DIO: Decomposable Implicit 4D Occupancy-Flow World Model

Christopher Diehl[1,†,*] Quinlan Sykora[2,3,*]
Ben Agro[3,†] Thomas Gilles[2] Sergio Casas[†] Raquel Urtasun[2,3]
[1]TU Dortmund University [2] Waabi [3] University of Toronto
christopher.diehl@tu-dortmund.de, {qsykora, tgilles, urtasun}@waabi.ai

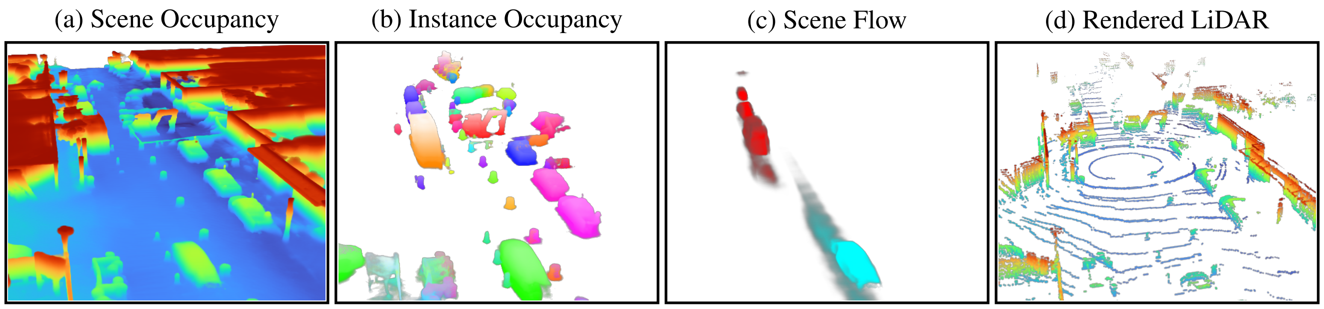

Figure 1. We present DIO, a world model that learns unsupervised 4D occupancy and flow (a, c) and can be decomposed into instance occupancy (b) as well as transfered to downstream tasks like LiDAR point cloud forecasting (d).

Abstract

We present DIO, a flexible world model that can estimate the scene occupancy-flow from a sparse set of LiDAR observations, and decompose it into individual instances. DIO can not only complete instance shapes at the present time, but also forecast their occupancy-flow evolution over a future horizon. Thanks to its flexible prompt representation, DIO can take instance prompts from off-the-shelf models like 3D detectors, achieving state-of-the-art performance in the task of 4D semantic occupancy completion and forecasting on the Argoverse 2 dataset. Moreover, our world model can easily and effectively be transferred to downstream tasks like LiDAR point cloud forecasting, ranking first compared to all baselines in the Argoverse 4D occupancy forecasting challenge.

1 Introduction

Planning a safe motion requires robots, such as self-driving vehicles (SDVs), to have an accurate understanding of the world. An important challenge of perception systems is their ability to understand individual objects and their shape when dealing with sparse and noisy observations. Moreover, scene completion at *the current time* is not sufficient, as a robot also needs a precise understanding of the evolution of the 4D scene *into the future* through a world model to plan a safe trajectory. These systems can benefit from reasoning about instances to get a complete understanding of the scene, allowing a more fine-grained understanding of the interactions between actors. For instance, if a planned trajectory overlaps with forecasted future occupancy, it is important to understand where that occupancy came from to reason about who has the right-of-way.

Two dominant groups of methods have emerged as interpretable world models for autonomy: *Instance-based methods* [1–10] first detect a discrete set of objects in the scene, followed by forecasts of possible future trajectories. While instance information can benefit motion planning, this approach leads to information bottlenecks (e.g., a finite set of predicted trajectories). *Bird's-eye-view (BEV) semantic occupancy fields* [11–17] are more suitable to represent future motion uncertainties [12, 18] but lack instance information that reduces expressivity and intepretability. Moreover, both these approaches are often supervised exclusively with bounding box labels, which leads to a poor approximation of the true object geometry, and the BEV assumption may not hold. This highlights the need for a 4D (x, y, z, t) understanding of the scene.

Recent works [18, 19] propose unsupervised 4D occupancy world models, allowing training on a vast amount of unlabeled LiDAR data, better capturing the true geometry, and enabling fine-tuning on specific tasks such as LiDAR point cloud forecasting and semantic occupancy perception and prediction. However, these models still face challenges with (1) decomposing dynamic scenes, (2) predicting occupancy of fine-grained structures, and (3) accurately fore-

[*] Denotes equal contribution. [†] Work done while at Waabi.

casting, all of which are relevant for safe motion planning.

To address these limitations, we propose DIO, a *4D occupancy-flow world model* that exploits unsupervised LiDAR data together with object bounding box labels as supervision. DIO is able to decompose the scene into instances, while improving on both the fine-grained details and the forecasting abilities of previous models. As shown in Fig. 1, this model can predict 4D scene occupancy and flow (3D + time), decompose it into individual objects, and can also be transferred to the task of LiDAR point cloud forecasting. In contrast to many scene completion approaches [20, 21], this method requires only 3D bounding boxes labels to predict instances that approximate the true geometry. We further propose a novel method allowing for the decomposition of 4D scene occupancy and flow into specific instances while maintaining the ability to predict scene occupancy. DIO accomplishes this by accepting source points $\mathbf{s} = (x_s, y_s, z_s)$ that allow the user to indicate the object of interest in a flexible way. In summary, our primary contributions are as follows:

Contributions: **1)** DIO is the first method to be able to decompose scene occupancy using source point prompting into individual instances, enabling scene occupancy, instance occupancy, and unsupervised flow prediction. **2)** It leverages both LiDAR and bounding box labels for supervision to predict instance-level occupancy and **3)** our new sparse architecture achieves state-of-the-art results in different tasks (LiDAR point cloud and 4D geometric occupancy completion and forecasting) on the Argoverse2 dataset.

2 Related Work

Spatial-Temporal Occupancy Prediction: This task involves predicting the probability of a specific area in space being occupied using sensory data such as LiDAR [12, 13, 16], RADAR [22, 23], and camera inputs [14, 15, 24, 25]. To tackle the downsides of bounding box and BEV assumptions, various works [26–29] focus on predicting the 4D semantic occupancy. However, many of these approaches rely on point cloud semantic segmentation labels [30], which are expensive, and only consider a predefined set of classes, which can limit an autonomous system's robustness to new or unusual objects. To tackle these limitations, recent works [18, 19] have instead proposed predicting 4D geometric occupancy, relying solely on LiDAR self-supervision.

Most works [19, 26–29] predict occupancy in space and future time as a 4D voxel grid, which is inefficient in terms of memory and runtime, leading to low resolution outputs with quantization errors. In contrast, UnO [18] proposes an implicit occupancy model, allowing it to efficiently query occupancy at continuous (x, y, z, t) points of interest. However, the predictions of this work still struggle to capture fine-grained structures as it encodes the scene into a single resolution BEV feature map.

LiDAR Point Cloud Forecasting: LiDAR point cloud forecasting emerged as a way to measure the capabilities of world models trained on large unlabeled datasets. While some methods [31–34] directly predict the point clouds, others first predicts a 4D occupancy representation [18, 19] followed by the LiDAR ray depth given the sensor intrinsics and future extrinsics. Our work builds on the latter and due to the decomposed occupancy output also allows us to complete and forecast LiDAR point clouds for specific objects of interest in higher detail.

Scene Completion: *Scene completion* [35–38] refers to the task of completing the geometry of a scene given noisy and sparse sensor measurements. *Semantic scene completion* methods [20, 39–41] additionally infer the semantic classes of each occupied area, and *panoptic scene completion* [20, 21, 42] extends this task to also understand instances. To obtain a fine-grained understanding of instances, works in this family often rely on labeled data such as CAD models [43, 44], meshes [45, 46], or semantic point cloud labels [21, 42], which are costly to obtain. While panoptic scene completion has similarities to our work, DIO is not only able to complete the scene and instances occupancy at the *present time*, but also forecasts its future evolution. Moreover, our method only requires unlabeled point clouds and bounding box labels which are cheaper to obtain than the above-mentioned labels.

3 Decomposable 4D Occupancy-Flow

We present DIO, a model that estimates the precise full 3D shape of objects and forecasts their motion from sparse and noisy LiDAR observations. We develop a model that can be prompted for occupancy and flow of a particular instance at a particular spatial-temporal 4D point. This enables us to estimate occupancy in an implicit manner, with the advantages of infinite continuous resolution, efficient querying constrained to regions of interests, and most importantly, leveraging large amounts of point cloud data for training. Aside from point cloud sensor inputs and localized ego poses –which are both the norm in modern self-driving platforms [34, 47]– the model requires an indicator, or "prompt", to identify which instance it should predict in the 4D space, as inspired by Segment Anything [48]. In a real application, these instance prompts can be obtained through many existing methods that estimate the centroid locations of objects in the scene (e.g., 3D object detectors [49–53]), and can be combined with methods highlighting the points of interest in a scene like a planner [54]. More details on the theoretical efficiency of these combined methods are in the supplementary.

3.1. Decomposable Occupancy-Flow Task

To decompose the scene, DIO requires a prompt made of a source point $\mathbf{s}$ and a query point $\mathbf{q}$. The source point rep-

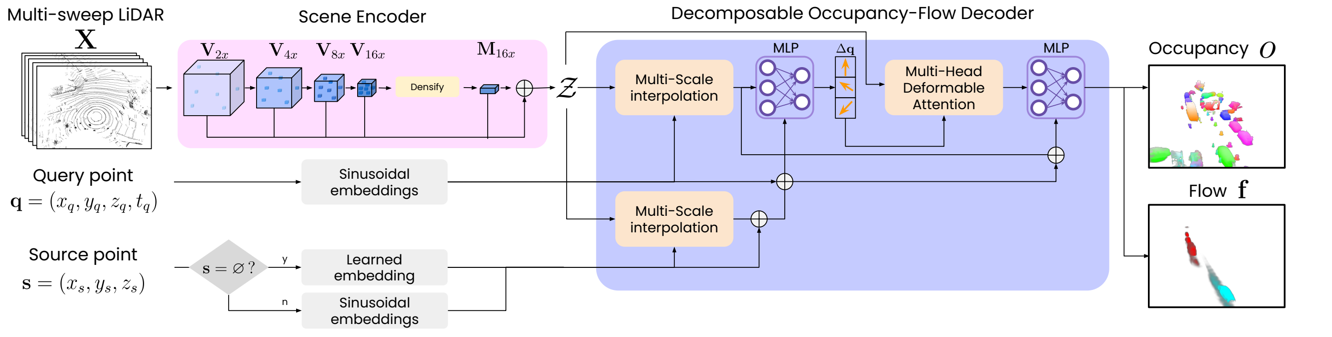

Figure 2. DIO predicts the probability o that the object currently occupying $\mathbf{s}$ will occupy (x_q, y_q, z_q) at time t_q, and in that event, the most likely flow vector $\mathbf{f}$. The occupancy and flow are visualized by overlaying a grid of query points for object centroid (as source point).

resents the assumed 3D location in the current frame of the instance we want to predict. The query point indicates the 4D point in space and time where we want to know if it will be occupied by the instance or not. To be more precise about the task, it is worth distinguishing different cases. If the source point lies in a foreground instance (e.g., a vehicle) we want to predict the 4D occupancy-flow for that specific instance (i.e., 1 for query points inside the instance and 0 otherwise). If the source point lies in free-space or background, we would like to predict zero occupancy for any query point. For instance, if a detector is used to propose source points, it might have false positive detections, and in that case we would like to predict no occupancy.

More formally, given a tuple $(\mathbf{X}, \mathbf{p})$ composed of past sensor data $\mathbf{X}$ including the most recent observation at time $t = t_0$, a prompt $\mathbf{p} = (\mathbf{q}, \mathbf{s})$ composed of a 4D *query point* $\mathbf{q} = (q_x, q_y, q_z, q_t)$, and a 3D *source point* $\mathbf{s} = (s_x, s_y, s_z)$, DIO predicts the probability o that the instance occupying $\mathbf{s}$ at $t = t_0$ will occupy (q_x, q_y, q_z) at q_t (i.e., $q_t - t_0$ seconds into the future), as well its 3D flow $\mathbf{f} = (f_x, f_y, f_z)$. In other words, the goal is to learn a model f_θ parametrized by θ such that

$$o, \mathbf{f} = f_\theta(\mathbf{X}, \mathbf{p}). \tag{1}$$

Finally, we would like the ability to also predict the overall scene occupancy similar to previous works [18, 19], including background and all foreground instances. We achieve this by prompting with an empty source point $\mathbf{s} = \varnothing$. In this case, we expect occupancy for any query point that belongs to either an instance (e.g., vehicle, person, etc.) or to the background (e.g., buildings, trees, ground, etc.). We define the results of prompting with a non-empty 3D source point as *instance occupancy*, and *scene occupancy* when prompting with an empty source point.

3.2. Decomposable Occupancy Model

We design a scene encoder that is composed of a sparse 3D backbone and a dense BEV neck to extract multi-resolution 3D sparse feature volumes as well as a dense BEV feature map. Given a prompt, the decoder predicts the occupancy and flow for each query point corresponding to the instance —or whole scene— described by the source point. We refer the reader to Figure 2 for an overview.

Scene Encoder: We leverage the 3D sparse backbone of SECOND [55] to encode multiple sweeps (360° scans) of past and current LiDAR data. Through a series of sparsity-preserving sub-manifold 3D convolutions [56, 57], this backbone gradually downsamples the sparse feature volume. Throughout this downsampling process, we obtain feature volumes with resolution downsampled by 2x, 4x, 8x, and 16x with respect to the input voxels, which we refer to as $\mathbf{V}_{2x}$, $\mathbf{V}_{4x}$, $\mathbf{V}_{8x}$, and $\mathbf{V}_{16x}$ respectively. These sparse feature volumes capture 3D details of the scene precisely, but they have a limited receptive field as features do not propagate into empty voxels, and thus are not well suited for capturing the full scene context. To tackle this, we propose using a *deformable spatial attention neck* to extract a BEV dense feature map with a large receptive field from $\mathbf{V}_{16x}$. This module first densifies the sparse features by padding any voxel without sparse features with zeros, concatenates the features along the height dimension, and runs a multihead spatial attention module [58] over the BEV feature map. Finally, we pass it through a series of BEV convolution blocks to obtain a dense BEV feature map $\mathbf{M}_{16x}$, and denote the transformation from $\mathbf{V}_{16x}$ to $\mathbf{M}_{16x}$ as *Densify* in Fig. 2. This approach exploits the sparsity inherent in the point cloud data for efficiency at high resolutions while extracting high-receptive field features at a low resolution where dense convolutions are affordable. The output are the concatenated scene features $\mathcal{Z} = [\mathbf{V}_{2x}, \mathbf{V}_{4x}, \mathbf{V}_{8x}, \mathbf{M}_{16x}]$.

Decomposable Occupancy-Flow Decoder: The decoder predicts occupancy and flow by attending to the scene features $\mathcal{Z}$ at multiple locations: the query point $\mathbf{q}$, the source point $\mathbf{s}$ as well as at multiple learned offsets from the query point $\mathbf{q} + \Delta\mathbf{q}$. Together with embeddings of $\mathbf{q}$ and $\mathbf{s}$, this is sufficient context for the decoder to perform this task. In more detail, we leverage a *Multi-Scale Interpolation* (MSI) module to perform tri-linear interpolations at (x, y, z) for $\mathbf{V}_{2x}$, $\mathbf{V}_{4x}$, $\mathbf{V}_{8x}$ and a bi-linear interpolation at (x, y) for

$\mathbf{M}_{16x}$, and concatenate all interpolated feature vectors. We use this MSI to interpolate the multi-resolution scene features at $\mathbf{q}$ and $\mathbf{s}$ first. Then, with those features concatenated with the positional embeddings, we predict $4H$ 3D offsets $\Delta\mathbf{q}$ through an MLP to be used in *Multi-Head Deformable Attention* [58]. This module interpolates features at the 4 resolutions with MSI, and then performs cross-attention over the H heads at each resolution. Finally, the resulting features are passed through another MLP, which outputs the occupancy probability and flow vector. Additional details can be found in the supplementary.

3.3. Training

Our goal is to learn fine-grained shape of objects and to be able to decompose them into separate instances. Scene-wise LiDAR unsupervised training [18] provides detailed occupancy labels but does not include any instance information. On the other hand, labelled boxes identify separate objects, but their shapes are very coarse since constrained to 3D boxes. We aim to combine the advantages of both by prompting DIO to reconstruct occupancy inside specific instance boxes. Furthermore, we showcase the ability of this method to generalize to new instances by always training only on a subset of the labelled classes, and evaluating the performance on the held-out remaining labels.

The choice of training prompts —i.e., the combination of source points and query points— is crucial to the performance of our model. It is important to find a good balance between positive and negative queries (i.e., occupied vs. unoccupied). To achieve decomposable occupancy, we use a diverse set of source points (inside instances, outside but close to instances, empty). Figure 3 illustrates the main types of prompts we utilize, and the associated positive vs. negative labels. The notation for prompts is as follows: A set of prompts $\mathcal{P}$ is the combination of two ordered sets of equal length: query points $\mathcal{Q}$ and source points $\mathcal{S}$. If the set of prompts is to be supervised as positive (occupied) we will denote it as $\mathcal{P}^+$, and if it will be supervised as negative (unoccupied) with $\mathcal{P}^-$.

Scene Occupancy Prompts: We take inspiration from 4D-Occ [19] and UnO [18] to leverage LiDAR rays, and the implicit information they contain about which regions are empty or occupied as our self-supervision. Since the SDV is moving, the position of the LiDAR sensor $\mathbf{l}(t) = (l_x(t), l_y(t), l_z(t))$ is a function of time t. We denote a LiDAR return (a.k.a. LiDAR point) with emission time t_r as $\mathbf{r} = (r_x, r_y, r_z)$. Following UnO [18], we assume that the ray segment $\mathcal{R}^-$ connecting $\mathbf{l}(t_r)$ to $\mathbf{r}$ is not occupied, and that the small segment $\mathcal{R}^+$ of length δ immediately past $\mathbf{r}$ and in the same direction is occupied. We then uniformly sample the same number points along each ray from the positive $\mathcal{R}^+$ and negative $\mathcal{R}^-$ portions of the ray, obtaining two sets of query points $\mathcal{Q}^-_{\text{scene}}$ and $\mathcal{Q}^+_{\text{scene}}$ with $|\mathcal{Q}^-_{\text{scene}}| = |\mathcal{Q}^+_{\text{scene}}|$. For all these query points, DIO is prompted with an empty source point ($s = \varnothing$), yielding $\mathcal{P}^-_{\text{scene}}$ and $\mathcal{P}^+_{\text{scene}}$ These prompts are illustrated in Fig. 3(a).

Instance Occupancy Prompts: To train the decomposable occupancy, we assume the existence of I instance labels, each defined by a bounding box $\mathbf{b}_i$ with 3D centroid, length, width, height, and heading information. For each instance $i \in 1, \dots, I$, we generate a set of positive instance prompts $\mathcal{P}^+_{\mathbf{b}_i} = \{(\mathbf{q}, \hat{\mathbf{s}}_i) \mid \mathbf{q} \in \mathcal{Q}^+_{\text{scene}} \wedge \mathbf{q} \subset \mathbf{b}_i\}$ from positive scene prompts that lie inside ($\subset$) an instance bounding box. Here, $\hat{\mathbf{s}}_i$ is a noisy source point sampled uniformly to be within $\mathbf{b}_i$. This noise makes the method robust against imperfect source points during evaluation (e.g., those coming from a 3D instance detector). Doing this for every instance, we obtain positive prompts $\mathcal{P}^+_{\text{obj}} = \bigcup_i \mathcal{P}^+_{\mathbf{b}_i}$. These positive prompts are illustrated in Fig. 3(b) with green circles. For negatives, we want to associate different source points depending if the query point lies within an instance bounding box or not. More precisely,

$$\mathcal{P}^-_{\text{obj}} = \left\{ \begin{array}{ll} (\mathbf{q}, \hat{\mathbf{s}}_i) & \text{if } \mathbf{q} \subset \mathbf{b}_{i \in 1 \dots I} \\ (\mathbf{q}, \hat{\mathbf{s}}_{j \sim 1 \dots I}) & \text{otherwise} \end{array} \;\middle|\; \mathbf{q} \in \mathcal{Q}^-_{\text{scene}} \right\}. \tag{2}$$

By assigning a noisy source point from a random instance j to negative query points that lie outside bounding boxes, we prevent the model from learning a bias that when a source point is provided the query point is likely occupied. There are also positive scene prompts that hit the background (see green points in tree in Fig. 3). However, for training the instance occupancy, we would like to supervise those as negatives. This gives us one more set of prompts $\mathcal{P}^-_{\text{bg}} = \{(\mathbf{q}, \hat{\mathbf{s}}_{j \sim 1 \dots I}) \mid \mathbf{q} \in \mathcal{Q}^+_{\text{scene}} \wedge \mathbf{q} \not\subset \mathbf{b}_i \forall i\}$. All these negative prompts are illustrated in Fig. 3(b) with red circles.

For each instance, we also add hard negative prompts that lie between the original box $\mathbf{b}_i$ and a scaled-up box $\mathbf{b}'_i$. In other words, $\mathcal{P}^-_{\text{box}} = \bigcup_i \{(\mathbf{q}, \hat{\mathbf{s}}_i) \mid \mathbf{q} \sim \mathcal{U}(\mathbf{b}'_i) \wedge \mathbf{q} \subset \mathbf{b}'_i \backslash \mathbf{b}_i\}$. The rationale behind this is the generation of negative supervision for query points for one instance (e.g., $i = 1$) inside the true shape of another nearby instance (e.g., $i = 2$) as the scaled-up box of the former might intersect with the latter, encouraging proper instance separation. These points are illustrated in Fig. 3(b) with orange diamonds.

Finally, to ensure the model predicts no occupancy when queried with a source point in free-space we add another set of negative query points during training. Using the LiDAR rays, we randomly sample K source points in the free-space of the current time step $\mathcal{R}^-_{t=t_0}$ and sample M associated query points to each source point with a gaussian ball.

$$\mathcal{P}^-_{\text{rand}} = \left\{ \left(\mathbf{q}_{k,m} \sim \mathcal{N}(\mathbf{s}_k, \sigma^2 I), s_k \right) \mid \mathbf{s}_k \sim \mathcal{R}^-_{t=t_0} \right\}_{k=1, m=1}^{K,M} \tag{3}$$

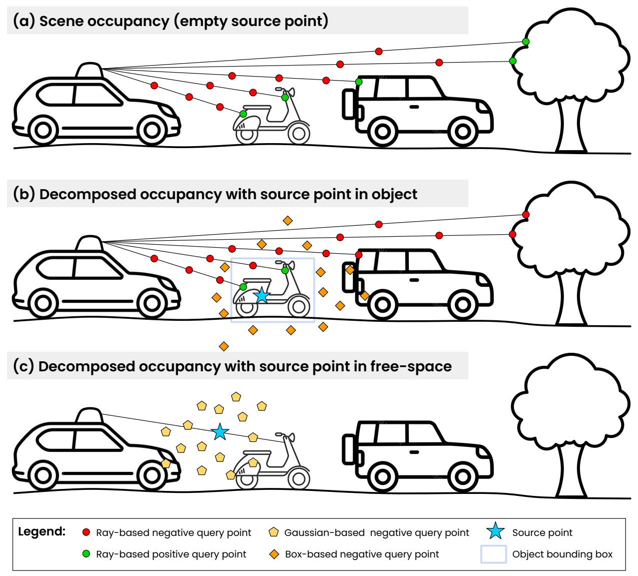
(a) Scene occupancy (empty source point)
(b) Decomposed occupancy with source point in object
(c) Decomposed occupancy with source point in free-space
Legend:
Ray-based negative query point
Ray-based positive query point
Gaussian-based negative query point
Box-based negative query point
Source point
Object bounding box

Figure 3. Prompt generation during training.

Loss Function: We train DIO's parameters θ using a linear combination of occupancy and flow consistency losses:

$$\mathcal{L} = \mathcal{L}_{\text{occ}} + \lambda \mathcal{L}_{\text{flow}}. \tag{4}$$

Recall that $o, \mathbf{f} = f_\theta(\mathbf{X}, \mathbf{q}, \mathbf{s})$ denotes the occupancy probability and flow outputs at a query point $\mathbf{q}$ given source point $\mathbf{s}$. In the following, we overload the outputs by writing them as a function $o(\mathbf{q}, \mathbf{s})$ and $\mathbf{f}_\theta(\mathbf{q}, \mathbf{s})$, where we also omit the input $\mathbf{X}$ for brevity.

Our *occupancy loss* $\mathcal{L}_{\text{occ}}$ is a simple binary cross-entropy (BCE) loss summed across positive prompts $\mathcal{P}^+_{\text{all}} = \mathcal{P}^+_{\text{scene}} \cup \mathcal{P}^+_{\text{obj}}$ and negative prompts $\mathcal{P}^-_{\text{all}} = \mathcal{P}^-_{\text{scene}} \cup \mathcal{P}^-_{\text{obj}} \cup \mathcal{P}^-_{\text{bg}} \cup \mathcal{P}^-_{\text{rand}}$ [16]. We also introduce a *flow consistency loss* to ensure the consistency of the occupancy predictions across timesteps. Given a query point $\mathbf{q} = (x, y, z, t)$ we get the occupancy and flow predictions $o, \mathbf{f} = f_\theta(\mathbf{q}, ...)$ so that we can calculate a new query point $\mathbf{q}' = (x + f_x\Delta t, y + f_y\Delta t, z + f_z\Delta t, t + \Delta t)$ where the occupancy will be after a certain period of time Δt. Assuming the flow is accurate, the occupancy at the new query point should be the same as the original occupancy due to instance permanence. Thus, we can enforce the consistency with a BCE loss:

$$\mathcal{L}_{\text{flow}} = -\sum_{(\mathbf{q},\mathbf{s})\in\mathcal{P}_{\text{all}}} \Big(o(\mathbf{q}, \mathbf{s}) \log(o(\mathbf{q}', \mathbf{s})) + (1 - o(\mathbf{q}, \mathbf{s})) \log(1 - o(\mathbf{q}', \mathbf{s})) \Big). \tag{5}$$

3.4. Tackling Downstream Tasks with DIO

Semantic Occupancy Prediction: With our model combined with a multi-class 3D detector we can recover semantic occupancy (current and future). To do this, we can use the source points corresponding to detection centroids of a certain class (or set of classes) $\mathcal{S}_{\text{cat}}$, and prompt the model at overlapping query points in a grid $\mathcal{Q}_{\text{grid}}$. In other words, use the prompts corresponding to the cross-product $\mathcal{P}_{\text{cat}} = \mathcal{S}_{\text{cat}} \times \mathcal{Q}_{\text{grid}}$. To get the semantic occupancy prediction at a particular query point $\mathbf{q}$, we can simply take the maximum across source points $o_\mathbf{q} = \max_i f_\theta(\mathbf{X}, (\mathbf{q}, \mathbf{s}_i))$. It should be noted that, due to the implicit nature of our decoder, we can also evaluate at different query points that are not restricted to a standard grid (e.g., by identifying regions for each instance based on kinematic constraints or additional map information), which would reduce computation. An advantage of this is the possibility to increase the sampling query resolution under compute constraints. We investigate this more in the supplementary.

Lidar Point Cloud Forecasting: Point cloud forecasting emerged as a task to evaluate 4D understanding in self-driving models by predicting future LiDAR point clouds based on past and current point clouds.[18, 19, 31–33]. This task involves estimating the depth that query rays from future LiDAR returns will travel before hitting a surface. We follow [18] and use a lightweight neural network, g_γ, trained to predict these depths by processing occupancy predictions from our model along each ray. During training, only the renderer parameters γ are being updated to minimize an ℓ_1 loss against ground truth depth, with the world model's parameters θ kept fixed. For LiDAR rendering of instance occupancy, the renderer also outputs a probability if the ray hits an object and supervises it with a cross-entropy loss. For more details, refer to the supplementary.

4 Experiments

This section investigates the following research questions while comparing against the state-of-the-art: **Q1**: Does DIO capture better instance shapes (i.e., completion)? **Q2**: Does DIO improve the occupancy-flow forecasts? **Q3**: Is DIO capable to do open-set predictions when prompted with source points belonging to instances of categories unseen during training? **Q4**: Is DIO robust to being prompted with different types of source points unseen during training, such as object centroids from off-the-shelf detector?

4.1. Experimental Setup

Implementation Details: Following prior work [18, 19], DIO and the baselines receive 3.0 s of past frames. Due to our more efficient sparse backbone, DIO is able to consume 16 past frames at an interval of 0.2 s. We set the learning rate to 1.0×10^{-4}, starting with a 1,000-iteration warmup from an initial rate of 1.0×10^{-5}, followed by a cosine decay learning rate schedule. Training is performed over 50,000 iterations using a batch size of 16 with the AdamW optimizer. On each batch sample, we train DIO on $|\mathcal{Q}^+_{\text{scene}}| = |\mathcal{Q}^-_{\text{scene}}| = 19,000$ used each for the scene prompts and the instance occupancy prompts. Hence,

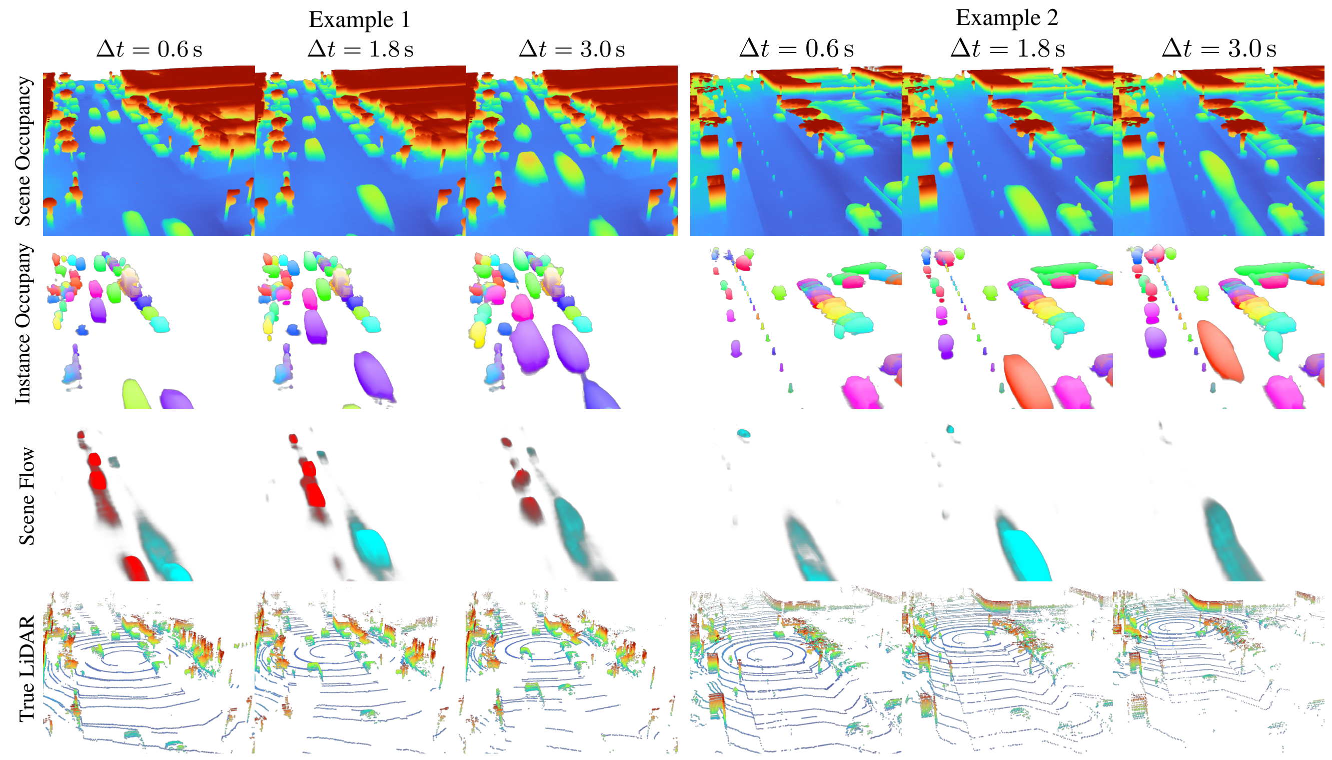
Example 1
$\Delta t = 0.6\,\mathrm{s}$
$\Delta t = 1.8\,\mathrm{s}$
$\Delta t = 3.0\,\mathrm{s}$
Example 2
$\Delta t = 0.6\,\mathrm{s}$
$\Delta t = 1.8\,\mathrm{s}$
$\Delta t = 3.0\,\mathrm{s}$
Scene Occupancy
Instance Occupancy
Scene Flow
True LiDAR

Figure 4. DIO forecasts visualized on two different examples. In example 1, we see fast-moving traffic in both directions and a wide variety of differently sized instances that DIO can decompose and predict. In example 2, we observe various highly detailed structures in the scene occupancy and a large number of individual objects that are precisely isolated.

$|\mathcal{P}^{+}_{\text{scene}}| + |\mathcal{P}^{-}_{\text{scene}}| = |\mathcal{P}^{+}_{\text{obj}}| + |\mathcal{P}^{-}_{\text{obj}}| = 38,000$. Additionaly, we sample each $6,000$ additional object-based $\mathcal{P}^{-}_{\text{box}}$ and free-space-based negative promts $\mathcal{P}^{-}_{\text{rand}}$. Further details are given in the supplementary.

Datasets: For our experiments, we use the popular autonomous driving Argoverse 2 (AV2) *Sensor* dataset. AV2 Sensor comprises 850 sequences collected across Austin, Detroit, Miami, Palo Alto, Pittsburgh, and Washington, D.C. Each sequence spans 15 seconds and includes 150 frames of LiDAR data with bounding box object annotations. We use 700 sequences of the official split for training and 150 sequences for validation.

Baselines: We compare with a variety of supervised and unsupervised world models. ImplicitO-4D [16] represents the state-of-the-art of world models supervised with 3D bounding box labels, and it also features an implicit occupancy architecture. Besides the supervision, an important difference with DIO is that they use dense convolutions instead of sparse. For unsupervised world models, trained with LiDAR-based pseudo-labels, we include 4D-Occ [19] and UnO [18]. 4D-Occ [19] is a representative method for explicit occupancy prediction (i.e., voxel grids over time). Note that we only compare against 4D-Occ [19] in forecasting metrics and not in completion as this baseline does not have occupancy or point cloud predictions for $t = t_0$ — it starts at $t = t_0 + 0.6$s. UnO [18] is the state-of-the-art unsupervised occupancy prediction model, and it uses an implicit architecture similar to ImplicitO-4D [16]. Lastly, we compare against a joint detection and trajectory forecasting oracle (Box Oracle) representing the group of instance-based methods [2, 25, 59, 60]. This oracle simulates a perfect prediction of 3D instance boxes over time.

Completion vs. Forecasting: We evaluate the ability to complete instances and whole scenes ($t = t_0$), as well as forecast them ($t > t_0$). To assess completion while preventing models from exploiting memorized input data, we only feed the models with 60% of the present frame LiDAR points, withholding the other 40% points to be exclusively used during evaluation. For future time forecasting we pass the model all available past and current LiDAR points and models are evaluated at $t = t_0 + \Delta t$, where $\Delta t \in \{0.6\,\mathrm{s}, 1.2\,\mathrm{s}, \ldots, 3.0\,\mathrm{s}\}$.

4.2. Occupancy Prediction

Semantic Occupancy Labels: Evaluation based solely on bounding box labels fails to capture the actual shape of abnormal instances, such as a person with their arms extended in different directions, an excavator, etc. To evaluate the actual shapes, we obtain occupancy pseudo-labels through a combination of ray-tracing of the LiDAR points and bounding boxes. We consider as negative (unoccupied)

	Completion ($t = t_0$)		Forecasting ($t > t_0$)			
	mAP$_I$ ↑	S-IoU$_I$ ↑	mAP$_S$ ↑	S-IoU$_S$ ↑	mAP$_I$ ↑	S-IoU$_I$ ↑
Box Oracle	50.49	51.33	99.40	99.36	47.71	51.07
4D-Occ [19]	-	-	70.48	30.90	56.63	29.48
ImplicitO-4D [16]	65.04	52.96	97.31	69.83	61.51	45.51
UNO [18]	89.43	70.00	97.04	77.86	82.99	60.74
DIO-D	**94.36**	**75.97**	**98.47**	**83.78**	**86.01**	**62.72**

Table 1. 4D semantic occupancy on the `vehicle` class.

	Completion ($t = t_0$)		Forecasting ($t > t_0$)			
	mAP$_I$ ↑	S-IoU$_I$ ↑	mAP$_S$ ↑	S-IoU$_S$ ↑	mAP$_I$ ↑	S-IoU$_I$ ↑
DIO-∅	**95.53**	**79.63**	97.16	80.14	86.14	63.19
DIO-L	95.23	77.65	**98.78**	**87.48**	**87.10**	**64.43**
DIO-NL	95.19	77.65	98.77	87.46	87.04	64.37
DIO-D	94.36	75.97	98.47	83.78	86.01	62.72

Table 2. Ablating the source point type for the `vehicle` class.

	Completion ($t = t_0$)		Forecasting ($t > t_0$)			
	mAP$_I$ ↑	S-IoU$_I$ ↑	mAP$_S$ ↑	S-IoU$_S$ ↑	mAP$_I$ ↑	S-IoU$_I$ ↑
4D-Occ [19]	-	-	85.74	36.95	70.42	43.67
ImplicitO-4D [16]	87.16	24.09	82.49	7.47	62.42	8.01
UNO [18]	85.85	85.84	96.37	70.48	80.10	65.58
DIO-∅	**98.43**	**88.15**	97.49	76.71	**88.45**	**71.42**
DIO-NL	97.61	82.40	**99.05**	**80.54**	82.95	68.77

Table 3. Inspecting the open-set generalization of occupancy reconstruction and forecasting by evaluating on the `holdout` class.

	L1 (m) ↓	AbsRel (%) ↓	NFCD (m^2) ↓	CD (m^2) ↓
RayTracing [19]	2.50	35.00	3.62	17.03
4D-Occ [19]	3.22	19.00	2.45	72.74
UNO [18]	2.24	12.00	0.86	**8.10**
DIO-∅	**2.11**	**11.00**	**0.85**	13.96

Table 4. Results on the Argoverse 2 leaderboard.

all the space traversed by all LiDAR rays up to their return point, and positive (occupied) 10 cm after the LiDAR return point in the direction along the ray for points that fall inside an instance bounding box that belongs to the set of desired semantic categories. The occupancy of points that are not traversed by any LiDAR ray (including the line segment after the positive segment described above) is considered unknown and therefore ignored during evaluation. To get a more fine-grained understanding of how well instance shapes vs. free-space are captured, we consider two versions of the semantic occupancy pseudo-labels: scene-level where we use all rays available (denoted in the metrics with subscript S) and instance-level where we only use ray segments that intersect with the instance bounding box (denoted with subscript I).

Semantic Occupancy Metrics: We follow prior work [11, 16, 47, 61] and employ *average precision* (AP) and *soft intersection over union* (Soft-IoU) as our occupancy metrics. AP measures how well the occupancy predictions are ranked, without taking into account the magnitude, whereas Soft-IoU focuses on the calibration of the predicted probabilities. When we average the AP over multiple time horizons (for evaluating forecasting), we refer to it as mean AP (mAP). We compute these metrics at the scene-level (AP$_S$, Soft-IoU$_S$) and instance-level (AP$_I$, Soft-IoU$_I$). Since the negative portion of the ray is much larger than the positive portion, we sample an balanced number of negative and positive query points for evaluation, ensuring the metric better differentiates between accurate and inaccurate models.

Comparison against state-of-the-art: Tab. 1 benchmarks our method on the task of 4D semantic occupancy prediction methods for the `vehicle` class (as defined in [18]). DIO-D —the version of our model where the source points in the prompts come from the object detector HEDNet [53]— performs best across all metrics. We emphasize that by employing a detector to propose source points, our method avoids any ground-truth leakage, ensuring a fair evaluation in comparison to the baselines. Remarkably, our method relying on instances from an imperfect detector —which may contain mistakes like false positives and false negatives— outperforms the other methods which are trained end-to-end as a single model. Compared to the Oracle Box, DIO achieves superior instance-level metrics, consistent with the fact that the oracle relies on boxes that do not accurately represent the true geometry of objects, highlighting the importance of predicting fine-grained shapes. We also observe that all methods achieve better metrics at the scene-level than at the instance-level. This is because there are easy negatives in the scene-level evaluation (e.g., areas near the SDV where there is clearly no occupancy as otherwise there would be dense LiDAR returns). In contrast, instance-level metrics lack such negatives, as all negative samples are confined within the object bounding boxes, placing them in close proximity to positive occupancy regions.

Qualitative results: In Fig. 4, we show exemplary visualizations with scene occupancy, instance occupancy, and scene flow. In the instance occupancy, DIO captures a variety of different instances such as pedestrians, vehicles, cones, and signs. In the scene occupancy, DIO is able to capture fine-grained details about the background.

Robustness to different prompt sources: We investigate the impact of different source points: empty (DIO-∅), label centroids (DIO-L), label centroids with uniform noise inside the bounding box similar to training (DIO-NL), detection centroids coming from off-the-shelf detector HEDNet [53] (DIO-D). The results in Tab. 2 show that our method is robust to the type of source points used. Here, DIO-NL also performs similar to DIO-L underlining robustness to inaccurate source points.

Open-Set Generalization: To benchmark the open-set capabilities of different methods, we evaluate all methods

	Completion ($t = t_0$)								Forecasting ($t > t_0$)							
	L1 (m) ↓	AbsRel (%) ↓	$\text{HR}_{\text{NFCD},\tau}$ (%) ↑			$\text{HR}_{\text{Asy. NFCD},\tau}$ (%) ↑			L1 (m) ↓	AbsRel (%) ↓	$\text{HR}_{\text{NFCD},\tau}$ (%) ↑			$\text{HR}_{\text{Asy. NFCD},\tau}$ (%) ↑		
			$\tau = 0.5$	$\tau = 2.0$	$\tau = 5.0$	$\tau = 0.5$	$\tau = 2.0$	$\tau = 5.0$			$\tau = 0.5$	$\tau = 2.0$	$\tau = 5.0$	$\tau = 0.5$	$\tau = 2.0$	$\tau = 5.0$
4D-Occ [19]	-	-	-	-	-	-	-	-	20.67	60.05	-	-	-	4.81	18.24	30.07
ImplicitO-4D [16]	3.36	10.01	42.95	63.17	70.75	37.97	56.64	61.41	4.94	15.56	30.02	46.38	52.61	31.18	48.23	57.90
UNO [18]	3.42	11.71	-	-	-	71.06	89.00	95.33	4.90	20.07	-	-	-	59.29	85.16	94.12
DIO-NL	**0.79**	**2.92**	**71.20**	**88.79**	**95.53**	**75.60**	**91.09**	**95.58**	**1.13**	**4.75**	**63.71**	**82.70**	**90.34**	**72.36**	**89.45**	**94.56**

Table 5. LiDAR completion and forecasting evaluated on objects.

on occupancy for the held-out categories, which were never seen during training: `Bicyclist`, `Stopsign`, `Dog`. We selected these three classes to represent various properties of unseen objects, including those with distinct dynamics (`Bicyclist`, `Dog`), static objects (`Stopsign`), and rare, small-sized objects (`Dog`), based on statistics from [34]. The results are presented in Tab. 3. We observe that DIO achieves the best overall performance, which demonstrates the generality of the learned occupancy representation. Interestingly, because no semantic understanding is required from DIO-$\varnothing$, it achieves the best shape completion, which shows both in the present and in the future. In contrast, we see that DIO-NL is stronger at forecasting due to information implicitly encoded in the source point prior, which can guide the forecasting process.

4.3. LiDAR Point Cloud Prediction

LiDAR prediction metrics: In line with [18, 19], we evaluate model performance using depth L1 error (L1), depth relative L1 error (AbsRel), Chamfer Distance (CD) and Near Field Chamfer Distance (NFCD). The AbsRel metric represents the L1 error normalized by the ground-truth depth. While L1 and AbsRel measure the predicted performance along a ray, NFCD and CD capture the similiarity between point clouds. We evaluate at the present time $t = 0.0\,\text{s}$, to capture the *point cloud completion* capabilities of different models, and in the future to capture the forecasting capabilities of each method.

To enable a fair comparison between object-based occupancy models (DIO, [16]) and scene occupancy model ([18]), which also predict background points, we also employ a *object-based* evaluation, also measuring how well our method can distinguish instances. For each evaluation frame, we select a random object in the ROI to run the metrics on. We can compare the LiDAR point of the ground truth point cloud that lie within the label box, and compare them the models' predicted point cloud to achieve our object-centric metrics.

However, using the NFCD with symmetric distance calculation between predicted and GT point clouds would favour object-based methods, which do not predict background points. Hence, we do not employ the NFCD computation for methods which also predict background points in the object-based evaluation setting. To still have a measure of point cloud similarity for all methods, we employ the asymmetric NFCD (Asy. NFCD) only computing distances from the GT point cloud to the predicted point cloud. Inspired by miss rate metrics in motion forecasting [62], we employ a HitRate $\text{HR}_\tau = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}(\text{NFCD}(X_i, Y_i) < \tau)$, denoting the number of predicted point clouds for which the NFCD is smaller than a set of previously defined thresholds τ with units in m^2. $\text{HR}_{\text{NFCD},\tau}$ and $\text{HR}_{\text{Asy. NFCD},\tau}$ denote the HitRate for the NFCD and Asy. NFCD.

Comparison against state-of-the-art: Tab. 4 reports a quantitative comparison of the state-of-the unsupervised models with DIO-$\varnothing$ in the task of *scene LiDAR point cloud forecasting* and achieves the best performance on the public leaderboard[1] in the ranking metric (L1). That underlines DIO-$\varnothing$s learned general occupancy representation and its utility as a world model.

Tab. 5 shows quantitative results of our models against the state-of-the-art in the object-based evaluation setting. It is apparent that DIO-NL outperforms all other models by a large margin across all metrics (e.g., *the completion L1 is 4.3 times smaller than the strongest baseline*), even in those ($\text{HR}_{\text{Asy. NFCD}}$) which favour baselines predicting the whole scene point cloud. This demonstrates the strong object-based understanding of the decomposed occupancy model.

5 Conclusion

In this paper, we propose DIO, a decomposable implicit occupancy world model capable of jointly predicting 4D scene occupancy, decomposed occupancy, and flow in continuous 3D space and time. We demonstrate that DIO outperforms state-of-the-art supervised and unsupervised occupancy baselines in occupancy prediction and LiDAR point cloud prediction experiments. Additionally, it exhibits strong open-set prediction capabilities, indicating its ability to learn a generalized occupancy representation of both individual instances and the scene. We showcase that our model can be flexibly prompted to decompose scene occupancy into instance-based occupancy. While our method requires fewer labeled data than related work, it still relies on bounding box labels during training. Future work could explore combining our approach with unsupervised object detection techniques to reduce labeling requirements further.

[1] https://eval.ai/web/challenges/challenge-page/1977/leaderboard/4662

References

[1] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *CoRL*, 2018. 1

[2] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. Pnpnet: End-to-end perception and prediction with tracking in the loop. In *CVPR*, 2020. 6

[3] Xinshuo Weng, Ye Yuan, and Kris Kitani. Ptp: Parallelized tracking and prediction with graph neural networks and diversity sampling. *RA-L*, 2021.

[4] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *IROS*, 2020.

[5] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *ECCV*, 2020.

[6] Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *ICCV*, 2019.

[7] Boris Ivanovic, Karen Leung, Edward Schmerling, and Marco Pavone. Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *RA-L*, 2020.

[8] Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *ICCV*, 2021.

[9] Alexander Cui, Sergio Casas, Abbas Sadat, Renjie Liao, and Raquel Urtasun. Lookout: Diverse multi-future prediction and planning for self-driving. In *ICCV*, 2021.

[10] Christopher Diehl, Tobias Klosek, Martin Krueger, Nils Murzyn, Timo Osterburg, and Torsten Bertram. Energy-based potential games for joint motion forecasting and control. In *Proceedings of The 7th Conference on Robot Learning*, pages 3112–3141, 2023. 1

[11] Reza Mahjourian, Jinkyu Kim, Yuning Chai, Mingxing Tan, Ben Sapp, and Dragomir Anguelov. Occupancy flow fields for motion forecasting in autonomous driving. *IEEE Robotics and Automation Letters*, 7(2):5639–5646, 2022. 1, 7

[12] Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412, 2021. 1, 2

[13] Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *ECCV*, 2020. 2

[14] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, 2020. 2

[15] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *ICCV*, 2021. 2

[16] Ben Agro, Quinlan Sykora, Sergio Casas, and Raquel Urtasun. Implicit occupancy flow fields for perception and prediction in self-driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1379–1388, 2023. 2, 5, 6, 7, 8

[17] Christopher Diehl, Timo Sebastian Sievernich, Martin Krüger, Frank Hoffmann, and Torsten Bertram. Uncertainty-aware model-based offline reinforcement learning for automated driving. *IEEE Robotics and Automation Letters*, pages 1167–1174, 2023. 1

[18] Ben Agro, Quinlan Sykora, Sergio Casas, Thomas Gilles, and Raquel Urtasun. Uno: Unsupervised occupancy fields for perception and forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14487–14496, June 2024. 1, 2, 3, 4, 5, 6, 7, 8

[19] Tarasha Khurana, Peiyun Hu, David Held, and Deva Ramanan. Point cloud forecasting as a proxy for 4d occupancy forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1116–1124, 2023. 1, 2, 3, 4, 5, 6, 7, 8

[20] Luis Roldao, Raoul de Charette, and Anne Verroust-Blondet. 3d semantic scene completion: a survey. pages 1978—-2005, 2022. 2

[21] Anh-Quan Cao, Angela Dai, and Raoul de Charette. Pasco: Urban 3d panoptic scene completion with uncertainty awareness. In *CVPR*, 2024. 2

[22] Christopher Diehl, Eduard Feicho, Alexander Schwambach, Thomas Dammeier, Eric Mares, and Torsten Bertram. Radar-based dynamic occupancy grid mapping and object detection. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2020. 2

[23] Fangqiang Ding, Xiangyu Wen, Yunzhou Zhu, Yiming Li, and Chris Xiaoxuan Lu. Radarocc: Robust 3d occupancy prediction with 4d imaging radar, 2024. 2

[24] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, pages 533–549. Springer, 2022. 2

[25] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17853–17862, June 2023. 2, 6

[26] Songen Gu, Wei Yin, Bu Jin, Xiaoyang Guo, Junming Wang, Haodong Li, Qian Zhang, and Xiaoxiao Long. Dome: Taming diffusion model into high-fidelity controllable occupancy world model, 2024. 2

[27] Junyi Ma, Xieyuanli Chen, Jiawei Huang, Jingyi Xu, Zhen Luo, Jintao Xu, Weihao Gu, Rui Ai, and Hesheng Wang. Cam4DOcc: Benchmark for Camera-Only 4D Occupancy Forecasting in Autonomous Driving Applications. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[28] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d

occupancy world model for autonomous driving. In *ECCV*, 2024.

[29] Erxin Guo, Pei An, You Yang, Qiong Liu, and An-An Liu. Fsf-net: Enhance 4d occupancy forecasting with coarse bev scene flow for autonomous driving, 2024. 2

[30] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019. 2

[31] Benedikt Mersch, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. Self-supervised point cloud prediction using 3d spatio-temporal convolutional networks. In *Conference on Robot Learning*, pages 1444–1454. PMLR, 2022. 2, 5

[32] Xinshuo Weng, Jianren Wang, Sergey Levine, Kris Kitani, and Nicholas Rhinehart. Inverting the pose forecasting pipeline with spf2: Sequential pointcloud forecasting for sequential pose forecasting. In *Conference on robot learning*, pages 11–20. PMLR, 2021.

[33] Xinshuo Weng, Junyu Nan, Kuan-Hui Lee, Rowan McAllister, Adrien Gaidon, Nicholas Rhinehart, and Kris M Kitani. S2net: Stochastic sequential pointcloud forecasting. In *European Conference on Computer Vision*, pages 549–564. Springer, 2022. 5

[34] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023. 2, 8

[35] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, page 381–389, 2006. 2

[36] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, page 61–70, Goslar, DEU, 2006. Eurographics Association.

[37] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737, 2018.

[38] Karel Zimmermann, Tomáš Petrícek, Vojtech Šalanský, and Tomáš Svoboda. Learning for active 3d mapping. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1548–1556, 2017. 2

[39] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2443, 2017. 2

[40] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3101–3109, 2021.

[41] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9296–9306, 2019. 2

[42] Yuqi Wang, Yuntao Chen, Xingyu Liao, Lue Fan, and Zhaoxiang Zhang. Panoocc: Unified occupancy representation for camera-based 3d panoptic segmentation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17158–17168, 2024. 2

[43] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Niessner. Scan2cad: Learning cad model alignment in rgb-d scans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[44] Tianjia Shao, Weiwei Xu, Kun Zhou, Jingdong Wang, Dongping Li, and Baining Guo. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*, 31(6), 2012. 2

[45] Ji Hou, Angela Dai, and Matthias Nießner. Revealnet: Seeing behind objects in rgb-d scans. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2095–2104, 2020. 2

[46] Yinyu Nie, Ji Hou, Xiaoguang Han, and Matthias Nießner. Rfd-net: Point scene understanding by semantic instance reconstruction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4606–4616, 2021. 2

[47] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 2, 7

[48] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, October 2023. 2

[49] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 2

[50] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019.

[51] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021.

[52] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds. In *European Conference on Computer Vision*, pages 426–442. Springer, 2022.

[53] Gang Zhang, Junnan Chen, Guohuan Gao, Jianmin Li, and Xiaolin Hu. HEDNet: A hierarchical encoder-decoder network for 3d object detection in point clouds. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023. 2, 7

[54] Sourav Biswas, Sergio Casas, Quinlan Sykora, Ben Agro, Abbas Sadat, and Raquel Urtasun. Quad: Query-based interpretable neural motion planning for autonomous driving. *arXiv preprint arXiv:2404.01486*, 2024. 2

[55] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018. 3

[56] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017. 3

[57] Haotian Tang, Shang Yang, Zhijian Liu, Ke Hong, Zhongming Yu, Xiuyu Li, Guohao Dai, Yu Wang, and Song Han. Torchsparse++: Efficient training and inference framework for sparse convolution on gpus. In *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2023. 3

[58] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9308–9316, 2019. 3, 4

[59] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018. 6

[60] Sergio Casas, Ben Agro, Jiageng Mao, Thomas Gilles, Alexander Cui, Thomas Li, and Raquel Urtasun. Detra: A unified model for object detection and trajectory forecasting. In *ECCV*, 2024. 6

[61] Jinkyu Kim, Reza Mahjourian, Scott Ettinger, Mayank Bansal, Brandyn White, Ben Sapp, and Dragomir Anguelov. Stopnet: Scalable trajectory and occupancy prediction for urban autonomous driving. In *ICRA*, 2022. 7

[62] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B. Choy, Philip H. S. Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2165–2174, 2017. 8