Advancing Occupancy and Flow Estimation for Autonomous Driving: Insights from SparseOcc

Gijs de Jong, Robin Reitsma, Harold Ruiter, Marina Orozco and Danilo Toapanta

University of Amsterdam
Amsterdam Science Park 904

gijs.de.jong@student.uva.nl
robin.reitsma@student.uva.nl
harold.ruiter@student.uva.nl
marina.orozco.gonzalez@student.uva.nl
danilo.toapanta.barahona@student.uva.nl

Abstract

A large amount of research is being done in the field of autonomous driving. Constructing a description of the surrounding environment is hard, due to the complex nature of traffic. Constructing a system that enables a car to understand its environment with the level of perception akin to a human is a big challenge. In this context, the Autonomous Grand Challenge is an opportunity to develop the field and search for its boundaries.

Our model is proposed within this framework, and with it we enhance to complete the current state of the art, SparseOcc. This model has surpassed previous models in terms of both performance and speed, but lacks of one of the main goals of the challenge: a representation of the flow in the scene. With our approach, SparseOcc includes a specific module for flow, and it is our task in this paper to examine it and explore which configurations lead to better results.

1. Introduction

The field of autonomous driving is evolving rapidly, and together with advancements within machine learning there are many new challenges to face within the areas of perception, planning and prediction of autonomous vehicles. This work was developed for the Autonomous Grand Challenge 2024 [8], a competition organized by OpenDriveLab that aims to improve autonomous driving in the form of seven challenge tracks. In this work, we focus on the Occupancy and Flow track, where the goal is to predict the 3D occupancy of the complete scene and the flow of the foreground objects given the input image from six cameras.

Occupancy estimation refers to determining whether a specific volume of space is occupied by objects. In autonomous driving systems, it is important for the vehicle to have an accurate estimation of the size, location and type of the objects surrounding it. While traditional methods usually rely on 3D bounding boxes, a full occupancy estimation provides a more detailed and accurate representation of the environment: bounding boxes often suffer from limitations such as imprecise object localization and inability to represent the true shape and size, especially for protruding or complex objects. In contrast, occupancy estimation allows for a voxel-based representation of the space like the one we can visualize in Figure 1b, capturing more information from the actual volume occupied by objects.

Flow estimation refers to determining the motion of the objects that are being detected as they move through different frames over time. This capability is crucial for dynamic scene understanding, enabling autonomous systems to track moving objects and anticipate their future positions.

While autonomous vehicles often have many different types of sensors in order to get more accurate and robust results, the task in the challenge is limited to a vision-centric approach, relying only on six cameras pointing in different directions as the singular perception system of the vehicle during inference. As cameras are generally lighter and less expensive compared to other sensors like LiDAR and radar, forgoing the use of those other sensors has the large benefit of being able to make more compact and affordable vehicle bodies. However, an approach relying solely on cameras also has limitations: cameras are affected by lighting conditions significantly, and their depth estimation is not as accurate and reliable as that provided by other sensors, so performance might not be as good or robust compared to a multi-sensor approach in all scenarios. Lastly, high quality vision-based algorithms also involve significant computational cost.

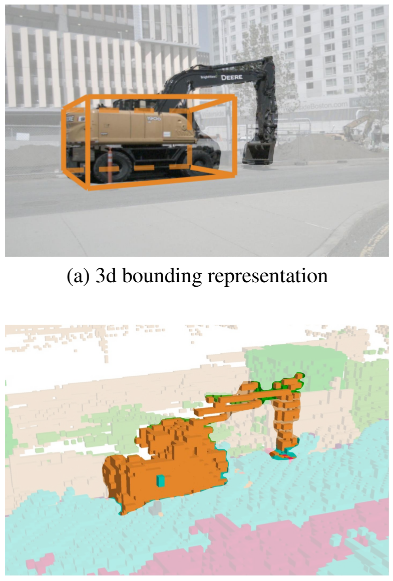
In this work, we combine ideas from OccNet [9], the baseline model for the challenge, and SparseOcc [7], a recent approach that performs occupancy prediction in a fully sparse manner. First, we have reproduced SparseOcc, confirming its reproducibility. Then, inspired by the challenge and its goals, we provided SparseOcc with an additional network that predicts flow vectors. Finally, we have experiment with different parameters in order to find better configurations of this extended SparseOcc.

The code is published at `https://github.com/oxkitsune/occ-flow-2024`.

2. Related Work

In recent years, the advancement of fully autonomous vehicles has driven substantial research in the area of 3D scene reconstruction and object detection. These fields play an important role in enabling automobiles to understand the world around them and make decisions on how to navigate through it.

However, the traditional 3D bounding box around objects often fails to accurately represent outliers and anomalies for specific object types. An example of this can be seen in Figure 1a, where the bounding box inherently misses the arm of the truck. This limitation underscores the need for more refined modeling techniques to accurately depict the complexity of real-world objects.


(a) 3d bounding representation

(b) Voxel representation

Figure 1. A 3d bounding box representation compared to the voxel representation, adapted from [9].

Due to the issues highlighted previously, it is essential to reconstruct the environment with the highest possible precision. A more accurate modeling method is the voxel representation (Figure 1b), where each voxel is defined as either occupied or empty. Occupied voxels can additionally be segmented into specific classes such as a car or pedestrian. The movement of objects can be modelled by assigning each voxel a "flow" vector, describing the current motion of the voxel.

The voxel representation is able to represent possible anomalies with greater detail and model individually moving parts of an object, making it perfect for complex environments such as traffic scenarios. The following section introduces recent developments in vision-centric voxel occupancy and flow prediction techniques, along with relevant background information.

Tong et al. introduced **OccNet**: a multi-view vision-centric pipeline aimed at reconstructing 3D occupancy by employing a cascade and temporal voxel decoder. It uses a backbone model to extract features from the images, these features are then used to create additional features, in a different space. Namely the Bird's Eye View (BEV) space, using the encoder part of the BEVFormer architecture [5]. The BEVFormer creates a BEV representation using multiple visual inputs. Both the image features and BEV features are then used to create a dense voxel representation of the environment, where each voxel is assigned a class and a flow vector. However, computing a dense voxel representation of an environment is computationally expensive, especially considering the temporal and semantic consistency are achieved using an attention mechanism. Temporal and semantic consistency are important to deal with possible occlusions of objects and clear object boundaries.

The **Mask Transformer** [3] was designed to address occlusions and object boundaries in segmentation tasks more efficiently using a mask-based attention mechanism. This mechanism masks certain parts of the input using a binary mask, allowing it filter out irrelevant parts of the input before applying the attention mechanism. The mask operates similarly to the queries, keys and values in the standard attention mechanism [10] and dynamically generates a mask for each input. The mask transformer is able to learn which parts of the input are important, by optimising the mask generation during the training process. While this could be used to improve the efficiency of the attention in the original OccNet, the dense voxel representation contains mostly empty voxels.

To address this, Liu et al. proposed SparseOcc; a novel approach using a fully sparse latent representation. Liu et al. measured the sparsity of geometry in the voxel scene representations and found that scenes mostly contain empty voxels (See Figure 2). SparseOcc builds on **SparseBEV** [6], which uses sparse convolutional techniques to create BEV representations. SparseBEV creates sparse BEV representations by learning the appropriate scale of attention for each query, allowing it to create multi-scale representations without explicitly constructing a dense BEV representation. Additionally it uses adaptive spatio-temporal sampling, using a linear layer to generate a set of sampling offsets adaptively from the query features. This improves the spatial and temporal coherence of the points used for sampling the image features. To decode features from the sparse BEV representation, it dynamically adjusts the weight of a feature based on the queries.

SparseOcc [7] utilizes the SparseBEV features and employs a cascading approach to decode them into multiple levels of voxel features. This hierarchical decoding process enables SparseOcc to effectively capture various levels of spatial details within the voxel grid. To classify the voxels, SparseOcc utilizes a mask transformer.

By integrating the SparseBEV features with the Mask2Former-based mask transformer [3], SparseOcc is able to accurately predict the occupancy of the scene. The sparse BEV features offer an efficient representation of the environment, while the mask transformer enhances this representation by introducing a powerful mechanism for voxel-wise classification. This combination enables SparseOcc to achieve precise and computationally efficient scene understanding, making it well-suited for applications in complex and dynamic environments.

SparseOcc

SparseOcc, proposed by Liu et al., introduces a novel approach to utilizing fully sparse latent representation. Previous methods typically used dense 3D features at some point of their models (as we will discuss in section 2). However, SparseOcc has shown that a sparse approach can be used efficiently when exploiting the sparsity of the scene data, as shown in Figure 2.

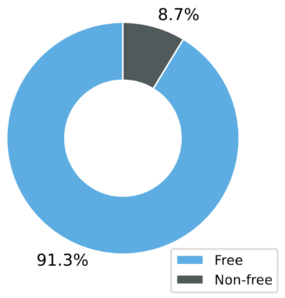


Figure 2. After measuring geometry sparsity, the Liu et al. found that even the scene with the fewest empty voxels still has over 90% empty voxels.

With this approach, SparseOcc becomes the first fully sparse occupancy network and achieves the state of the art in both running time and performance, thanks to the introduction of the sparse latent diffuser, which effectively leverages the sparse nature of the of the scenes.

By incorporating sparse convolution layers and contextual aggregation blocks, the sparse latent diffuser achieves a balance between spatial sparsity and scene completion. Additionally, SparseOcc incorporates a sparse feature pyramid to enable multi-scale representation and employs a transformer head for the final occupancy prediction.
Although SparseOcc primarily utilizes sparse representations to enhance efficiency and minimize computational burden, it is not completely sparse. The model introduces dense components at certain stages to improve performance. More specifically, SparseOcc integrates dense layers in the later stages of its pipeline to refine the features extracted by the sparse convolutions. Furtheremore, the transformer head use for the final processing involves some level of densification to ensure precise 3D scene reconstruction.

3. Methodology

3.1. Model description

As introduced in previous sections, our work uses SparseOcc [7] as a baseline. This model achieved a state of the art performance by exploiting the sparse nature of the scene geometry.

The SparseOcc architecture consists of the following steps, visualized in Figure 3: First, it uses an image backbone to extract 2D features from the multi-view images that are collected in each time step. In our case, the pre-trained weights [1] for the ResNet-50 [4] backbone are taken from a Cascading R-CNN [2]. This backbone is pre-trained on the nuScenes dataset, this makes it easier for the model learn the voxel representation. Next, these features are sent into a Sparse Voxel Decoder and a Mask Transformer.

The **Sparse Voxel Decoder** divides the 3D space in K voxels of equal size, assigns an occupancy score to each one and finally, conducts pruning to remove empty voxel grids. The pruning is done by either eliminating voxels with a score under a certain threshold, or by retaining the top-k voxels with the largest score. This pruning strategy is named dynamic thresholding as scores under certain thresholds are eliminated to on areas more likely to be relevant. As the original authors concluded that the top-k approach was the preferred method on the nuScenes dataset, we leverage it for our experiments as well.

The detailed architecture of the Sparse Voxel Decoder is visualized in Figure 4. It uses a transformer-like architecture with a self-attention mechanism to aggregate local and global features for each query voxel. Each voxel is represented by query, key, and value vectors. The decoder processes voxel representations through multiple layers, with self-attention followed by feed-forward networks,

[1] https://github.com/open-mmlab/mmdetection3d/tree/main/configs/nuimages

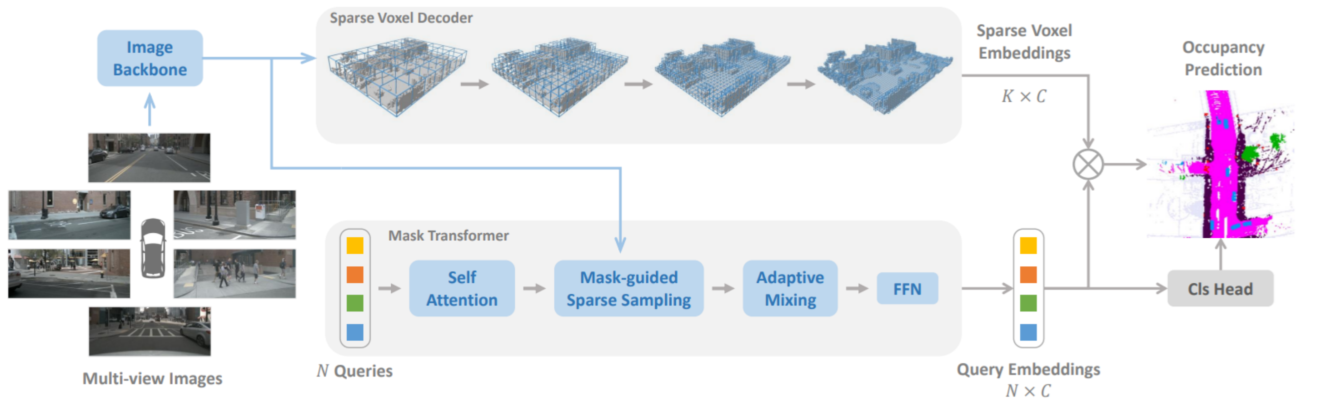


Figure 3. The SparseOcc architecture consists of a Sparse Voxel Decoder and a Mask Transformer. They take the features from the image backbone, and output the predicted occupied voxels with their class label prediction [7].

combining local and global features to enhance spatial detail and overall scene understanding.

At each layer's end, the decoder upsamples each voxel by 2×, subdividing it for higher resolution predictions. It then estimates occupation scores for these smaller voxels. Voxel grids are pruned based on scores, either by eliminating low-score voxels or retaining the top-K highest scores, focusing on the most relevant regions. The occupation scores are supervised using a Binary Cross Entropy (BCE) loss, guiding the model to improve predictions. The remaining voxel tokens after pruning serve as inputs for the next layer, refining voxel representations iteratively and enhancing model accuracy. This architecture effectively transforms 2D image features into high-resolution 3D occupancy predictions.

The **Mask Transformer** used in the model is inspired by Mask2Former [3]. It consists of three main steps: multi-head self-attention (MHSA), mask-guided sparse sampling, and adaptive mixing. The MHSA mechanism allows the model to capture dependencies within the data by attending to different parts of the input simultaneously. After this, a mask-guided sparse sampling focuses the attention mechanism on relevant parts of the input. This effectively reduces computational load and improves on efficiency. Finally, adaptive mixing combines the information from the sampled regions to generate refined mask predictions. The Mask Transformer uses N sparse queries, including a ”no object” category, to predict the mask and label for each segment.

Finally, the outputs of the Sparse Voxel Decoder and Mask Transformer are combined to give an occupancy prediction of the surroundings, with a class assigned to each occupied voxel.

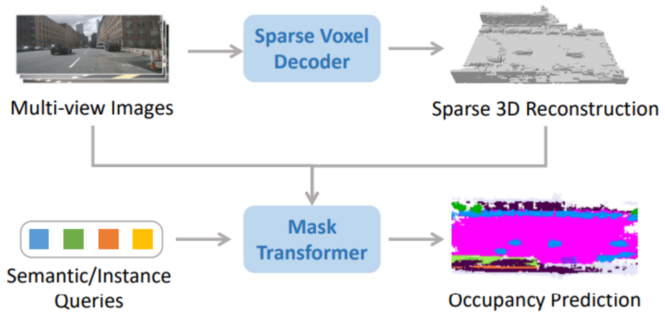


Figure 4. Detailed architecture of the Sparse Voxel Decoder architecture [7]

3.2. Flow Estimation

The original implementation of the SparseOcc network only predicts the occupancy and classes of the voxels, and not their flow over the frames. In the original dense OccNet implementation [9], a single MLP is added to predict the flow vectors for each voxel. However, this approach will not work for the voxel decoder used in SparseOcc. This is because the sparse occupancy is predicted on multiple levels of detail, and the sparse nature of the occupancy predictions makes it impractical to simply run the occupancy through an MLP to predict flow vectors.

Instead, the flow vectors are predicted in meters per second for each level in the voxel decoder and are returned alongside the occupancy predictions. The loss for the flow is calculated using the mean squared error between the predicted flow vectors and the ground truth. This loss is computed individually for each level of the voxel decoder, similar to the occupancy predictions.

The voxel decoder predicts occupancy and flow from coarse to fine, while the ground truth label is only available for the fine occupancy grid. To compute the loss for the coarser levels, the ground truth is down sampled to match the level's resolution. The prediction by the final level of the voxel decoder is more important than the lower levels, as it is used for the mask loss. Similarly we assign a higher weight to the flow prediction at the final level, to avoid the loss of the lower levels causing noisy losses.

4. Experiments

4.1. Dataset

In the original work for SparseOcc, Liu et al. trained the model on Occ3D-nus, an occupancy dataset based on the full nuScenes dataset [1]. Due to training time constraints, we modified the code and annotation files to train on only the nuScenes-mini subset. This subset trains and validates on only 8 and 2 scenes respectively, compared to the 700 and 150 in the full nuScenes dataset. While the code was adapted for the subset, training the model remains fully compatible with the full dataset.

4.2. Occupancy metric

SparseOcc [7] also introduces a novel metric for occupancy scoring: Ray-level mIoU (RayIoU for short). The authors motivate that the classic mIoU approach is insufficient, as it does not consistently penalize behind occluded areas. This often leads to predicting distances to be closer than they really are, as seen in Figure 5.

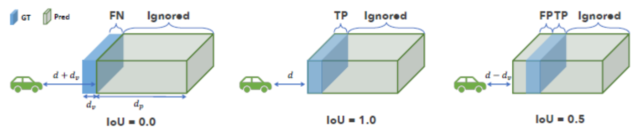


Figure 5. Computed IoU in three different situations: The model penalizes the depth of objects to be inconsistently. In the first case, where the distance is predicted to be d_v further than the ground truth, the IoU is 0.0. In the third case, where the distance is predicted to be d_v closer than the ground truth, the IoU is 0.5. This discrepancy leads to distance predictions being biased to being closer than the ground truth.

RayIoU manages to avoid this issue through:

1. Emulating LiDAR by projecting query rays into both the predicted 3D occupancy volume and to the ground-truth occupancy, computing the distance rays travel before intersecting any surface, and retrieving the corresponding class label.
2. Computing the metric by only evaluating voxels that are present in the ground truth.
3. Considering as True Positive predictions that are in the correct class label and have an L1 error of the predicted depth of objects under a certain threshold, that might be specified through hyperparameters with values as 1m, 2m, 4m, etc.

4.3. Flow metric

The metric we use to measure the flow is the one provided by nuScenes datasets [1], average velocity error (AVE). This metric is computed as the absolute velocity error in m/s.

4.4. Our contribution

In this project our contribution can be summarize in these two goals:

- Reproducing the original SparseOcc from Liu et al. over the nuScenes-mini subset of the dataset, in order to evaluate its **reproducibility**.
- Incorporate a **flow module** to the model, aligning with the goals of the challenge. We will test this new model over different configurations, changing the backbone and batchsize, and comparing the different results we got.

5. Results

In Table 1 we compare the reported RayIoU in the original paper for SparseOcc with the results we got after reproducing the model. The lower value of RayIoU in our model can be due to the fact that we trained over a reduced subset of the dataset: a decrease of the 88% of the dataset has lead to a decrease of the 73% in the RayIoU.

Table 1. Comparison of RayIoU obtained by the original paper of SparseOcc, our reproduction of SparseOcc over the subset of data and our extension of SparseOcc with the flow module.

Method	RayIoU
Original SparseOcc	34.0
Rep. SparseOcc	9.848
SparseOcc + Flow Module	8.725

Table 2. Comparison of training runs with and without the flow module.

Model	LR	RayIoU	RayIoU$_{1m}$	RayIoU$_{2m}$	RayIoU$_{4m}$
R50-SparseOcc (flow)	4e-4	9.6	6.3	9.3	13.1
R50-SparseOcc	4e-4	9.0	5.9	8.8	12.3
R50-SparseOcc (flow)	2e-4	8.7	5.8	8.4	12.0
R50-SparseOcc	2e-4	9.8	6.7	9.6	13.2

In the table above Table 2, we notice something interesting in the Ray-IoU metrics, where the runs trained with a higher learning rate seem to benefit more from the flow module. This is likely caused by the fact that the flow prediction could indirectly lead to a focus on temporal consistency.

Lastly, we ran an experiment with a different image feature backbone, using a ResNet-101 model, with pre-trained weights taken from the R101 version of the Cascading R-CNN [2] used for the main experiment.

Table 3. RayIoU scores on the test set for models using a ResNet101 feature backbone.

Model	LR	RayIoU	RayIoU$_{1m}$	RayIoU$_{2m}$	RayIoU$_{4m}$
R101-SparseOcc (flow)	4e-4	9.9	6.2	9.4	14.2
R101-SparseOcc	4e-4	8.3	5.4	8.2	11.4

From the results in Table 3 it is clear that the larger image backbone results in better scores. However, the larger backbone resulted in an increased training time, which is to be expected.

6. Discussion

6.1. Reproduction and improvement

The work cited by the Occupancy and Flow challenge as baseline proved difficult to reproduce and improve on. It was difficult to reproduce, as some dependencies were not available on our compute cluster. The approach to predicting the voxel flow in the original OccNet, using a single feed-forward layer, could not be adapted to the SparseOcc pipeline directly, due to the differences in the voxel representations. Instead it inspired the technique applied on each level in SparseOcc's the voxel decoder.

6.2. Limitations

Due to the enormous amount of data and limitations with our compute cluster we decided to train our models on a small subset of the nuScenes dataset. This resulted in significantly lower scores on the metrics. We trained a baseline on the same subset of data, to have a point of reference for the results of the other experiments. We were able to improve the scores of the model using our improvements.

Due to time constraints we were unable to verify whether our changes scale, and still lead to improved scores when training on the full dataset. Additionally the weights set for each level of the flow loss should be experimented with, as they were arbitrarily set to $0.5, 1.0, 2.0$ respectively.

We also hypothesize that the base SparseOcc can be improved significantly by improving the pruning of empty voxels. The current top-k approach works well, and is computationally efficient. However, using a more complex approach inspired by the masked attention in Mask2Former or adapting the matrix memory approach introduced by the novel xLSTM architecture could definitely lead to improvements.

7. Conclusion

In this paper, we have successfully implemented a flow module to the current state-of-the-art model, SparseOcc, aligning it with the remaining goal of the challenge that it lacked of. More experiments should be done in the future in order to improve the performance in our model, as due to the limited time and computation resources we had access to, our model will serve as a first baseline over which improve.

In order to make the dataset more manageable, we have created a mini subset of the original nuScenes dataset,reducing it to the 89% of the dataset. This substantial reduction resulted in significantly increased processing speed.

We have also run the original SparseOcc code as an experiment in reproducibility over our reduced dataset, resulting in a smaller performance compared to the original approach. This decrease in the RayIoU finds its explanation in the smaller dataset we have used, which did not allow the model to reach its full potential. Nonetheless, we can conclude that the experiment was successful as we didn't find any problems making their code work.

Finally, another experiment was performed with different configurations of backbones, batches and learning rates in order to find the best configuration of all of them.

8. Work Distribution among members

We divided our efforts among organization, research, coding, and writing. Initially, our tasks were more fixed, but by the end of the project, all of us contributed to different aspects. Here is an overview of each team member's contributions:

- *Gijs de Jong*: For the code: Reproduction, extra experiments, dataset preprocessing, flow module, flow loss, running experiments. Poster preparation. For the report, related work, discussion, methodology, introduction and results.
- *Robin Reitsma*: Report writing, related work studies & summaries, poster work
- *Harold Ruiter*: For the code: Reproduction, extra experiments, dataset preprocessing, flow module. Poster design.
- *Marina Orozco*: Organization and coordination. Theoretical research. Poster design and presentation. For the report: Abstract, Introduction, Model description, Occupancy and Flow metric, Results. Reviewing.
- *Danilo Toapanta*: Review model architectures to improve SparceOcc and expand on related work. Prepare preprocessing on the smaller nuScenes dataset

References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Gi-

ancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 5

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 43(5):1483–1498, 2019. 3, 6

[3] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation, 2022. 2, 3, 4

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 3

[5] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European conference on computer vision*, pages 1–18. Springer, 2022. 2

[6] Haisong Liu, Yao Teng, Tao Lu, Haiguang Wang, and Limin Wang. Sparsebev: High-performance sparse 3d object detection from multi-camera videos, 2023. 2

[7] Haisong Liu, Yang Chen, Haiguang Wang, Zetong Yang, Tianyu Li, Jia Zeng, Li Chen, Hongyang Li, and Limin Wang. Fully sparse 3d occupancy prediction, 2024. 2, 3, 4, 5

[8] OpenDriveLab. Opendrivelab challenge 2024. `https://opendrivelab.com/challenge2024/`, 2024. Accessed: 2024-06-03. 1

[9] Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, et al. Scene as occupancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8406–8415, 2023. 2, 4

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 2