal labeled data.
- **Weak Supervision Performance:** Ability to learn effectively under weakly-supervised or self-supervised paradigms.

- **Pseudo-Label Quality:** Generated pseudo-labels should be reliable enough to improve generalization on new scenes.
- **Scalability to Unlabeled Data:** Models should effectively leverage large-scale unlabeled datasets for learning.

5.3.3 Analysis on Approach

To improve label efficiency while maintaining strong semantic occupancy performance, we integrate key design choices that reduce dependence on dense 3D annotations:

- **Spherical Differentiable Rendering [78] with 2D Pseudo-Labels:** We use this to align 2D pseudo-labels with 3D occupancy predictions, allowing semantic supervision signals from 2D perspective view to transfer into 3D cartesian space.
- **LiDAR Accumulation for Sparse 3D Ground Truth:** Instead of relying on manually annotated dense 3D labels, we use LiDAR sweep accumulation as a weak supervision signal for occupancy scene completion. This provides a sparse but effective occupancy representation.

Similar to our robustness analysis, we conduct both qualitative and quantitative evaluations, selecting diverse scenes to assess weakly supervised semantic occupancy prediction. For each scene, we present all three variations of MinkOcc, differentiated by the warm-start phase α, which determines the percentage of dense 3D annotations available from the Occ3D-nuScenes dataset. The corresponding 3D ground truth is provided for direct comparison.

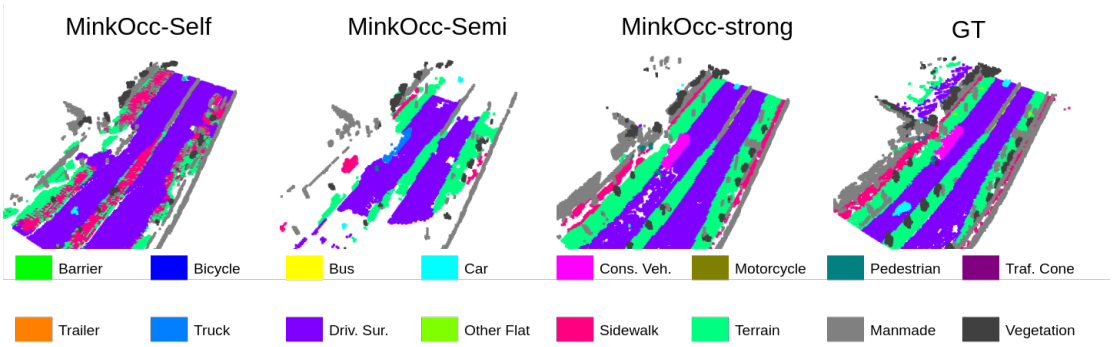


Figure 5.13: **Scene:** Single-lane road with a parked construction vehicle, a pedestrian, and a traffic cone in the distance.
MinkOcc-strong: Accurately reconstructs most objects but omits the distant traffic cone.
MinkOcc-semi: Detects the pedestrian but mislabels the construction vehicle as a truck and fails to fill distant drivable surface.
MinkOcc-self: Labels both pedestrian and construction vehicle as manmade yet achieves reasonable road completion.

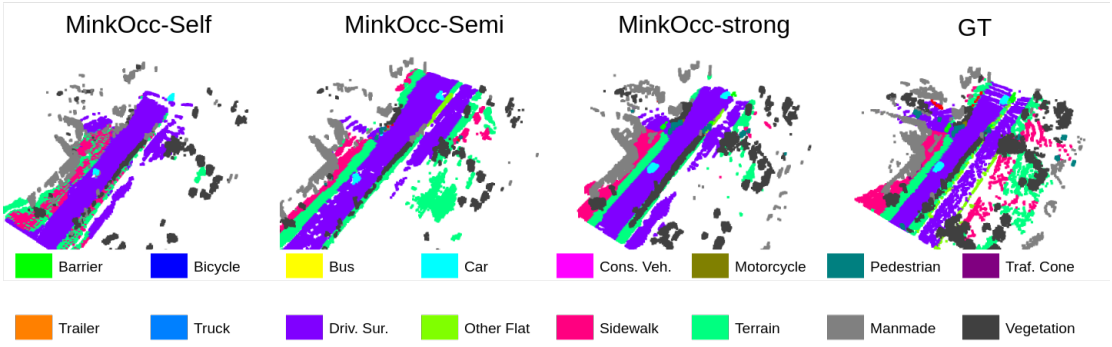


Figure 5.14: **Scene:** T-junction with pedestrians on the drivable surface and vehicles positioned ahead and behind.
MinkOcc-strong: Correctly identifies pedestrians and vehicles.
MinkOcc-semi: Partially detects pedestrians, mislabeling some as manmade. Vehicles are well-predicted.
MinkOcc-self: Misclassifies most pedestrians as manmade, but still completes the drivable surface and nearby vehicles.

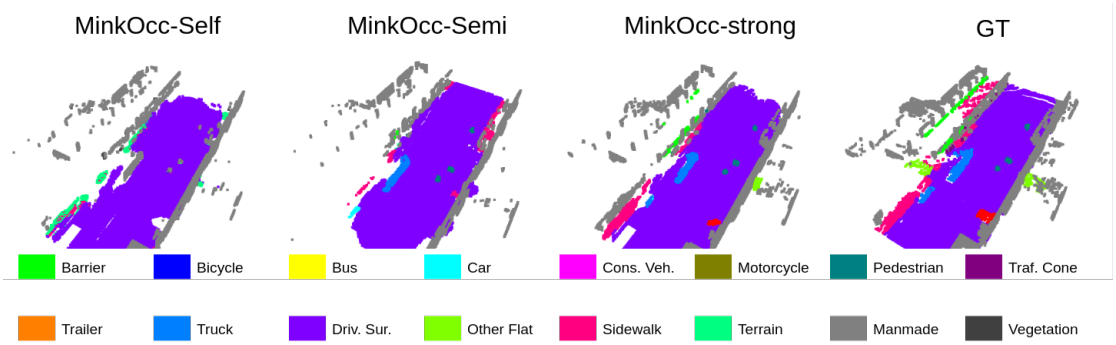


Figure 5.15: **Scene:** Road without a divider, with a truck on the left and three pedestrians on the right.
MinkOcc-strong: Accurately detects the truck and pedestrians.
MinkOcc-semi: Matches MinkOcc-strong performance overall, capturing both truck and pedestrians.
MinkOcc-self: Labels the truck and pedestrians as manmade, and under-completes distant drivable surfaces.

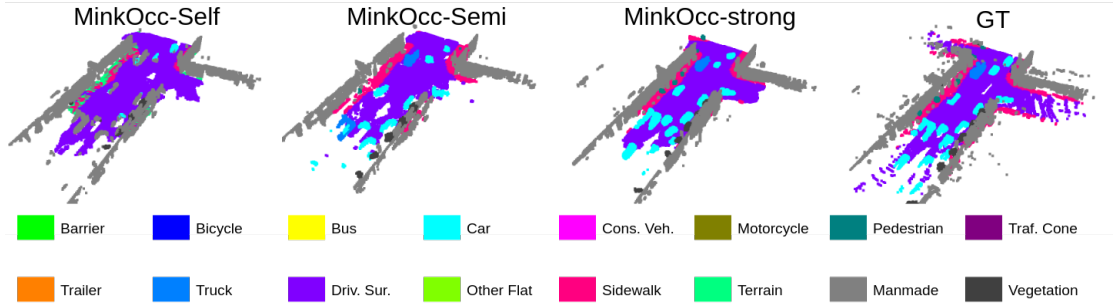


Figure 5.16: **Scene:** Busy T-junction with multiple cars, trucks, and sidewalk pedestrians.
MinkOcc-strong: Identifies most vehicles and pedestrians but leaves occluded drivable surface incomplete.
MinkOcc-semi: Misses some cars and pedestrian labels. Incomplete drivable surface.
MinkOcc-self: Heavily mislabels vehicles as manmade, detecting few pedestrians. Incomplete drivable surface.

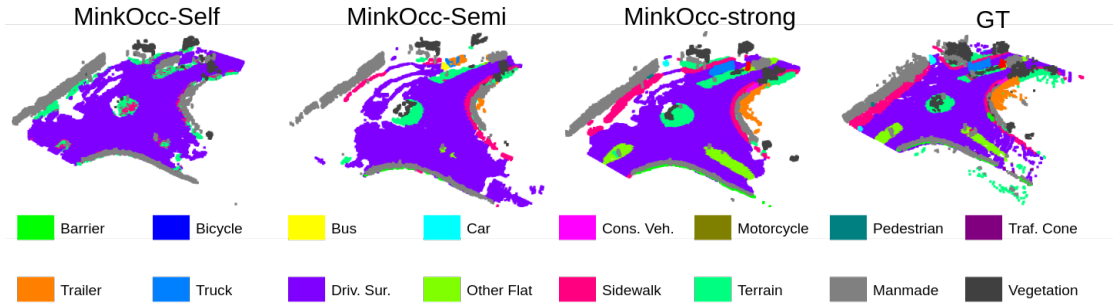


Figure 5.17: **Scene:** Roundabout with road dividers and an oncoming truck.
MinkOcc-strong: Reconstructs the roundabout structure and truck accurately.
MinkOcc-semi: Partially visualizes the roundabout, confusing the truck for a bus or trailer.
MinkOcc-self: Mislabels the truck as manmade but shows slightly more drivable surface overall.

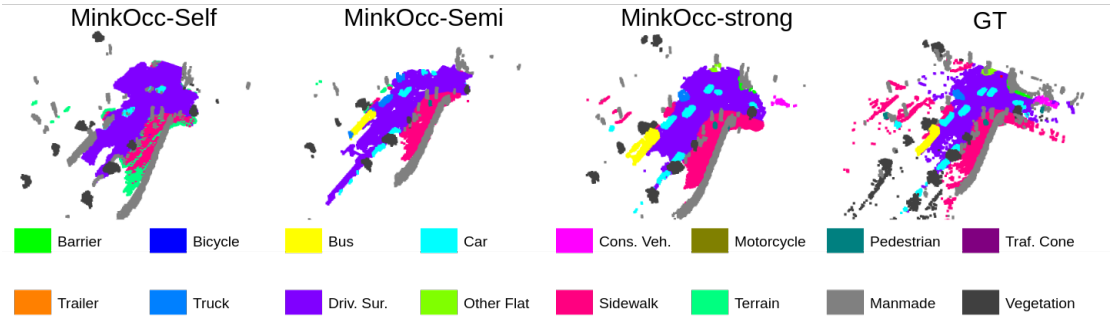


Figure 5.18: **Scene:** T-junction with numerous turning vehicles and heavy occlusion.
MinkOcc-strong: Predicts all vehicles with partial scene reconstruction.
MinkOcc-semi: Covers most vehicles, offering less completeness than strong supervision.
MinkOcc-self: Identifies fewer vehicles but sometimes reconstructs more drivable surface.

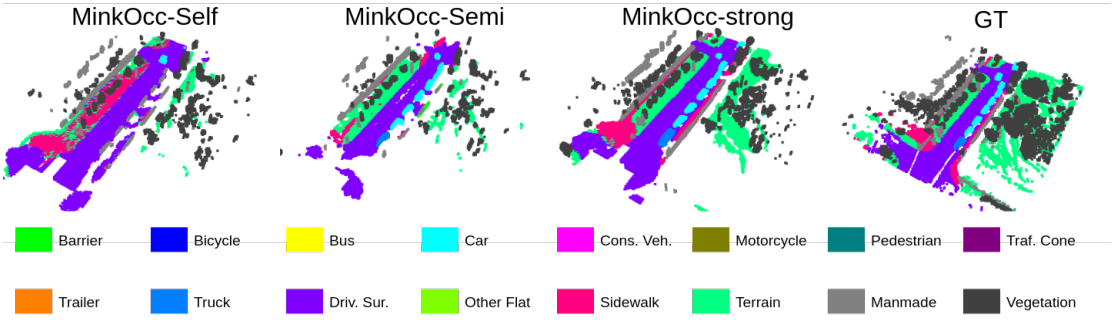


Figure 5.19: **Scene:** Straight roadway with parked cars on one side and clear boundaries.
MinkOcc-strong: Accurately models vehicles and road edges.
MinkOcc-semi: Matches strong supervision for key objects yet underperforms in occluded regions behind parked cars.
MinkOcc-self: Classifies most vehicles as manmade, though road boundaries remain well-defined.

In addition to assessing MinkOcc's 3D occupancy predictions, we also examine its 2D semantic renderings against the pseudo-labels used during training. Figures 5.20–5.22 highlight how effectively the model's learned 3D features align with pseudo-supervision in 2D perspective space. This evaluation provides insight into both the quality of the pseudo-labels and the model's ability to refine or extrapolate beyond potentially incomplete or noisy 2D annotations.

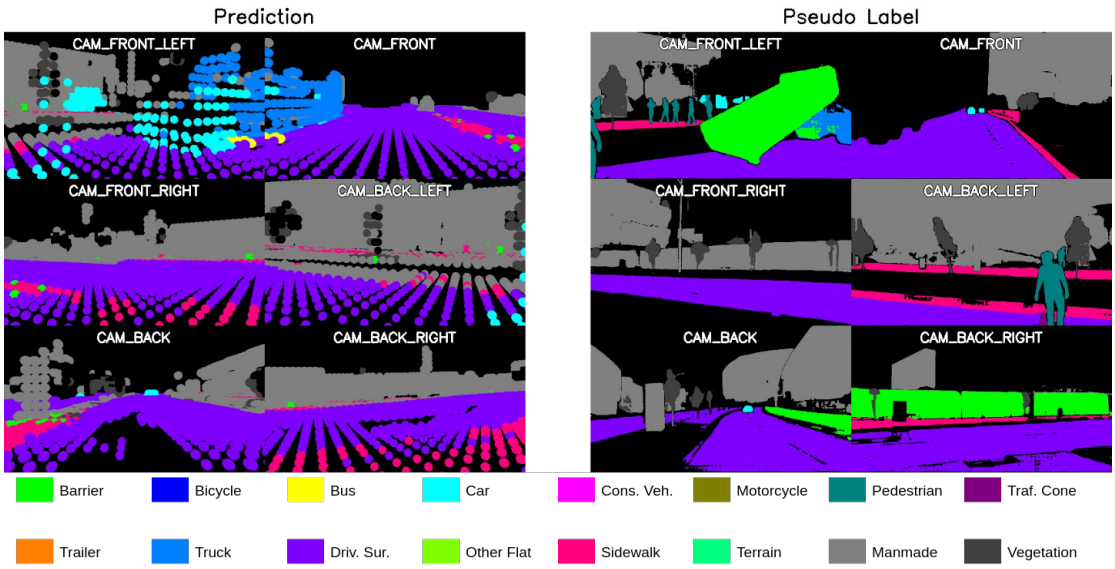


Figure 5.20: **Pseudo-Label Quality:** Labels the trailer as a barrier and omits some sidewalk and terrain regions, but identifies pedestrians.
2D Prediction Quality: Accurately captures drivable surfaces and road boundaries, although it misclassifies pedestrians as cars and mislabels distant vehicles as manmade.

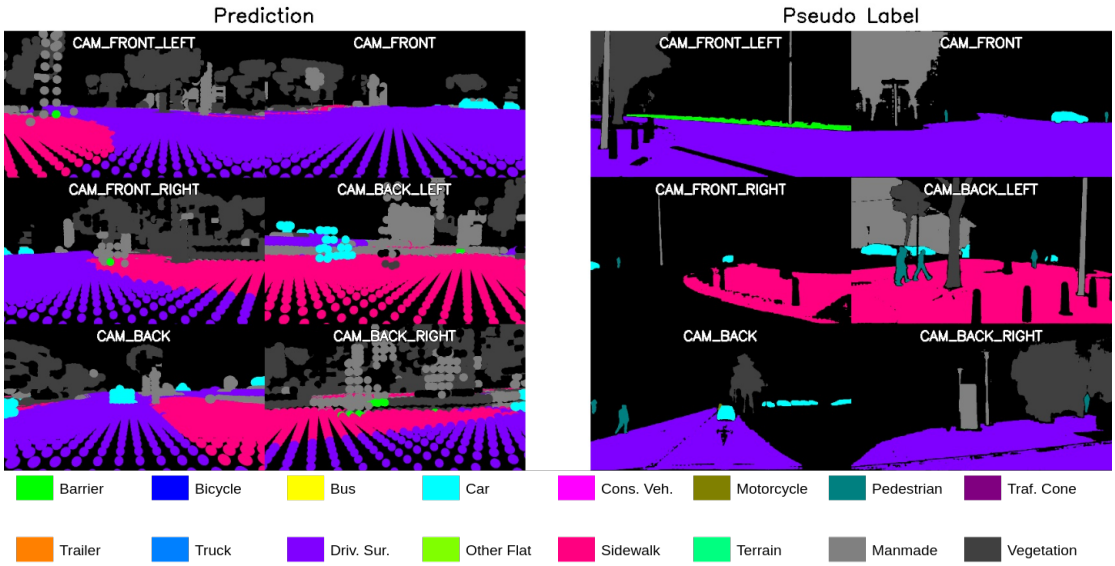


Figure 5.21: **Pseudo-Label Quality:** Fails to annotate drivable surfaces in CAM_FRONT_RIGHT and omits certain manmade, sidewalk, and terrain elements.
2D Prediction Quality: Correctly classifies most cars and clearly marks road boundaries and sidewalks, though it confuses some pedestrians with cars.

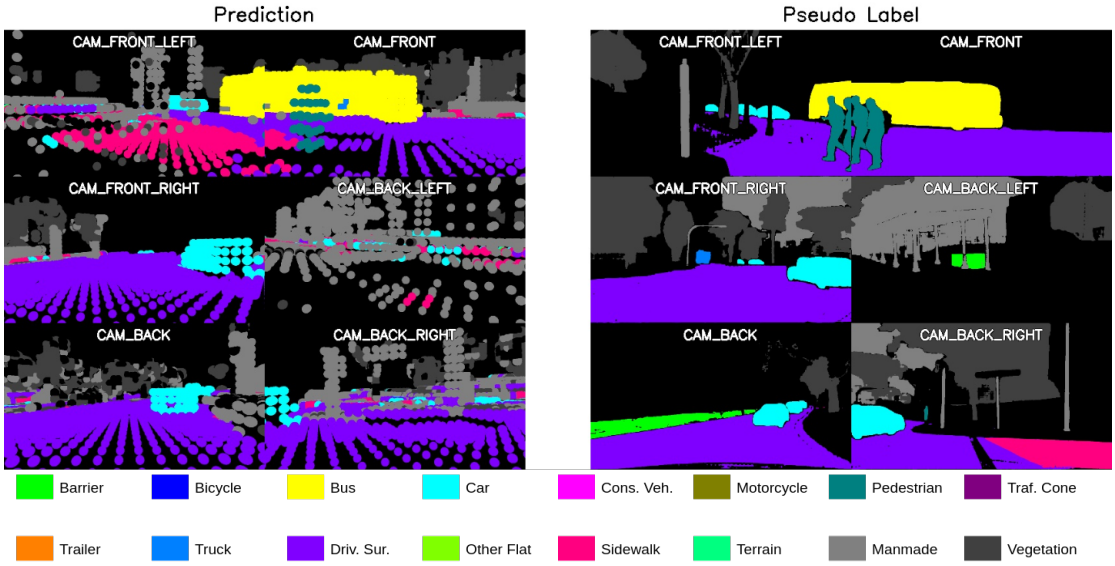


Figure 5.22: **Pseudo-Label Quality:** Shows pedestrians and vehicles clearly, but misses sidewalk/manmade areas in CAM_BACK_LEFT and CAM_FRONT_LEFT.
2D Prediction Quality: Accurately detects drivable surfaces, pedestrians, buses, and cars. In unlabeled regions, the model's predictions appear inconsistent.

Limitations

In the semi-supervised regime, our experiments show that MinkOcc-semi reduce reliance on dense 3D annotations by 90% while still detecting a considerable amount of vehicles, objects and road infrastructure. However, detecting very small or highly occluded objects remains challenging. In particular, misclassifications and incomplete scene completion are common in cluttered urban scenarios, suggesting a need for further refinement and more supervision cues to handle these edge cases. At the fully self-supervised regime, our approach removes the need for explicit 3D labels entirely. However, it recognizes only larger and more distinct structures (e.g., road boundaries, buildings, major road surfaces), while smaller objects and peripheral areas are often misclassified or omitted.

We find that the quality of 2D pseudo-labels directly influences learned 3D feature representation. Systematic errors or omissions (e.g., missing traffic cones or distant pedestrians) can propagate into 3D predictions, leading to incomplete scene coverage. Additionally, the voxel-to-point-to-sphere transformation before spherical differentiable rendering introduces information loss by collapsing each voxel into a single point, discarding some semantic details. This results in fewer rendered spheres, limiting the model's ability to align predictions with pseudo-labels. Strategies to improve label generation and the voxel-to-sphere transformation pipeline could substantially enhance both robustness and fine-grained accuracy. Finally, these label-efficient models were evaluated primarily on Occ3D-nuScenes. Future work should explore additional large-scale datasets to validate the performance of MinkOcc-semi and MinkOcc-self.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis presents a comprehensive approach to 3D semantic occupancy prediction for autonomous driving, integrating multiple strategies to achieve accurate spatial understanding of complex road environments. The primary goal is to create a deep learning framework that operates in real time, reliably handles diverse and challenging driving conditions, and avoids the prohibitive cost of large-scale 3D annotations.

To enable real-time deployment, the first contribution introduces a memory-efficient Minkowski Engine-based sparse convolution design that eliminates extensive 2D-to-3D view transformations and post-processing modules commonly found in existing 3D occupancy pipelines. Rather than processing fully dense voxel grids with many unnecessary features, the approach selectively operates on important non-empty regions in 3D space, significantly reducing computational overhead while preserving high-resolution detail. This architecture achieves low-latency inference, making it well suited for time-critical perception in autonomous driving. Experimental evaluations confirm that the proposed method maintains

real-time performance without sacrificing semantic accuracy, ensuring its feasibility for onboard deployment.

Building on this foundation, the second contribution, MinkOcc, enhances 3D semantic occupancy prediction by improving multi-modal sensor fusion and reducing reliance on dense 3D labels. In terms of sensor fusion, MinkOcc performs LiDAR and camera feature extraction first, before fusing them together using adaptive deep point fusion. For label efficiency, MinkOcc incorporates spherical differentiable rendering, aligning 2D image space with the 3D voxel space. Together with vision-language model generated 2D pseudo-labels, this allows 3D voxels to be projected to camera images and be trained. Evaluations across different supervision levels confirm that MinkOcc maintains strong semantic accuracy while significantly lowering annotation costs. Although full self-supervision only predicts road structures and struggles with fine details, the semi-supervised approach is able to classify key semantic classes.

In depth analysis demonstrates that our 3D semantic occupancy prediction framework can achieve both computational efficiency and robustness while significantly reducing reliance on dense 3D annotations. The proposed methods establish a scalable framework for 3D scene understanding, paving the way for future advancements such as open-set semantic recognition and self-supervised strategies to further minimize annotation costs and enhance generalization across diverse autonomous driving environments.

6.2 Future Work

6.2.1 Improving Weak-supervision

The self-supervised (MinkOcc-self) and semi-supervised (MinkOcc-semi) models still face challenges in semantic and geometric reconstruction. These models exhibit difficulties in segmenting small objects, completing occluded regions, and maintaining consistent semantic predictions, particularly in cluttered and rainy urban environments. A key limitation is the dependency on 2D pseudo-labels, which, while providing broad semantic guidance, lack precise depth and structural information. To address these shortcomings, additional supervision signals should be incorporated.

One approach is the integration of 2D depth maps as an auxiliary supervision signal. Recent advances in monocular depth estimation have demonstrated highly accurate depth predictions in metric space [94], which can complement the accumulated LiDAR sweeps for occupancy scene completion. Another direction is to use 2D camera feature maps from pre-trained feature extraction models. Vision-language models have been shown to extract strong contextual features from large-scale datasets [95]. Aligning these feature maps with the 3D voxel space via differentiable rendering can potentially provide additional supervision, particularly in cases where current 2D semantic pseudo-labels are noisy or incomplete.

Further refinements in the current differentiable rendering procedure also will improve weakly-supervised 3D occupancy predictions, particularly in the voxel-to-point-sphere transformation. Methods such as direct voxel rendering [96] or creating a lossless voxel-to-sphere transformation can be implemented to mitigate information loss during this process.

6.2.2 Other Datasets

The framework of MinkOcc has been validated primarily on Occ3D-nuScenes, an extension of the nuScenes dataset that provides dense 3D semantic occupancy annotations. Occ3D-nuScenes has been critical in assessing the performance of MinkOcc under diverse driving conditions with extensive multi-modal sensor data. However, to further demonstrate the generalisatibility and robustness of the framework, it is necessary to evaluate it on additional datasets featuring different sensor configurations and environmental conditions.

Several autonomous vehicle datasets offer diverse testing grounds due to variations in sensor setups, annotation densities, and scene complexities. For instance, while nuScenes [1], Argoverse 2 [67], and Waymo [64] provide complementary urban driving perspectives, predominantly from the US and Singapore, datasets such as Apolloscape [97] and SemanticKITTI [71] contribute further insights into performance across different geographic regions.

Beyond these traditional AV datasets, emerging datasets like the Oxford RobotCycle Project [98] present a unique opportunity for further evaluation. The RobotCycle dataset is collected using a backpack-mounted sensor suite that includes inertial, range, and visual sensors, as well as eye-tracking data. Although its primary focus is on cycling safety, the dataset captures dynamic, on-road data that is directly relevant to 3D semantic occupancy prediction. Evaluating MinkOcc on the RobotCycle dataset will test its adaptability to non-vehicle sensor configurations and provide insights into its performance under varied urban conditions. Such evaluations are expected to reveal additional refinement needs, particularly in addressing domain shifts and sensor-specific variations, thereby broadening the framework's applicability in real-world traffic environments.

Bibliography

[1] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 621–11 631.

[2] H. Xu, J. Chen, S. Meng, Y. Wang, and L.-P. Chau, "A survey on occupancy perception for autonomous driving: The information fusion perspective," *arXiv preprint*, vol. arXiv:2405.05173, 2024.

[3] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Nature, 2022.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[5] L. Juan and O. Gwun, "A comparison of sift, pca-sift and surf," *International Journal of Image Processing (IJIP)*, vol. 3, no. 4, pp. 143–152, 2009.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, 2012.

[7] K. O'Shea, "An introduction to convolutional neural networks," *arXiv preprint*, vol. arXiv:1511.08458, 2015.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[12] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, "Scaling vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[13] A. Sampath and J. Shan, "Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 3, pp. 1554–1567, 2010.

[14] C.-H. Lin, J.-Y. Chen, P.-L. Su, and C.-H. Chen, "Eigen-feature analysis of weighted covariance matrices for LiDAR point cloud classification," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 94, pp. 70–79, 2014.

[15] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 3212–3217.

[16] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4338–4364, 2021.

[17] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[18] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[19] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, "Searching efficient 3D architectures with sparse point-voxel convolution," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 685–702.

[20] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 16 239–16 248.

[21] A. W. Harley, Z. Fang, J. Li, R. Ambrus, and K. Fragkiadaki, "Simple-BEV: What really matters for multi-sensor BEV perception?" in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 2759–2765.

[22] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai, "BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[23] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3D semantic occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 9223–9232.

[24] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal voxelnet for 3D object detection," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7276–7282.

[25] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 244–253.

[26] Q. Ma, X. Tan, Y. Qu, L. Ma, Z. Zhang, and Y. Xie, "COTR: Compact occupancy transformer for vision-based 3D occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[27] C. Sima, W. Tong, T. Wang, L. Chen, S. Wu, H. Deng, Y. Gu, L. Lu, P. Luo, D. Lin, and H. Li, "Scene as occupancy," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 8372–8381.

[28] Z. Ye, T. Jiang, C. Xu, Y. Li, and H. Zhao, "Cvt-occ: Cost volume temporal fusion for 3d occupancy prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024, pp. 381–397.

[29] J. Zhang, Y. Ding, and Z. Liu, "Occfusion: Depth estimation free multi-sensor fusion for 3d occupancy prediction," in *Proceedings of the Asian Conference on Computer Vision*, 2024, pp. 3587–3604.

[30] Z. Ming, J. S. Berrio, M. Shan, and S. Worrall, "Occfusion: Multi-sensor fusion framework for 3d semantic occupancy prediction," *IEEE Transactions on Intelligent Vehicles*, 2024.

[31] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, "Vision transformer with deformable attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 4784–4793.

[32] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3D," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 194–210.

[33] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall, "FIERY: Future instance prediction in bird's-eye view from surround monocular cameras," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 15 273–15 282.

[34] Y. Li, Z. Ge, G. Yu, J. Yang, Z. Wang, Y. Shi, J. Sun, and Z. Li, "Bevdepth: Acquisition of reliable depth for multi-view 3d object detection," in *Proceedings of the AAAI conference on artificial intelligence*, 2023, pp. 1477–1485.

[35] Y. Li, H. Bao, Z. Ge, J. Yang, J. Sun, and Z. Li, "Bevstereo: Enhancing depth estimation in multi-view 3d object detection with temporal stereo," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 1486–1494.

[36] H. Zhou, Z. Ge, Z. Li, and X. Zhang, "Matrixvt: Efficient multi-camera to bev transformation for 3d perception," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 8548–8557.

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[38] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 252–268.

[39] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint*, vol. arXiv:1706.05587, 2017.

[40] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3075–3084.

[41] B. Graham and L. Van der Maaten, "Submanifold sparse convolutional networks," *arXiv preprint*, vol. arXiv:1706.01307, 2017.

[42] H. Tang, Z. Liu, X. Li, Y. Lin, and S. Han, "Torchsparse: Efficient point cloud inference engine," *Proceedings of Machine Learning and Systems*, vol. 4, pp. 302–315, 2022.

[43] R. Cheng, C. Agia, Y. Ren, X. Li, and B. Liu, "S3CNet: A sparse semantic scene completion network for LIDAR point clouds," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2021, pp. 2148–2161.

[44] S. Zuo, W. Zheng, Y. Huang, J. Zhou, and J. Lu, "Pointocc: Cylindrical tri-perspective view for point-based 3d semantic occupancy prediction," *arXiv preprint*, vol. arXiv:2308.16896, 2023.

[45] J. Wang, W. Yin, X. Long, X. Zhang, Z. Xing, X. Guo, and Q. Zhang, "Occrwkv: Rethinking efficient 3d semantic occupancy prediction with linear complexity," *arXiv preprint*, vol. arXiv:2409.19987, 2024.

[46] C. Zhang, J. Yan, Y. Wei, J. Li, L. Liu, Y. Tang, Y. Duan, and J. Lu, "Occnerf: Advancing 3d occupancy prediction in lidar-free environments," *IEEE Transactions on Image Processing*, 2025.

[47] R. Miao, W. Liu, M. Chen, Z. Gong, W. Xu, C. Hu, and S. Zhou, "Occdepth: A depth-aware method for 3d semantic scene completion," *arXiv preprint*, vol. arXiv:2302.13540, 2023.

[48] J. Hou, X. Li, W. Guan, G. Zhang, D. Feng, Y. Du, X. Xue, and J. Pu, "FastOcc: Accelerating 3D occupancy prediction by fusing the 2D bird's-eye view and perspective view," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 16 425–16 431.

[49] J. Pan, Z. Wang, and L. Wang, "Co-Occ: Coupling explicit feature fusion with volume rendering regularization for multi-modal 3D semantic occupancy prediction," *IEEE Robotics and Automation Letters*, pp. 5687–5694, 2024.

[50] P. Wolters, J. Gilg, T. Teepe, F. Herzog, A. Laouichi, M. Hofmann, and G. Rigoll, "Unleashing hydra: Hybrid fusion, depth consistency and radar for unified 3d perception," *arXiv preprint*, vol. arXiv:2403.07746, 2024.

[51] Y. Wei, L. Zhao, W. Zheng, Z. Zhu, J. Zhou, and J. Lu, "SurroundOcc: Multi-camera 3D occupancy prediction for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 21 729–21 740.

[52] G. Wang, Z. Wang, P. Tang, J. Zheng, X. Ren, B. Feng, and C. Ma, "OccGen: Generative multi-modal 3D occupancy prediction for autonomous driving," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[53] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[54] H. Liu, Y. Chen, H. Wang, Z. Yang, T. Li, J. Zeng, L. Chen, H. Li, and L. Wang, "Fully sparse 3D occupancy prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[55] Y. Li, Z. Yu, C. Choy, C. Xiao, J. M. Alvarez, S. Fidler, C. Feng, and A. 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 (CVPR)*, 2023, pp. 9087–9098.

[56] U. Ruby and V. Yendapalli, "Binary cross entropy with deep learning technique for image classification," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 10, 2020.

[57] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[58] S. Wu, J. Yang, X. Wang, and X. Li, "Iou-balanced loss functions for single-stage object detection," *Pattern Recognition Letters*, vol. 156, pp. 96–103, 2022.

[59] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[60] M. Berman, A. R. Triki, and M. B. Blaschko, "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4413–4421.

[61] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[62] Y. Li, S. Li, X. Liu, M. Gong, K. Li, N. Chen, Z. Wang, Z. Li, T. Jiang, F. Yu, Y. Wang, H. Zhao, Z. Yu, and C. Feng, "Sscbench: A large-scale 3d semantic scene completion benchmark for autonomous driving," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[63] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3292–3310, 2022.

[64] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, S. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2446–2454.

[65] X. Wang, Z. Zhu, W. Xu, Y. Zhang, Y. Wei, X. Chi, Y. Ye, D. Du, J. Lu, and X. Wang, "Openoccupancy: A large-scale benchmark for surrounding semantic occupancy perception," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[66] X. Tian, T. Jiang, L. Yun, Y. Wang, Y. Wang, and H. Zhao, "Occ3D: A large-scale 3D occupancy prediction benchmark for autonomous driving," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023, pp. 64 318–64 330.

[67] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," *arXiv preprint*, vol. arXiv:2301.00493, 2023.

[68] X. Liu, M. Gong, Q. Fang, H. Xie, Y. Li, H. Zhao, and C. Feng, "Lidar-based 4D occupancy completion and forecasting," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 11 102–11 109.

[69] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2021.

[70] J. Ma, X. Chen, J. Huang, J. Xu, Z. Luo, J. Xu, W. Gu, R. Ai, and H. Wang, "Cam4DOcc: Benchmark for camera-only 4D occupancy forecasting in autonomous driving applications," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 21 486–21 495.

[71] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of LiDAR sequences," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9297–9307.

[72] A.-Q. Cao and R. de Charette, "MonoScene: Monocular 3D semantic scene completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 3991–4001.

[73] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. Rus, and S. Han, "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," *arXiv preprint*, vol. arXiv:2205.13542, 2022.

[74] Z. Li, Z. Yu, D. Austin, M. Fang, S. Lan, J. Kautz, and J. M. Alvarez, "Fb-occ: 3d occupancy prediction based on forward-backward view transformation," *arXiv preprint*, vol. arXiv:2307.01492, 2023.

[75] J. Xu, L. Peng, H. Cheng, L. Xia, Q. Zhou, D. Deng, W. Qian, W. Wang, and D. Cai, "Regulating intermediate 3D features for vision-centric autonomous

driving," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[76] M. Tan and Q. V. Le, "EfficientNetV2: Smaller models and faster training," in *Proceedings of International Conference on Machine Learning (ICML)*, 2021, pp. 10 096–10 106.

[77] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon, "Differentiable rendering: A survey," *arXiv preprint*, vol. arXiv:2006.12057, 2020.

[78] C. Lassner and M. Zollhöfer, "Pulsar: Efficient sphere-based neural rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[79] M. Pan, J. Liu, R. Zhang, P. Huang, X. Li, B. Wang, H. Xie, L. Liu, and S. Zhang, "RenderOcc: Vision-centric 3D occupancy prediction with 2D rendering supervision," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[80] Y. Huang, W. Zheng, B. Zhang, J. Zhou, and J. Lu, "SelfOcc: Self-supervised vision-based 3D occupancy prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[81] Q. Sun, C. Shu, S. Zhou, Z. Yu, Y. Chen, D. Yang, and Y. Chun, "GSRender: Deduplicated occupancy prediction via weakly supervised 3D gaussian splatting," 2024.

[82] H. Jiang, L. Liu, T. Cheng, X. Wang, T. Lin, Z. Su, W. Liu, and X. Wang, "GaussTR: Foundation model-aligned gaussian transformer for self-supervised

3D spatial understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[83] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, *et al.*, "Mmdetection: Open mmlab detection toolbox and benchmark," *arXiv preprint*, vol. arXiv:1906.07155, 2019.

[84] H. Cai, C. Gan, and S. Han, "Efficientvit: Enhanced linear attention for high-resolution low-computation visual recognition," *arXiv preprint*, vol. arXiv:2205.14756, 2022.

[85] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[86] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson, "SynSin: End-to-end view synthesis from a single image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[87] Z. Wang, "3D representation methods: A survey," *arXiv preprint*, vol. arXiv:2410.06475, 2024.

[88] K.-A. Aliev, A. Sevastopolsky, M. Kolos, D. Ulyanov, and V. Lempitsky, "Neural point-based graphics," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 696–712.

[89] W. Yifan, F. Serena, S. Wu, C. Öztireli, and O. Sorkine-Hornung, "Differentiable surface splatting for point-based geometry processing," *ACM Transactions on Graphics*, vol. 38, no. 6, 2019.

[90] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[91] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, 2023.

[92] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024.

[93] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[94] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," *arXiv preprint arXiv:2302.12288*, 2023.

[95] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, *et al.*, "Swin transformer v2: Scaling up capacity and resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 12 009–12 019.

[96] C. Sun, J. Choe, C. Loop, W.-C. Ma, and Y.-C. F. Wang, "Sparse voxels rasterization: Real-time high-fidelity radiance field rendering," in *Proceedings*

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2025.

[97] X. Huang, P. Wang, X. Cheng, *et al.*, "The apolloscape open dataset for autonomous driving and its application," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[98] E. Panagiotaki, T. Reinmund, S. Mouton, L. Pitt, A. S. Shanthini, W. Tubby, M. Towlson, S. Sze, B. Liu, C. Prahacs, *et al.*, "Robotcycle: Assessing cycling safety in urban environments," in *IEEE Intelligent Vehicles Symposium (IV)*, 2024, pp. 357–363.