Latent Gaussian Splatting for 4D Panoptic Occupancy Tracking

Maximilian Luz[1], Rohit Mohan[1], Thomas Nürnberg[2], Yakov Miron[2,3], Daniele Cattaneo[1], and Abhinav Valada[1]

***Abstract*—Capturing 4D spatiotemporal scene structure is crucial for the safe and reliable operation of robots in dynamic environments. However, existing approaches typically address only part of the problem: they either provide coarse geometric tracking via bounding boxes or detailed 3D occupancy estimates that lack explicit temporal association and instance-level reasoning. In this work, we present Latent Gaussian Splatting (LaGS) for 4D Panoptic Occupancy Tracking (4D-POT). We revisit the underlying representation and model 3D features as a sparse set of feature-bearing Gaussians. These act as dynamic, volume-oriented keypoints that enable spatially continuous, distance-weighted aggregation of multi-view features before being splatted into a voxel grid for decoding. This point-centric formulation enables flexible, data-dependent receptive fields and long-range spatial interactions that are difficult to capture with local and dense voxel-based operators. A hierarchical Gaussian representation further enables multi-scale reasoning by combining global context from coarse super-points with fine-grained detail from higher-resolution streams. Extensive experiments on Occ3D nuScenes and Waymo demonstrate state-of-the-art performance for 4D-POT. We provide code and models at https://lags.cs.uni-freiburg.de/.**

***Index Terms*—Semantic Scene Understanding, Reconstruction, Autonomous Driving**

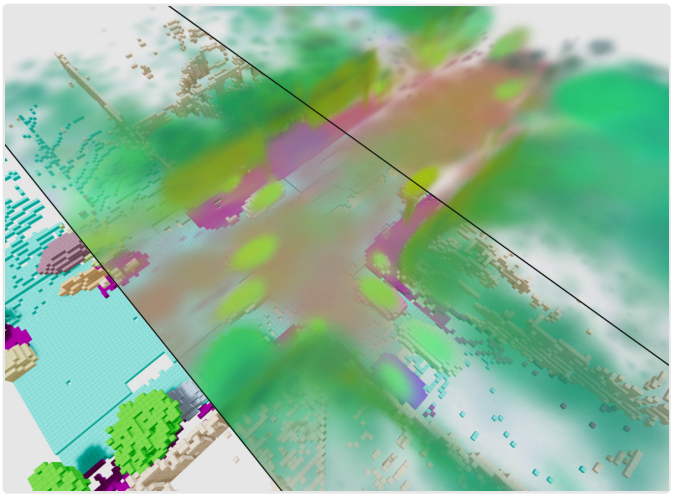

Fig. 1. Illustration of our latent Gaussian representation. Bottom left: panoptic voxel predictions. Center: latent Gaussian features colorized via principal component analysis, splatted to 2D, and overlaid over voxel predictions. Top right: latent Gaussians.

I. Introduction

CAMERA-based 4D panoptic occupancy tracking (4D-POT) [1] combines dense geometric reconstruction, semantic understanding, and temporal consistency within a unified framework. This holistic formulation promises a comprehensive representation of dynamic environments, addressing key shortcomings of prior paradigms [2], [3]. Classical box-based tracking methods model scenes using coarse cuboids [4]–[6], lacking fine-grained geometry and volumetric semantics, while standard 3D occupancy prediction [7]–[9] typically operates per frame, producing dense voxel grids without instance identities or temporal association. In contrast, 4D-POT assigns a semantic class to every voxel while simultaneously distinguishing and tracking individual object instances across time, bridging geometric fidelity and instance-level awareness.

Despite the relative novelty of 4D-POT, existing approaches largely follow a direct composition of mask-based 3D occupancy prediction with query-based end-to-end 3D multi-object tracking (3D MOT) [1], typically relying on dense voxel feature encoders adapted from 3D occupancy prediction. While effective, these designs inherit limitations of dense voxel representations, including restricted receptive fields and limited flexibility in modeling long-range spatial interactions.

In this work, we revisit the underlying representation and propose to model 3D features as a sparse set of feature-bearing Gaussians. Concretely, we treat 3D Gaussians as dynamic, volume-oriented keypoints that carry feature embeddings and spatial extent, enabling spatially continuous, distance-weighted aggregation of sparse features into dense voxel representations via Gaussian splatting. This yields a point-centric formulation in which features are aggregated in a sparse latent space before being splatted back into a voxel grid for downstream processing. We term this concept *Latent Gaussian Splatting* (LaGS), illustrated in Fig. 1. By replacing dense voxel feature volumes with a sparse Gaussian representation, LaGS enables more flexible, data-dependent receptive fields and long-range spatial interactions that are difficult to capture with local voxel-based operators such as COTR [9]. Furthermore, by leveraging hierarchical Gaussian representations, we construct coarse super-points that support global reasoning while preserving fine-grained detail in higher-resolution point sets. This hierarchical formulation enables multi-scale reasoning, where coarse representations capture global context while fine streams preserve local detail, providing a principled balance between local detail and global context within a unified encoder design.

Beyond this representation, we identify an additional chal-

Manuscript received 21 December 2025; accepted 13 May 2026. This article was recommended for publication by Associate Editor G. Dubbelman and Editor M. Vincze upon evaluation of the reviewers' comments.

The work of Abhinav Valada was supported in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant 539134284, in part by EFRE under Grant FEIH_2698644, and in part by the State of Baden-Württemberg. This work was supported by the Bosch Research Collaboration on AI-Driven Automated Driving.

[1]Department of Computer Science, University of Freiburg, Germany.

[2]Bosch Research, Robert Bosch GmbH, Germany.

[3]Leon H. Charney School of Marine Sciences, University of Haifa, Israel.

Corresponding author: luz@cs.uni-freiburg.de

lenge in mask-based 4D-POT: the increased number of instance queries introduces substantial memory demands, as query-based tracking typically requires many more queries than the number of expected objects [10]–[13]. This is further exacerbated when backpropagating gradients across multiple frames, as in prior end-to-end tracking approaches [12], [13]. However, we observe that propagating detached queries and optimizing each frame independently maintains strong performance while significantly reducing resource requirements and allowing capacity to be reallocated to the decoder transformer.

In summary, our primary contributions are five-fold: **(1)** We introduce *Latent Gaussian Splatting*, a sparse, feature-bearing Gaussian representation for dense 3D/4D prediction that enables spatially continuous aggregation and more flexible, expressive feature encoding. **(2)** We design a hierarchical Gaussian encoder that enables multi-scale reasoning by combining coarse global context with fine-grained local detail within a unified point-centric formulation. **(3)** We re-evaluate existing 4D-POT metrics, address inconsistencies in prior implementations, and extend evaluation to nuScenes [14] with newly generated ground-truth annotations and reproduced baselines. **(4)** We achieve state-of-the-art performance on Occ3D nuScenes and Waymo, improving by up to $+15.2$ p.p. STQ on nuScenes. **(5)** We provide code and models at https://lags.cs.uni-freiburg.de/.

II. Related Work

In this section, we first briefly review methods for 3D occupancy prediction, followed by a discussion of sparse representations addressing the dense 3D nature of the task, and finally cover 4D panoptic occupancy tracking.

3D Occupancy Prediction: While the majority of 3D perception in autonomous driving still relies on bounding-box representations, the promise of finer geometric details has led to significant advances in 3D semantic occupancy prediction in recent years [9], [15]. Notably, Occ3D [16] extends the widely adopted nuScenes [14] and Waymo [17] datasets to semantic occupancy prediction, providing more challenging dynamic scenes. Current state-of-the-art thereon [7]–[9] largely follows MaskFormer [18], employing a mask-based transformer decoder and posing the task as 3D segmentation. SparseOcc [19] and PaSCo [20] show that, akin to 2D panoptic image segmentation, such a decoder can be adapted straightforwardly to predict 3D panoptic occupancy.

Sparse 3D Occupancy Prediction: A major practical challenge in 3D semantic occupancy prediction is the dense 3D nature of the task. Several works explore compressed representations [9], [21] or exploit the intrinsic sparsity of the task by focusing only on occupied regions [19], [22], yet remain largely retain voxel-aligned. Notably, GaussianFormer [23] proposes 3D Gaussians for representing both occupied and free space, followed by GaussianFormer-2 [24], only modeling occupied space for a sparse object-centric approach. Both methods splat the predicted 3D Gaussians to the final voxel grid for the task output. We argue this creates an opportunity to integrate ideas from recent point-based 3D perception methods [25], [26]. Specifically, Gaussians can be treated as superpoints, yielding a sparse latent representation that alleviates costly dense 3D processing while remaining convertible to dense voxel representations via splatting. Our novelty, therefore, lies in treating splatting as an intermediate step in the encoder, splatting features instead of appearance or semantics. Moreover, the sparse representation enables more effective reasoning across larger and more flexible neighborhoods, improving information exchange, 2D-to-3D lifting, and scalability. While sparsity has also been explored in classical object detection [27], the prevalent architectures remain dense.

4D Panoptic Occupancy Tracking: By extending 3D panoptic occupancy prediction temporally, TrackOcc [1] introduces the task of 4D panoptic occupancy tracking (4D-POT). While it can be understood as a 3D extension to video panoptic segmentation (VPS) [28], its dense 3D nature requires specifically tailored approaches due to its inherent computational complexity. To this end, TrackOcc incorporates ideas from end-to-end 3D multi-object tracking [10]–[13], ensuring temporally consistent instance assignments by propagating instance-mask-queries across frames. While TrackOcc relies on MUTR3D [12] for query-based tracking, we instead build on PF-Track [13], leveraging its lightweight temporal query refinement, and further streamline our approach to allow for more than one decoder layer, significantly improving tracking performance.

III. Method

4D panoptic occupancy tracking requires the tight integration of geometric reconstruction, semantic understanding, and temporal association (Section III-A). Our approach (Fig. 2) combines mask-based 3D occupancy prediction [1], [9] with a query-based 3D multi-object tracking decoder following the tracking-by-attention paradigm [13]. We largely adopt existing designs for image encoding, lifting, and voxel-based feature extraction, and build upon a PETR-style tracking decoder [13], [29]. Our core novelty lies in the representation and 3D encoder design. Specifically, we introduce a latent Gaussian feature representation, replacing dense voxel processing with a hierarchical point-based representation, and propose Serialized Multi-Stream Attention (SMSA) for efficient interaction across hierarchical point sets. We further extend Gaussian splatting to feature aggregation and adapt the decoder to jointly operate on voxel and Gaussian features.

Our pipeline first lifts multi-view image features into voxels (Section III-B), refines them via the latent Gaussian encoder (Section III-C), and decodes them into semantic and instance masks (Section III-D), with tracking performed via propagated queries (Section III-E). Further details on training and inference are provided in Sections III-F and III-G.

A. Task Definition

Given a consecutive sequence of multi-view images $\boldsymbol{I} = \{I_t^j, I_{t-1}^j, \dots I_{t-T}^j\}_{j=1}^N$, where t is the current timestep, T the sequence length, and N the number of camera views, camera-based 4D panoptic occupancy tracking requires joint prediction of occupancy, semantics, and instance associations for each 3D voxel surrounding the ego-vehicle [1]. Specifically, the task requires predicting $(c_{p,t}, i_{p,t})$ for each voxel position p, where $c_{p,t}$ represents the semantic class and occupancy

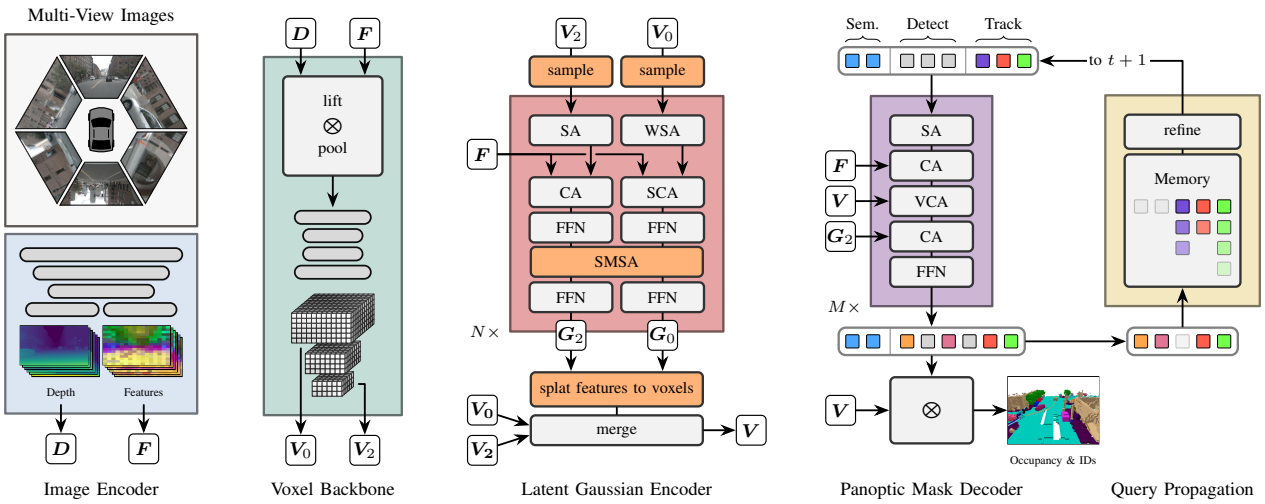

Fig. 2. Architecture overview (left to right). An image encoder (□) produces image features $\boldsymbol{F}$ and depth $\boldsymbol{D}$. Features are lifted via depth to a 3D volume and further encoded into a 3D voxel feature pyramid ($\boldsymbol{V}_0$, $\boldsymbol{V}_2$, □). Our latent Gaussian encoder (□) samples points from the pyramid volumes and processes them in a coarse (left) and fine (right) stream, using self-attention (SA), windowed self-attention (WSA), cross-attention (CA), spatial cross-attention (SCA), and feed-forward networks (FFN). Our novel Serialized Multi-Stream Attention (SMSA) facilitates information exchange between streams. Refined points are decoded as Gaussians ($\boldsymbol{G}_0$, $\boldsymbol{G}_2$) and splatted back to a 3D feature volume, which is then refined to the final voxel volume $\boldsymbol{V}$. Our transformer decoder (□) then decodes this volume into semantic and instance masks using volume cross-attention (VCA) for efficient query-to-3D-volume attention. Tracking is facilitated by the tracking-by-attention paradigm. We refine track queries by spatio-temporal reasoning before passing them onto the next frame (□).

state (i.e., category including *unknown/other* and *free*) and $i_{p,t}$ the associated instance ID at timestep t. Notably, semantic predictions incorporate both tracked *thing* and non-tracked *stuff* classes, while instance IDs are only assigned for tracked *thing* classes. Intrinsic and extrinsic parameters of all cameras as well as ego-motion of the vehicle are assumed to be known.

B. Image Encoder, Explicit Lifting, and Voxel Backbone

Following prior work [1], [9], we begin by extracting multi-view image features $\boldsymbol{F}$ with an off-the-shelf image encoder (Fig. 2, □). A small head is trained to predict a binned depth distribution $\boldsymbol{D}$ independently for each image, which is then used to lift the image features (explicitly) to 3D via the outer product $\boldsymbol{P} = \boldsymbol{F} \otimes \boldsymbol{D}$, creating a pseudo point cloud. This is subsequently pooled to a 3D voxel grid representation and further refined into a voxel feature pyramid ($\boldsymbol{V}_0$ at output resolution, $\boldsymbol{V}_2$ at $1/4$ output resolution) using standard 3D convolutions [1], [9] (Fig. 2, □).

C. Latent Gaussian Occupancy Feature Encoder

Instead of further refining the dense voxel representation, as done in COTR [9], we introduce a representation of 3D features as a set of latent Gaussians, treating them as volumetric keypoints. This point-centric formulation allows us to leverage insights from recent point transformer architectures [26]. By serializing points via space-filling curves, we obtain larger and more flexible receptive fields than dense 3D operations (e.g., voxel self-attention in COTR [9]), improving scalability and replacing fixed neighborhoods with learned relations via positional encoding. Concretely, we represent latent Gaussian queries as feature-coordinate pairs $\boldsymbol{G}_s = \{(\boldsymbol{q}_{s,i}, \boldsymbol{c}_{s,i})\}_{i=1}^{k_s}$, where $\boldsymbol{q}_{s,i} \in \mathbb{R}^C$ denotes the feature embedding and $\boldsymbol{c}_{s,i} \in \mathbb{R}^3$ its 3D coordinate. The coordinates are inherited from voxel locations during the sampling process described below. Our encoder (Fig. 2, □) follows a hierarchical design with two streams: a high-resolution (fine) stream $\boldsymbol{G}_0$ with more points for capturing local detail, and a coarse stream $\boldsymbol{G}_2$ with fewer points for efficient global reasoning.

Voxel-to-Gaussian Sampling: We convert voxel features into point representations via a four-step process: (1) compute voxel-wise priority scores, (2) suppress spatial redundancy via max-pooling, (3) sample indices using multinomial sampling, and (4) map sampled indices back to the original voxel grid. Given a voxel feature volume $\boldsymbol{V}_s$, we use feature magnitudes as scalar priorities and apply 3D max-pooling (kernel size $n = 2$) to reduce spatial redundancy, yielding pooled priorities and an argmax index map. We normalize the pooled priorities to obtain a probability distribution and sample k_s indices without replacement using multinomial sampling, which are mapped back to the original resolution via the argmax map. The resulting voxel features $\boldsymbol{q}_{s,i}$ and their spatial coordinates $\boldsymbol{c}_{s,i}$ define the initial point set $\boldsymbol{G}_s$, enabling the transition from dense voxel grids to sparse point-based representations.

Point-Based Refinement (Fine Stream): Following Point Transformer V3 [26], we serialize points via a space-filling curve applied to the coordinates $\boldsymbol{c}_{0,i}$ and process the resulting linearized sequence with a multi-layer transformer. We employ sliding-window self-attention (WSA) with overlapping windows of size w, centered at each position i, such that attention is restricted to local neighborhoods via masking. This ensures that memory scales linearly with the number of points k_0, while maintaining effective local context. To efficiently integrate image features, we use spatial cross-attention (SCA) [30], projecting points into image space and applying 2D deformable attention.

Hierarchical Extension (Coarse Stream): We extend the

encoder by introducing hierarchical streams derived from a 3D feature pyramid ($\boldsymbol{V}_0$, $\boldsymbol{V}_2$). Instead of sampling from a single resolution, we construct streams $\boldsymbol{G}_0$ and $\boldsymbol{G}_2$ independently from different scales and process them independently through their respective transformer blocks. While the fine stream ($\boldsymbol{G}_0$) employs sliding-window self-attention together with spatial cross-attention, enabling efficient local refinement, the coarse stream ($\boldsymbol{G}_2$) uses full self-attention (SA) and full image-point cross-attention (CA), leveraging its lower point count to capture more global context.

Interaction between streams is performed exclusively via a novel Serialized Multi-Stream Attention (SMSA) module, enabling controlled information exchange between resolutions while preserving independent per-stream processing. SMSA merges all streams and, similar to the single-stream case, re-serializes the combined set using a space-filling curve, producing a unified linear sequence. Sliding-window self-attention (WSA) is then applied to this sequence, enabling local interactions and cross-stream information exchange as points from different streams are interleaved based on spatial proximity. Positional encoding is applied via Fourier feature embeddings of the coordinates and added to the input keys and queries prior to attention. After attention, features are restored to the original ordering using the inverse serialization permutation and split back into the respective streams. SMSA naturally handles varying point densities across streams without requiring explicit balancing.

Gaussian Feature Aggregation: Our downstream architecture depends on a 3D feature volume. To this end, we extend 3D Gaussian occupancy splatting [24] from occupancy prediction to feature aggregation. For each latent query $\boldsymbol{q}_i$, we predict centers $\boldsymbol{\mu}_i = \boldsymbol{c}_i + \boldsymbol{\delta}_i$ (with offsets $\boldsymbol{\delta}_i$), covariances Σ_i (from scale and rotation), opacities α_i, and feature embeddings $\boldsymbol{e}_i \in \mathbb{R}^C$. From these, occupancy o and voxel features $\boldsymbol{f}$ are computed as

$$o(\boldsymbol{x}) = 1 - \prod_i \Big(1 - \exp\big(-\tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2_{\Sigma_i^{-1}}\big)\Big), \quad (1)$$

$$\boldsymbol{f}(\boldsymbol{x}) = o(\boldsymbol{x}) \cdot \frac{\sum_i \alpha_i \mathcal{G}_i(\boldsymbol{x}) \boldsymbol{e}_i}{\sum_i \alpha_i \mathcal{G}_i(\boldsymbol{x})}, \quad (2)$$

where $\mathcal{G}_i(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_i, \Sigma_i)$ is the 3D Gaussian PDF, and $\|\boldsymbol{v}\|^2_M = \boldsymbol{v}^\intercal M \boldsymbol{v}$ denotes Mahalanobis distance. Note that the decoded Gaussians only represent occupied space. Hierarchical streams are concatenated before aggregation. Subsequently, we merge the aggregated feature volume with the initial feature pyramid volumes, essentially introducing skip connections and creating a U-Net-like structure.

D. Panoptic Mask Decoder

We build upon the PETR-style mask transformer decoder (Fig. 2, ▢) used in PF-Track [13], employing detection queries for instance (*thing*) segmentation and semantic queries for global instance-less (*stuff*) masks. We retain the query-based transformer formulation, but extend it with additional cross-attention to 3D voxel features via deformable attention and, for hierarchical Gaussian encoding, to refined coarse Gaussian features. Furthermore, we introduce a voxel mask prediction head for volumetric decoding. After refinement by the transformer, queries are decoded into a mask embedding and semantic class scores. Similar to MaskFormer [18], binary occupancy masks are computed via a dot product between mask embeddings and voxel features.

E. Tracking, Query Propagation, and Refinement

To facilitate tracking, we propagate decoded detection queries across frames following the tracking-by-attention paradigm [10], [11]. As in TrackOcc [1], we separate detection (*thing*) and semantic (*stuff*) queries, propagating only detection queries temporally while re-initializing semantic queries for each frame. The propagated queries, together with newly initialized detection and semantic queries, form the decoder input for the subsequent frame. New detections initialize tracks while propagated queries maintain existing ones. To further improve tracking performance, we adopt the spatio-temporal refinement module of PF-Track [13] (Fig. 2, ▢) without modification, refining queries based on memory (past) and trajectory prediction (future). Predicted trajectories are used to fill gaps for intermittently missed or low-confidence detections.

F. Training and Supervision

We supervise our approach on multiple levels. Following COTR [9], we supervise the depth prediction for explicit feature lifting in the encoder via sparse depth from LiDAR. Further, we add a small head to decode semantic scores for each Gaussian, splatting them to a semantic voxel grid for direct supervision via a cross-entropy loss, akin to GaussianFormer-2 [24]. The decoder is supervised via both semantic and instance masks as well as box predictions (akin to center supervision in TrackOcc [1]). For detection queries, we use bipartite matching, considering both predicted boxes and masks to assign ground truth instances. Once a ground-truth instance has been assigned, it is kept across all subsequent training frames. For semantic queries, we use bipartite matching with masks only. Supervision of the spatio-temporal refinement module follows PF-Track [13].

To enable temporal supervision, we train on short multi-frame sequences. We prevent linear scaling of gradient buffers by detaching track and detection queries after decoding and before refinement, meaning gradients from subsequent frames still flow back to the refinement stage, but not the decoder itself, decoupling individual frames.

G. Inference

The raw predictions of our approach are class scores $\boldsymbol{c}_q$ and occupancy mask scores $m_{q,\boldsymbol{x}}$ for both instance and *stuff* queries. We score queries via the maximum class score, i.e., $s_q = \|\boldsymbol{c}_q\|_\infty$, filter out inactive ones via a threshold, and compute the dominant query $\hat{q}_{\boldsymbol{x}} = \arg\max_q \{s_q \cdot m_{q,\boldsymbol{x}}\}$ for each voxel $\boldsymbol{x}$. In contrast to previous methods [1], [18], [31], we compute dominant queries independently for instance and *stuff* classes, merging both afterward by overriding the *stuff* predictions with instance ones.

IV. EXPERIMENTS

The effectiveness of our proposed approach is demonstrated by extensive evaluation. To this end, we first describe the datasets used for benchmarking in Section IV-A and the evaluation metric in Section IV-B, including necessary revisions to correct existing inaccuracies. Brief descriptions of the baselines follow in Section IV-C, along with implementation and training details in Section IV-D. Benchmarking results are presented in Section IV-E, followed by a detailed ablation study of the architectural components (Section IV-F), concluding with qualitative comparisons (Section IV-G). All evaluations are performed on the respective validation splits.

A. Datasets

We evaluate our approach on both nuScenes [14] and Waymo [17], with 3D occupancy ground-truth provided by Occ3D [16]. For both datasets, the spatial range is bounded from $-40\,\mathrm{m}$ to $40\,\mathrm{m}$ for x and y and from $-1\,\mathrm{m}$ to $5.4\,\mathrm{m}$ for z, with a voxel size of $0.4\,\mathrm{m}$ in each axis, resulting in a voxel grid resolution of $200\times200\times16$. For Waymo, we follow TrackOcc [1] and subsample the dataset at every 5th frame, yielding 789 training scenes and 202 validation scenes with 40 samples each. For nuScenes, we train and evaluate on the full dataset with 700 training and 150 validation scenes, with each scene containing around 40 samples.

To obtain 4D panoptic occupancy labels for nuScenes, we assign instance IDs to all thing-class voxels of the Occ3D semantic occupancy data by using the ground-truth box labels. Specifically, instance IDs are assigned based on the intersecting box of the same class. Ambiguities, such as voxels being intersected by none or multiple boxes, are resolved by choosing the closest instance. Notably, the assigned instance IDs are consistent with the nuScenes box instance IDs, facilitating direct box-to-voxel correspondence. We make this data preprocessing available alongside our code. For Waymo, we rely on the data provided by TrackOcc [1].

B. Evaluation Metrics

Following TrackOcc [1], we adapt the Segmentation and Tracking Quality (STQ), originally introduced for video panoptic segmentation [32], to 4D occupancy prediction and tracking. STQ is defined as the geometric mean of Segmentation Quality (SQ) and Association Quality (AQ),

$$\mathrm{STQ} = \sqrt{\mathrm{SQ}\cdot\mathrm{AQ}}, \tag{3}$$

where SQ is the classical mean intersection over union (mIoU) of semantic occupancy prediction [16], [33].

AQ represents the mean IoU between each ground-truth and predicted 4D tube, where each individual IoU is weighted by the respective intersected fraction to facilitate a soft assignment and bound the AQ score by one. Mathematically, we define the 4D panoptic occupancy predictions $P = \{(p,t,i,c)\}$ as the set over tuples of 3D voxel position p, time step t, instance ID i, and class c. Equivalently, we define G as the set of ground-truth tuples. From this, we derive $P_i = \{(p,t,i) \mid (p,t,i,c) \in P\}$ and equivalently G_i as the instance prediction and ground-truth for instance i. Finally, we define

$$\mathrm{AQ} = \frac{1}{|I_G|}\sum_{i\in I_G}\frac{1}{|G_i|}\sum_{j\in I_P}|G_i\cap P_j|\cdot\frac{|G_i\cap P_j|}{|G_i\cup P_j|}, \tag{4}$$

where $I_G = \{i \mid (p,t,i,c) \in G\}$ is the set of ground-truth instance IDs and I_P is the set of predicted instance IDs.

Extending over TrackOcc, we further propose the AQ_1 metric for single-frame panoptic assessment. AQ_1 is constructed analogously to Eq. (4), with the exception that we do not consider tracking tubes but only single-frame instances for matching, i.e., enforce $\forall(p_1,t_1,i_1,c_1),(p_2,t_2,i_2,c_2) \in G : t_1 \neq t_2 \Rightarrow i_1 \neq i_2$. STQ_1 follows analogously again as the geometric mean over the already non-temporal mIoU and AQ_1. This avoids the well-documented shortcomings of the panoptic quality (PQ) metric [32]. In addition to STQ, AQ, and mIoU/SQ, we also use the binary IoU to assess the quality of the binary free/non-free occupancy prediction.

Following Occ3D [16], we evaluate only on visible regions (using the "camera" mask). Notably, AQ does not depend on any class assignments, and SQ does not depend on any instance information, strictly separating semantic and instance segmentation between SQ and AQ. While mathematically sound, however, we find that the metric implementations of TrackOcc [1] are flawed: they solely consider areas occupied in the ground-truth data and ignore regions marked as free space. This skews the metric significantly, as any false positives in known free space are disregarded, essentially only counting true positives and false negatives. Mathematically, this is equivalent to applying Eq. (4) to $P_i' = P_i \cap M$ and $G_i' = G_i \cap M$, where M is a mask indicating occupied space in G. We therefore reconstruct and re-evaluate the baselines presented by TrackOcc, as well as TrackOcc itself, and provide corrected implementations alongside our code.

C. Baselines

To ensure fair comparison under the revised metrics, we reimplement the baselines proposed by TrackOcc [1] within a unified framework. As the original baseline implementations are not publicly available, we build upon TrackOcc and adapt it into a single-frame 3D panoptic occupancy model by removing query propagation, which serves as a common backbone for all methods. On top of this shared setup, we implement different tracking strategies: MinVIS-style bipartite matching of queries across frames based on cosine similarity [34], a CTVIS-based variant with learned association via contrastive training [35], heuristic box extraction followed by tracking with AB3DMOT [36], and IoU-based matching inspired by 4D-LiDAR panoptic segmentation [37]. For post-processing-based approaches, we tune hyperparameters independently for each dataset. To evaluate the effectiveness of temporal association, we additionally introduce a Per-Frame baseline, which assigns independent instance IDs at each timestep. This provides a lower bound for tracking performance, below which temporal association degrades instance segmentation quality. All baselines share identical voxel encoders, image backbones (ResNet-50), and training protocols.

TABLE I
4D-POT PERFORMANCE ON OCC3D NUSCENES.

Approach	STQ	AQ	STQ_1	AQ_1	mIoU all	mIoU things	mIoU stuff	IoU
Per-Frame	9.0	2.5	21.8	14.7	32.5	26.4	41.2	63.2
MinVIS† [34]	11.8	4.3	21.8	14.7	32.5	26.4	41.2	63.2
CTVIS† [35]	11.4	3.9	22.5	15.4	33.0	27.0	41.5	63.8
4D-LCA† [37]	12.5	4.8	21.8	14.7	32.5	26.4	41.2	63.2
AB3DMOT† [36]	13.1	5.3	21.8	14.7	32.5	26.4	41.2	63.2
TrackOcc‡ [1]	12.2	4.7	19.7	12.1	32.1	25.3	41.8	63.7
LaGS-2s R50 (Ours)	**27.4**	**20.9**	**31.4**	**27.2**	**36.2**	**31.4**	**43.0**	**64.4**
TrackOcc V99‡ [1]	15.5	6.5	23.8	15.3	37.0	31.0	45.6	67.0
LaGS-2s V99 (Ours)	**32.3**	**25.6**	**36.4**	**32.5**	**40.7**	**37.0**	**46.0**	**67.3**

†: Baselines reproduced by us. ‡: Official code trained for nuScenes/V99 backbone.

TABLE II
4D-POT PERFORMANCE ON OCC3D WAYMO.

Approach	STQ	AQ	STQ_1	AQ_1	mIoU all	mIoU things	mIoU stuff	IoU
Per-Frame	11.9	4.7	—	—	30.0	32.7	29.3	—
MinVIS† [34]	15.0	7.5	—	—	30.0	32.7	29.3	—
CTVIS† [35]	16.4	9.3	—	—	28.9	31.9	28.1	—
4D-LCA† [37]	16.2	8.7	—	—	30.0	32.7	29.3	—
AB3DMOT† [36]	18.0	10.8	—	—	30.0	32.7	29.3	—
TrackOcc‡ [1]	20.2	13.8	—	—	29.4	29.7	29.4	—
LaGS-2s R50 (Ours)	**26.2**	**22.0**	—	—	**31.1**	**36.6**	**29.6**	—
TrackOcc V99	21.7	14.6	—	—	32.3	32.5	32.3	—
LaGS-2s V99 (Ours)	**30.1**	**25.8**	—	—	**35.3**	**41.3**	**33.6**	—
Per-Frame	9.1	4.0	18.1	15.6	20.9	21.9	20.6	58.4
MinVIS† [34]	11.0	5.8	18.1	15.6	20.9	21.9	20.6	58.4
CTVIS† [35]	12.5	7.3	18.7	16.5	21.2	22.3	20.9	59.6
4D-LCA† [37]	12.1	7.0	18.1	15.6	20.9	21.9	20.6	58.4
AB3DMOT† [36]	13.3	8.5	18.1	15.6	20.9	21.9	20.6	58.4
TrackOcc‡ [1]	15.2	10.7	18.1	15.2	21.6	21.9	**21.5**	**60.1**
LaGS-2s R50 (Ours)	**18.4**	**15.3**	**20.6**	**19.2**	**22.0**	**24.3**	21.4	59.8
TrackOcc V99* [1]	16.7	11.7	20.0	16.7	23.9	24.7	**23.7**	**62.9**
LaGS-2s V99 (Ours)	**21.2**	**18.3**	**23.6**	**22.6**	**24.6**	**28.1**	**23.7**	61.5

†: Baselines reproduced by us. ‡: Results reproduced with official code and weights. *: Official code adapted for and trained with V99 backbone. Gray text: metrics as implemented by TrackOcc [1], ignoring false positives.

TABLE III
3D OCCUPANCY PREDICTION PERFORMANCE ON OCC3D NUSCENES.

Approach	Image Backbone	mIoU	IoU
TPVFormer [21]	ResNet-50	34.2	66.8
SurroundOcc [8]	ResNet-101	34.6	65.5
OccFormer [7]	ResNet-50	37.4	70.1
BEVDet4D [39]	ResNet-50	39.3	73.8
BEVDet4D + COTR [9]	ResNet-50	44.5	**75.0**
BEVDet4D + COTR [9]	Swin-B	**46.2**	74.9
BEVDet4D + COTR* [9]	ResNet-50	34.6	66.3
TrackOcc‡ [1]	ResNet-50	32.1	63.7
LaGS-2s (Ours)	VoVNetV2-99	**40.7**	**67.3**

Top: 3D semantic occupancy prediction methods without instance segmentation or tracking. Bottom: 4D panoptic occupancy tracking methods requiring instance-level modeling and temporal association. Gray row/*: Restricted baseline without longterm aggregation and stereo-depth feature lifting, representing the closest comparable setting. ‡: Official code trained for nuScenes.

D. Implementation and Training Details

Following PF-Track [13], we use VoVNetV2-99 [38] as image backbone with input resolution 800×320 for nuScenes and 704×256 for Waymo, and additionally evaluate ResNet-50 for fair comparison. Image-to-3D lifting follows the BEVDet4D variant of COTR [9] and TrackOcc [1], with stereo and temporal aggregation disabled due to memory constraints. All methods are trained on 8 NVIDIA L40 GPUs for 24 epochs with batch size 1 per GPU. Baselines use a single-frame adaptation of TrackOcc and follow its training procedure. Our method is trained in two stages (12 epochs pre-training + 12 epochs tracking) using a one-cycle schedule (base LR 2×10^{-4}, 500 warmup steps) [13]. The encoder employs a two-stream latent Gaussian representation with 512 coarse and 8192 fine points, embedding dimension C=256, and window size w=1024, using 3D Hilbert curve serialization. For VoVNetV2, we use a single coarse image feature scale for spatial cross-attention, whereas for ResNet-50 we follow TrackOcc and use the full feature pyramid. Both encoder and decoder use 4 transformer layers. Our model has 76.0M parameters (ResNet-50) and 118.9M parameters (VoVNetV2-99). Inference takes $374.0\,\mathrm{ms}$ (FP32) and $352.1\,\mathrm{ms}$ (BF16), with peak memory usage of $11.2\,\mathrm{GB}$ and $5.7\,\mathrm{GB}$, respectively. We observe no measurable accuracy difference between FP32 and BF16 inference.

E. Comparison with State-of-the-Art

Results for Occ3D nuScenes and Waymo are presented in Tables I and II. Across both datasets and backbones, LaGS consistently outperforms prior approaches in both overall (STQ) and tracking-specific (AQ) metrics. Using a ResNet-50 (R50) backbone, our method improves over TrackOcc by up to $+15.2$ p.p. STQ and $+16.2$ p.p. AQ on nuScenes, and $+3.4$ p.p. STQ and $+4.6$ p.p. AQ on Waymo, with further gains observed for the stronger VoVNetV2-99 (V99) backbone. Notably, improvements are most pronounced for instance-related metrics (AQ, AQ_1, mIoU-*things*), indicating more effective modeling of object structure and re-identification, while performance on *stuff* classes remains comparable. The larger gains on nuScenes can be attributed to its more diverse set of *thing* classes, where fine-grained instance discrimination is critical. In contrast, Waymo contains more diverse *stuff* classes (e.g., bicycle, tree trunk), which may require higher-resolution representations (i.e., more Gaussians) to model adequately.

Further, LaGS reduces the performance gap between 4D panoptic occupancy tracking and 3D semantic occupancy prediction (mIoU) (Table III). We emphasize that methods in the upper part of the table address only 3D semantic occupancy and do not perform instance segmentation or temporal tracking, resulting in significantly lower memory requirements. In contrast, 4D-POT requires modeling instance masks and temporal associations, necessitating more queries and higher computational cost. Consequently, both TrackOcc [1] and our method operate under a constrained setting without long-term temporal aggregation or stereo-based lifting, making the gray-shaded BEVDet4D+COTR variant the closest comparable baseline. The remaining gap to the full BEVDet4D+COTR model can likely largely be attributed to its use of additional components, such as temporal aggregation and multi-frame ("stereo") lifting, and the absence of instance-level modeling.

TABLE IV
ABLATION STUDY ON THE LATENT GAUSSIAN ENCODER.

Type	Gaussians	Layers	STQ	AQ	AQ_1	mIoU all	mIoU thing	mIoU stuff	IoU
COTR	—	1	31.5	25.0	31.6	39.6	36.3	44.4	65.1
LaGS	8192	1	31.5	25.1	32.0	39.6	36.2	44.5	64.9
COTR	—	2	31.5	24.9	31.6	39.8	36.6	44.4	64.9
LaGS	8192	2	31.7	25.2	32.2	39.9	36.4	45.0	65.3
COTR	—	4	31.4	24.9	31.5	39.6	36.3	44.4	65.2
LaGS	2048	4	32.1	25.5	32.3	40.4	36.8	45.5	66.3
LaGS	4096	4	32.0	25.4	32.2	40.3	36.8	45.3	65.7
LaGS	8192	4	31.9	25.4	32.5	40.1	36.6	44.9	65.3
LaGS-2s	{512, 2048}	4	**32.3**	**25.7**	**32.7**	40.6	36.8	45.9	67.0
LaGS-2s	{512, 4096}	4	32.1	25.6	32.4	40.4	36.5	45.9	67.1
LaGS-2s	{512, 8192}	4	**32.3**	25.6	32.5	**40.7**	**37.0**	**46.0**	**67.3**

Metrics reported on the nuScenes validation split with VoVNetV2-99 as backbone. Gray indicates the original COTR encoder. Blue indicates our chosen configuration.

TABLE V
ABLATION STUDY ON THE DECODER TRANSFORMER.

Layers	Refine	STQ	AQ	AQ_1	mIoU all	mIoU things	mIoU stuff	IoU
1	✓	28.2	20.6	26.8	38.7	34.2	**45.1**	64.9
4	✗	30.5	24.1	**32.4**	38.5	34.3	44.4	65.0
4	✓	**31.5**	**25.0**	31.6	**39.6**	**36.3**	44.4	**65.1**

Performance on the nuScenes validation split reported using the COTR encoder with 1 encoder layer (cf. Table IV, first entry). Blue indicates our chosen configuration.

F. Ablation Studies

Latent Gaussian Encoder: We evaluate the proposed encoder on the nuScenes validation split (Table IV). With a single transformer layer, our method performs comparably to the COTR [9] encoder. As encoder depth increases, however, our approach shows consistent improvements over the voxel-based baseline, particularly for 4-layer configurations, indicating more effective utilization of additional model capacity. Varying the number of Gaussians leads to non-uniform effects across metrics, reflecting a trade-off between representation granularity and efficiency. The hierarchical two-stream design further improves performance and yields the best overall results. This demonstrates that combining fine and coarse representations enhances both local detail and global reasoning, while enabling more effective long-range interactions at coarser resolutions. We observe approximately linear scaling in runtime with both the number of Gaussians and the encoder depth.

Decoder Transformer: The decoder plays a central role in instance segmentation and tracking (Table V). While prior work [1], [9] typically employs a single decoder layer, increasing the number of layers substantially improves instance segmentation (mIoU-*things*) and tracking quality (AQ), despite the associated increase in computational cost. This indicates that sufficient decoder capacity is required to fully exploit the improved encoder features. In addition, spatio-temporal refinement further improves performance, particularly for *thing*-class segmentation, indicating that temporal aggregation enhances both semantic consistency and instance association.

Runtime and Scaling: We analyze the computational cost of our approach in Table VI. Runtime increases predictably with both encoder depth and the number of Gaussians, scaling approximately linearly with depth and moderately with the number of points. Compared to the voxel-based COTR encoder, our approach incurs additional cost, primarily due to Gaussian-to-voxel feature splatting (up to approx. 58 ms for 8192 Gaussians). At comparable encoder depth, however, runtimes are similar, and for smaller Gaussian sets (e.g., 2048) our approach can be more efficient, indicating favorable scaling with model capacity. The hierarchical two-stream variant

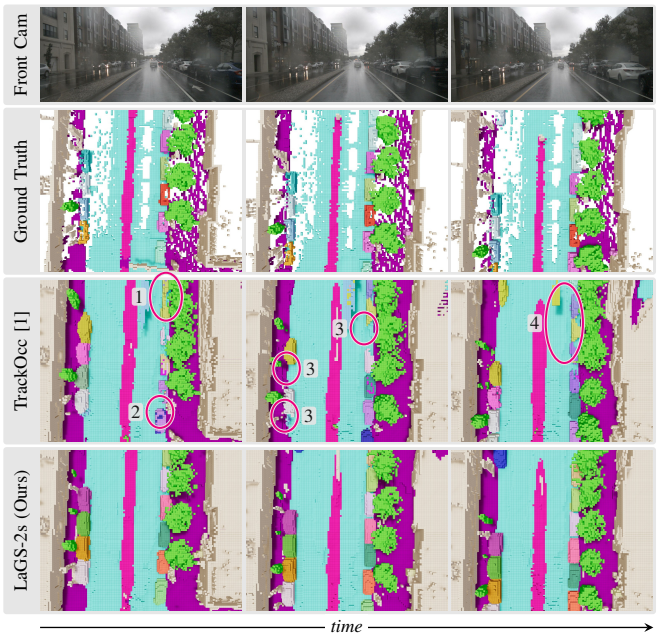

Fig. 3. Qualitative results on the Occ3D nuScenes validation split. Our approach shows clear improvements in (1) instance separation, (2) instance association, (3) missing detections, and (4) underconfident detections.

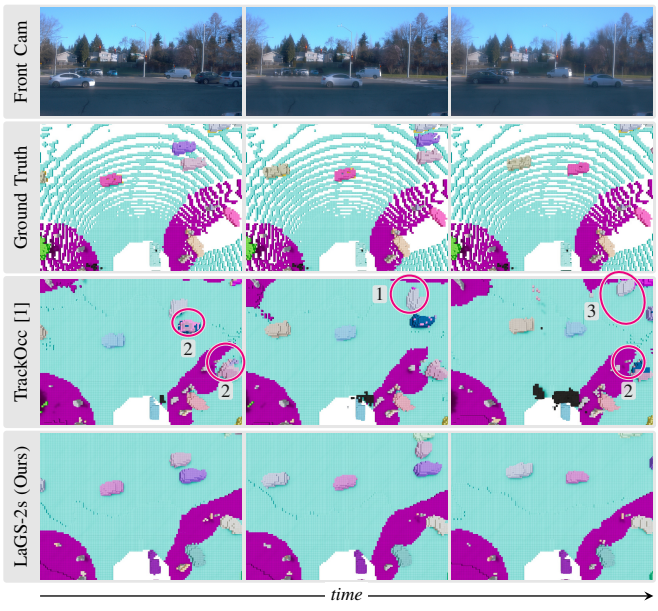

Fig. 4. Qualitative results on the Occ3D Waymo validation split. Our approach shows clear improvements in (1) instance separation, (2) instance association, and (3) ID switches.

TABLE VI
INFERENCE RUNTIME AND SCALING.

Voxel Encoder			Decoder	
Type	Gaussians	Layers	Layers	Runtime (ms)
COTR	—	1	1	215.9
COTR	—	1	4	246.3
LaGS	8192	1	4	304.0
COTR	—	2	4	266.8
LaGS	8192	2	4	312.7
COTR	—	4	4	334.4
LaGS	2048	4	4	289.7
LaGS	4096	4	4	307.0
LaGS	8192	4	4	328.3
LaGS-2s	{512, 8192}	4	4	352.1

Runtimes averaged over the Occ3D nuScenes validation split using VoVNetV2-99 as backbone and BF16 precision. Blue indicates our final configuration.

introduces additional overhead due to cross-stream interaction, but yields the best overall performance. Increasing the number of decoder layers further increases runtime moderately, yet is crucial for strong instance segmentation and tracking. Memory usage remains largely constant across configurations (approx. 5.7 GB in BF16).

G. Qualitative Evaluation

Qualitative results are presented in Fig. 3 (nuScenes) and Fig. 4 (Waymo). We demonstrate improvements across five categories: (1) instance separation, correctly separating nearby instances, (2) instance association, concisely assigning an object to a single instance, (3) missing detections, leading to incorrect free-space predictions, (4) underconfident mask predictions, leading to incomplete instance segmentation, and (5) ID switches, where the same instance ID is wrongly assigned to different objects in subsequent frames. Across both datasets, LaGS produces clearer and more consistent instance masks, while TrackOcc [1] often exhibits underconfident masks or inconsistent instance assignments (cf. highlighted regions). These observations are consistent with the quantitative instance-level gains reported in Tables I and II.

V. CONCLUSION

We proposed a novel Gaussian-driven architecture, outperforming the state-of-the-art in 4D panoptic occupancy tracking. Inspired by recent advancements in Gaussian splatting and point-transformer methods, we designed a novel 3D occupancy feature encoder. Using Gaussians as a 3D representation allows us to convert traditionally dense voxel-grid-based encoders into a sparse, point-wise formulation suitable for transformer-based architectures. This enables more flexible, data-driven information aggregation, effective 2D-to-3D lifting, and improved scalability in representation capacity. By utilizing splatting to convert the sparse representation back into a voxel grid, our encoder can replace classical voxel feature encoders while retaining compatibility, albeit at additional computational cost that scales with the number of Gaussians, offering opportunities for further optimization. Extensive evaluations on both Occ3D nuScenes and Waymo datasets demonstrate the effectiveness of our approach. We hope that this paves the way for further exploration of dynamic, effective, and saliency-driven 3D representations.

REFERENCES

[1] Z. Chen, K. Li, X. Yang, T. Jiang, Y. Li, and H. Zhao, "TrackOcc: Camera-based 4d panoptic occupancy tracking," in *ICRA*, 2025.

[2] M. Luz, R. Mohan, A. R. Sekkat, O. Sawade, E. Matthes, T. Brox, and A. Valada, "Amodal optical flow," in *ICRA*, 2024.

[3] R. Mohan, F. Drews, Y. Miron, D. Cattaneo, and A. Valada, "Up-fuse: Uncertainty-guided lidar-camera fusion for 3d panoptic segmentation," 2026. [Online]. Available: https://arxiv.org/abs/2602.19349

[4] M. Büchner and A. Valada, "3d multi-object tracking using graph neural networks with cross-edge modality attention," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9707–9714, 2022.

[5] C. Lang, A. Braun, L. Schillingmann, and A. Valada, "Self-supervised multi-object tracking for autonomous driving from consistency across timescales," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7711–7718, 2023.

[6] M. Käppeler, Ö. Çiçek, D. Cattaneo, C. Gläser, Y. Miron, and A. Valada, "Bridging perspectives: Foundation model guided bev maps for 3d object detection and tracking," *arXiv preprint arXiv:2510.10287*, 2025.

[7] Y. Zhang, Z. Zhu, and D. Du, "OccFormer: Dual-path transformer for vision-based 3D semantic occupancy prediction," in *ICCV*, 2023.

[8] Y. Wei, L. Zhao, W. Zheng, Z. Zhu, J. Zhou, and J. Lu, "SurroundOcc: Multi-camera 3D occupancy prediction for autonomous driving," in *ICCV*, 2023, pp. 21 729–21 740.

[9] Q. Ma, X. Tan, Y. Qu, L. Ma, Z. Zhang, and Y. Xie, "COTR: Compact occupancy transformer for vision-based 3D occupancy prediction," in *CVPR*, 2024, pp. 19 936–19 945.

[10] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer, "TrackFormer: Multi-object tracking with transformers," in *CVPR*, 2022, pp. 8844–8854.

[11] F. Zeng, B. Dong, Y. Zhang, T. Wang, X. Zhang, and Y. Wei, "MOTR: End-to-end multiple-object tracking with transformer," in *ECCV*, 2022.

[12] T. Zhang, X. Chen, Y. Wang, Y. Wang, and H. Zhao, "MUTR3D: A multi-camera tracking framework via 3D-to-2D queries," in *CVPRW*, 2022.

[13] Z. Pang, J. Li, P. Tokmakov, D. Chen, S. Zagoruyko, and Y.-X. Wang, "Standing between past and future: Spatio-temporal modeling for multi-camera 3D multi-object tracking," in *CVPR*, 2023, pp. 17 928–17 938.

[14] W. K. Fong, R. Mohan, J. V. Hurtado, L. Zhou, H. Caesar, O. Beijbom, and A. Valada, "Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3795–3802, 2022.

[15] R. Mohan, J. V. Hurtado, R. Mohan, and A. Valada, "ForecastOcc: Vision-based semantic occupancy forecasting," *arXiv preprint arXiv:2602.08006*, 2026.

[16] X. Tian, T. Jiang, L. Yun, Y. Mao, H. Yang, Y. Wang, Y. Wang, and H. Zhao, "Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving," in *NeurIPS*, 2023.

[17] 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 *CVPR*, 2020, pp. 2446–2454.

[18] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," in *NeurIPS*, vol. 34, 2021, pp. 17 864–17 875.

[19] P. Tang, Z. Wang, G. Wang, J. Zheng, X. Ren, B. Feng, and C. Ma, "SparseOcc: Rethinking sparse latent representation for vision-based semantic occupancy prediction," in *CVPR*, 2024, pp. 15 035–15 044.

[20] A.-Q. Cao, A. Dai, and R. de Charette, "PaSCo: Urban 3D panoptic scene completion with uncertainty awareness," in *CVPR*, 2024, pp. 14 554–14 564.

[21] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "Tri-perspective view for vision-based 3D semantic occupancy prediction," in *CVPR*, 2023, pp. 9223–9232.

[22] J. Wang, Z. Liu, Q. Meng, L. Yan, K. Wang, J. Yang, W. Liu, Q. Hou, and M.-M. Cheng, "OPUS: Occupancy prediction using a sparse set," in *NeurIPS*, 2024, pp. 119 861–119 885.

[23] Y. Huang, W. Zheng, Y. Zhang, J. Zhou, and J. Lu, "GaussianFormer: Scene as gaussians for vision-based 3D semantic occupancy prediction," in *ECCV*, 2025, pp. 376–393.

[24] Y. Huang, A. Thammatadatrakoon, W. Zheng, Y. Zhang, D. Du, and J. Lu, "GaussianFormer-2: Probabilistic gaussian superposition for efficient 3D occupancy prediction," in *CVPR*, 2025, pp. 27 477–27 486.
[25] Z. Liu, J. Hou, X. Wang, X. Ye, J. Wang, H. Zhao, and X. Bai, "LION: Linear group RNN for 3D object detection in point clouds," in *NeurIPS*, vol. 37, 2024, pp. 13 601–13 626.
[26] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao, "Point transformer v3: Simpler faster stronger," in *CVPR*, 2024, pp. 4840–4851.
[27] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia, "VoxelNeXt: Fully sparse voxelnet for 3D object detection and tracking," in *CVPR*, 2023, pp. 21 674–21 683.
[28] D. Kim, S. Woo, J.-Y. Lee, and I. S. Kweon, "Video Panoptic Segmentation," in *CVPR*, 2020, pp. 9859–9868.
[29] Y. Liu, T. Wang, X. Zhang, and J. Sun, "PETR: Position embedding transformation for multi-view 3D object detection," in *ECCV*, 2022, pp. 531–548.
[30] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, "BEVFormer: Learning bird's-eye-view representation from LiDAR-camera via spatiotemporal transformers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 3, pp. 2020–2036, 2025.
[31] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *CVPR*, 2022, pp. 1290–1299.
[32] M. Weber, J. Xie, M. D. Collins, Y. Zhu, P. Voigtlaender, H. Adam, B. Green, A. Geiger, B. Leibe, D. Cremers, *et al.*, "STEP: Segmenting and tracking every pixel," in *NeurIPS*, 2021.
[33] 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 *ICCV*, 2019, pp. 9297–9307.
[34] D.-A. Huang, Z. Yu, and A. Anandkumar, "MinVIS: A minimal video instance segmentation framework without video-based training," in *NeurIPS*, vol. 35, 2022, pp. 31 265–31 277.
[35] K. Ying, Q. Zhong, W. Mao, Z. Wang, H. Chen, L. Y. Wu, Y. Liu, C. Fan, Y. Zhuge, and C. Shen, "CTVIS: Consistent training for online video instance segmentation," in *CVPR*, 2023, pp. 899–908.
[36] X. Weng, J. Wang, D. Held, and K. Kitani, "3D multi-object tracking: A baseline and new evaluation metrics," in *IROS*, 2020.
[37] M. Aygün, A. Osep, M. Weber, M. Maximov, C. Stachniss, J. Behley, and L. Leal-Taixé, "4d panoptic lidar segmentation," in *CVPR*, 2021, pp. 5527–5537.
[38] Y. Lee, J.-w. Hwang, S. Lee, Y. Bae, and J. Park, "An energy and GPU-computation efficient backbone network for real-time object detection," in *CVPRW*, 2019, pp. 752–760.
[39] J. Huang and G. Huang, "BEVDet4D: Exploit temporal cues in multi-camera 3D object detection," arXiv preprint, arXiv:2203.17054, 2022.