GaussianFlowOcc: Sparse and Weakly Supervised Occupancy Estimation using Gaussian Splatting and Temporal Flow

Supplementary Material

A. Additional Quantitative Results

A.1. Per-Class IoU

For completeness, we provide per-class IoU scores for all evaluated methods on the Occ3D-nuScenes dataset in Tab. A.1. Our proposed GaussianFlowOcc consistently outperforms all prior methods trained with pseudo labels, both with and without depth supervision. The most significant performance gains can be observed in small object classes, such as motorcycles and traffic cones, but also in other categories. Notably, our method achieves the best or second best score across nearly all categories. These results further highlight the effectiveness of our Gaussian-based scene representation and temporal modeling, particularly in handling fine-grained object details that are often challenging for voxel-based approaches.

A.2. Depth Estimation

To further demonstrate the 3D geometric capabilities of GaussianFlowOcc, we report depth estimation results on the nuScenes dataset in Tab. A.2. Our method achieves competitive performance compared to other state-of-the-art self-supervised and weakly supervised approaches on this task. We observe that methods utilizing photometric losses [15, 20, 70] tend to produce slightly more accurate camera depth predictions than our approach. However, it is important to note that while our method may yield marginally lower depth estimation accuracy, it substantially outperforms these methods in 3D occupancy estimation, underscoring the strength of our model in comprehensive 3D scene understanding.

A.3. Training with 3D Voxel Labels

Our main contributions, namely the Gaussian representation and the dynamic object compensation during temporal Gaussian Splatting, specifically target the weakly supervised setting for occupancy estimation. Nonetheless, our model can be trained using 3D voxel labels similar to previous methods [29, 30, 53, 72]. Specifically, we apply a differentiable voxelization to the final Gaussian predictions (see Sec. 3.7), enabling the computation of a cross-entropy loss with ground-truth voxel labels for supervised training. Note that for this experiment, the *Temporal Module* and Gaussian Splatting supervision are not used. In Tab. A.3, we present a comparison on the Occ3D-nuScenes dataset between our method and established fully supervised approaches, following the recommended protocol of not using the camera visibility mask [53]. As shown, our model outperforms prior methods such as BEVFormer [29] and OccFormer [72], while achieving competitive performance with larger models specifically designed for full supervision, including FB-Occ [30] and CTF-Occ [53].

B. Qualitative Results

B.1. Predicted Occupancy

In Fig. B.1, we present qualitative examples of GaussianFlowOcc's predicted occupancy compared to the ground truth, visualized from a third-person perspective. These examples demonstrate our model's ability to precisely reconstruct the 3D scene geometry using 3D Gaussians, despite the absence of explicit 3D supervision. The Gaussians align naturally to capture the shapes and details of objects, effectively representing scene structures without requiring per-scene optimization. This highlights the flexibility and efficiency of our approach in modeling complex environments. Additionally, we provide a set of videos in our GitHub repository (see Appendix D) showcasing inference results on full validation scenes:

- *scene-0346.mp4* and *scene-0790.mp4* visualize the predicted Gaussians from three perspectives: input camera view, third-person view, and Bird's-Eye View.
- *scene-0521_gt.mp4* and *scene-0925_gt.mp4* compare our model's predictions to the ground truth occupancy, both from a third-person perspective.

These results clearly demonstrate the model's ability to accurately represent flat surfaces (e.g., streets and walls), thin objects (e.g., poles), and small objects (e.g., traffic cones). Unlike voxel-based methods, which are constrained by coarse resolution, our approach captures continuous geometric details with far greater fidelity, enabling a more precise and realistic 3D scene understanding.

B.2. Rendered Depth and Semantics

In Fig. B.3, we present examples of rendered depth and semantics obtained by applying Gaussian Splatting to the 3D Gaussians estimated from our trained model. For comparison, we also show the pseudo labels used during training, which are generated by GroundedSAM [45] for semantics and Metric3D [66] for depth. The rendered outputs demonstrate a high level of detail, with most objects accurately segmented. Notably, even thin structures such as trees and pedestrians are well-preserved, showcasing the model's ability to preserve fine geometric details; something voxel-

Table A.1. **Occupancy estimation performance on the Occ3D-nuScenes validation set.** The *Mode* indicates the source of the 2D labels. C refers to camera images, D refers to usage of pseudo depth. Best performing per column and *Mode* section in **bold**, second best in *italics*. Methods that use pseudo labels ignore the *others* and *other flat* classes.

Method	Mode	mIoU	others	barrier	bicycle	bus	car	cons. vehicle	motorcycle	pedestrian	traffic cone	trailer	truck	driv. surf.	other flat	sidewalk	terrain	manmade	vegetation
SelfOcc [20]	C	10.54	-	0.15	0.66	5.46	*12.54*	0.00	0.80	2.10	0.00	0.00	8.25	**55.49**	-	*26.30*	*26.54*	*14.22*	5.60
OccNeRF [70]	C	10.81	-	0.83	0.82	5.13	12.49	*3.50*	0.23	3.10	1.84	0.52	3.90	52.62	-	20.81	24.75	**18.45**	**13.19**
GaussianOcc [70]	C	*11.26*	-	*1.79*	*5.82*	**14.58**	**13.55**	1.30	*2.82*	**7.95**	**9.76**	*0.56*	*9.61*	44.59	-	20.10	17.58	8.61	*10.29*
GaussianFlow (Ours)	C	**14.07**	-	**6.27**	**8.54**	*13.36*	12.38	**4.92**	*10.05*	*6.84*	*8.75*	**1.12**	**10.43**	*53.40*	-	**26.44**	**28.89**	10.39	9.33
GaussTR [24]	C+D	*13.26*	-	*2.09*	*5.22*	*14.07*	**20.43**	**5.70**	*7.08*	*5.12*	*3.93*	*0.92*	**13.36**	*39.44*	-	*15.68*	*22.89*	**21.17**	**21.87**
GaussianFlow (Ours)	C+D	**17.08**	-	**7.23**	**9.33**	**17.55**	*17.94*	*4.50*	**9.32**	**8.51**	**10.66**	**2.0**	*11.80*	**63.89**	-	**31.11**	**35.12**	*14.64*	*12.59*

Table A.2. **Depth estimation results on nuScenes.** The best result is highlighted in **bold**, second best in *italics*.

Method	Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	$\delta < 1.25^2$ ↑	$\delta < 1.25^3$ ↑
SurroundDepth [59]	0.280	4.401	7.467	0.364	0.661	0.844	0.917
SimpleOcc [14]	0.224	3.383	7.165	0.333	*0.753*	*0.877*	0.930
OccNeRF [70]	*0.202*	2.883	*6.697*	0.319	**0.768**	**0.882**	*0.931*
SelfOcc [20]	0.215	2.743	6.706	*0.316*	*0.753*	0.875	**0.932**
GaussianOcc [70]	**0.197**	**1.846**	6.733	**0.312**	0.746	0.873	*0.931*
GaussianFlowOcc (Ours)	0.278	*2.522*	**5.232**	0.389	0.677	0.826	0.898

based approaches often struggle with due to grid resolution constraints.

In Fig. B.4, we further illustrate the effect of our *Temporal Gaussian Splatting* by comparing predicted semantic segmentation maps across consecutive frames: The first row shows the rendered semantics for the current input frame. The second row depicts the segmentation map generated by rendering into the next frame and using the 3D flow estimated by the *Temporal Module* to compensate the object motion. The third row shows the same next-frame projection but without dynamic object handling — simulating the approach taken by previous works. The last row provides the pseudo-semantic labels of the next frame for comparison. The comparison clearly reveals the importance of dynamic object modeling. Without applying the 3D flow (Row 3), the rendered semantics fail to align with the dynamic objects in the next frame, as these objects have already moved. This misalignment would introduce incorrect supervisory signals if used to compute the loss. In contrast, when applying the *Temporal Module*'s flow estimation (Row 2), the rendered semantics better match the pseudo labels of the next frame. This demonstrates that our *Temporal Module* has successfully learned to approximate object motion, improving the alignment of dynamic objects across frames and ensuring more accurate supervision during training.

B.3. Long-term Temporal Flow

We further examine the behavior of the *Temporal Module* by visualizing the estimated flow over a longer temporal horizon, spanning three frames into the past and future. Figure B.2 displays rendered semantic maps that illustrate the qualitative effectiveness of the estimated 3D flow. As is always the case, the model predicts the 3D Gaussians for the current frame t given the images of the current frame (along with the previous frame's Gaussians for temporal fusion). The *Temporal Module* then estimates the motion of objects across each of the surrounding frames, and the Gaussian means are updated according to the predicted flow. The figure compares rendered results using Gaussians moved by the estimated flow with those using static, unmoved Gaussians. The visualizations demonstrate that the model learns reasonable object motion that better aligns with the temporal position of the camera frame, despite not being explicitly trained with motion or scene flow supervision. The *Temporal Module* is optimized solely through consistency constraints between rendered temporal frames and the Gaussians adjusted by the estimated flow. This temporal consistency reduces artifacts during *Temporal Gaussian Splatting*, which is critical for stable training. We note that learning scene flow in this weakly supervised manner remains challenging, and errors are still present in practice. Nevertheless, as shown in Sec. 4.4.3, leveraging the estimated flow to improve temporal consistency leads to a notable increase

Table A.3. **Occupancy estimation performance on the Occ3D-nuScenes validation set using 3D voxel labels for training.** The best result per column is highlighted in **bold**, second best in *italics*.

Method	Backbone	mIoU	others	barrier	bicycle	bus	car	cons. vehicle	motorcycle	pedestrian	traffic cone	trailer	truck	driv. surf.	other flat	sidewalk	terrain	manmade	vegetation
BEVFormer [29]	ResNet-101	23.67	5.0	38.8	10.0	34.4	*41.1*	13.2	16.5	18.2	17.8	*18.7*	*27.7*	49.0	27.7	29.1	25.4	15.4	14.5
OccFormer [72]	ResNet-101	21.93	5.9	30.3	12.3	34.4	39.2	14.4	16.4	17.2	9.3	13.9	26.4	51.0	31.0	34.7	22.7	6.8	7.0
FB-Occ [30]	ResNet-50	*27.09*	0.0	**40.9**	**21.2**	**39.2**	40.8	**20.6**	*23.8*	**23.6**	**25.0**	16.6	26.4	*59.4*	27.6	31.4	29.0	16.7	**18.4**
CTF-Occ [53]	ResNet-101	**28.53**	**8.1**	*39.3*	*20.6*	*38.3*	**42.2**	*16.9*	**24.5**	*22.7*	*21.0*	**23.0**	**31.1**	53.3	*33.8*	*38.0*	*33.2*	**20.8**	*18.0*
GaussianFlowOcc (Ours)	ResNet-50	25.28	*7.6*	25.4	13.4	26.0	25.3	15.3	14.2	12.7	10.7	18.2	20.9	**76.2**	**37.4**	**48.2**	**47.8**	*17.3*	13.2

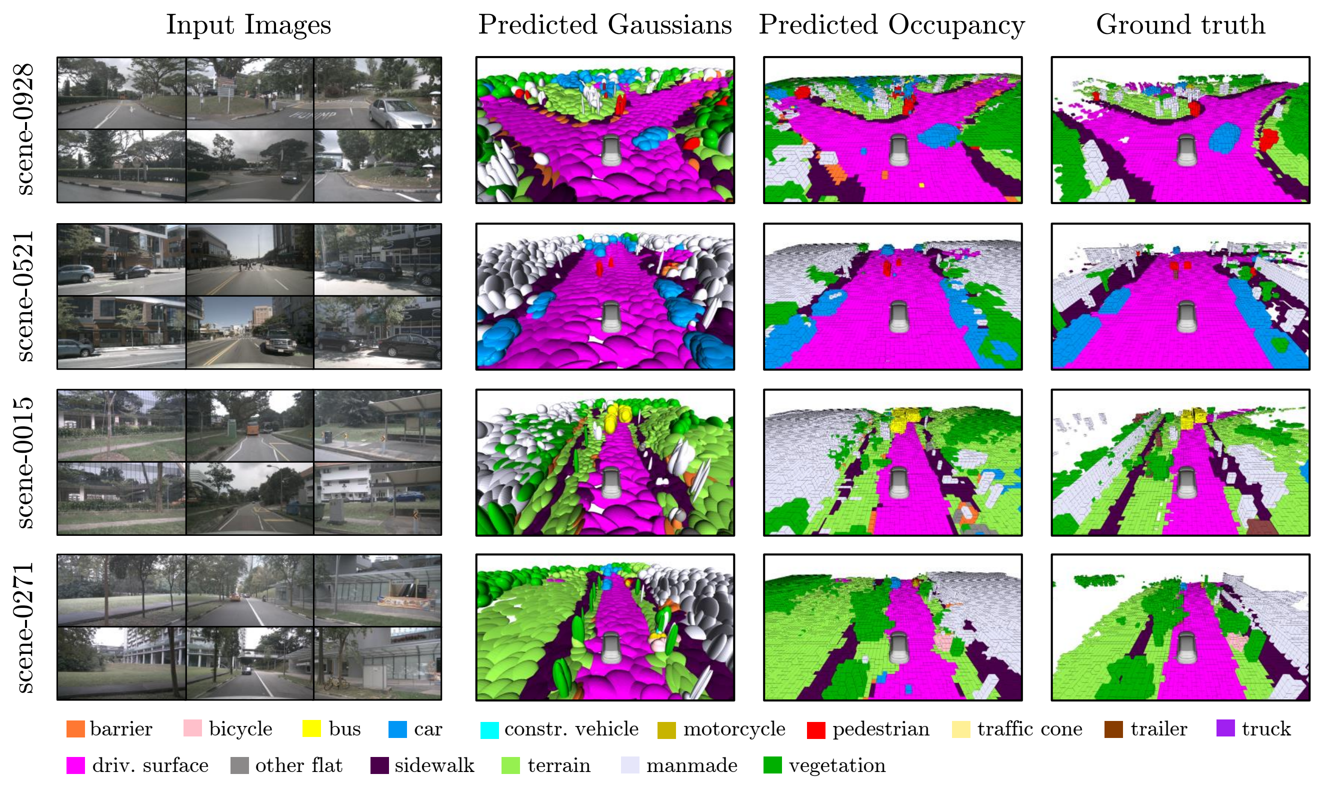

Figure B.1. **Qualitative results on the Occ3D-nuScenes dataset.** We show the estimated 3D Gaussians, how the predictions look when voxelized, and the ground truth occupancy. Best viewed when zoomed in.

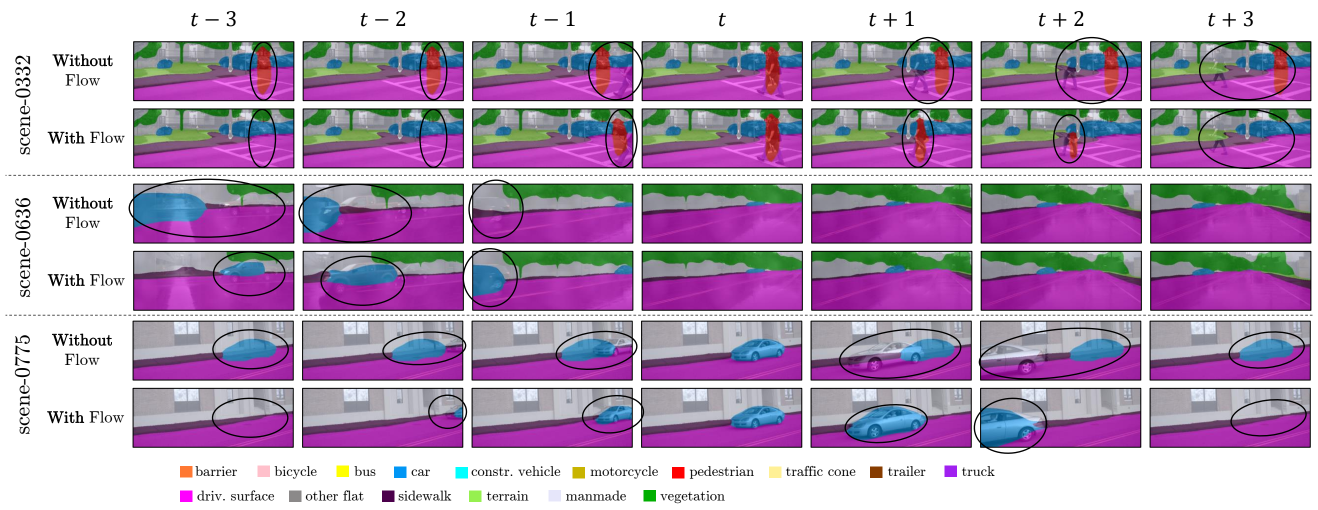

Figure B.2. **Qualitative results on the Occ3D-nuScenes validation set.** Each row depicts the rendered estimated Gaussians when using the estimated temporal flow to move the Gaussians versus not using the flow, for a time horizon of 1.5s (3 frames) into the past and the future, respectively. The Gaussians are predicted for the current frame, and then rendered into adjacent frames. It is visible that the model learns to compensate object motion over multiple frames.

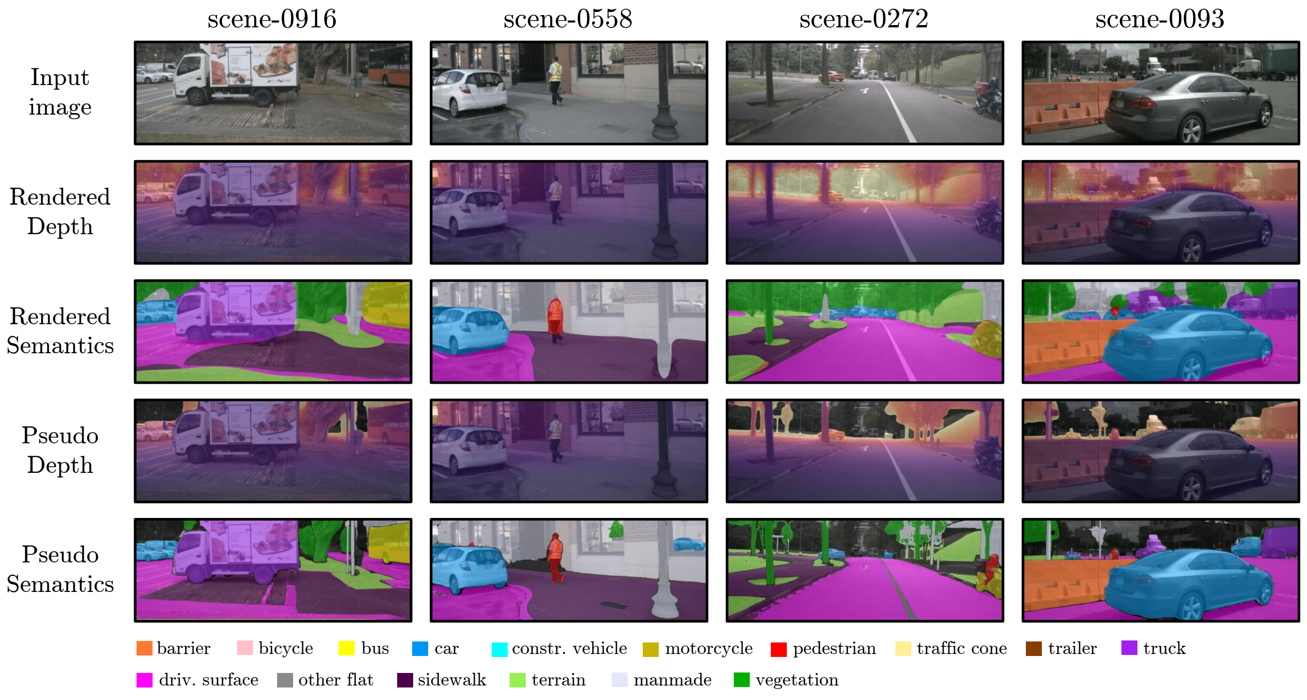

Figure B.3. **Qualitative results on the Occ3D-nuScenes validation set.** Each column shows an *Input Image*, *Rendered Depth* and *Rendered Semantics* generated by using GS to render predicted Gaussians into input cameras (as is done during training). We additionally show the pseudo labels used during training for the relevant sample.

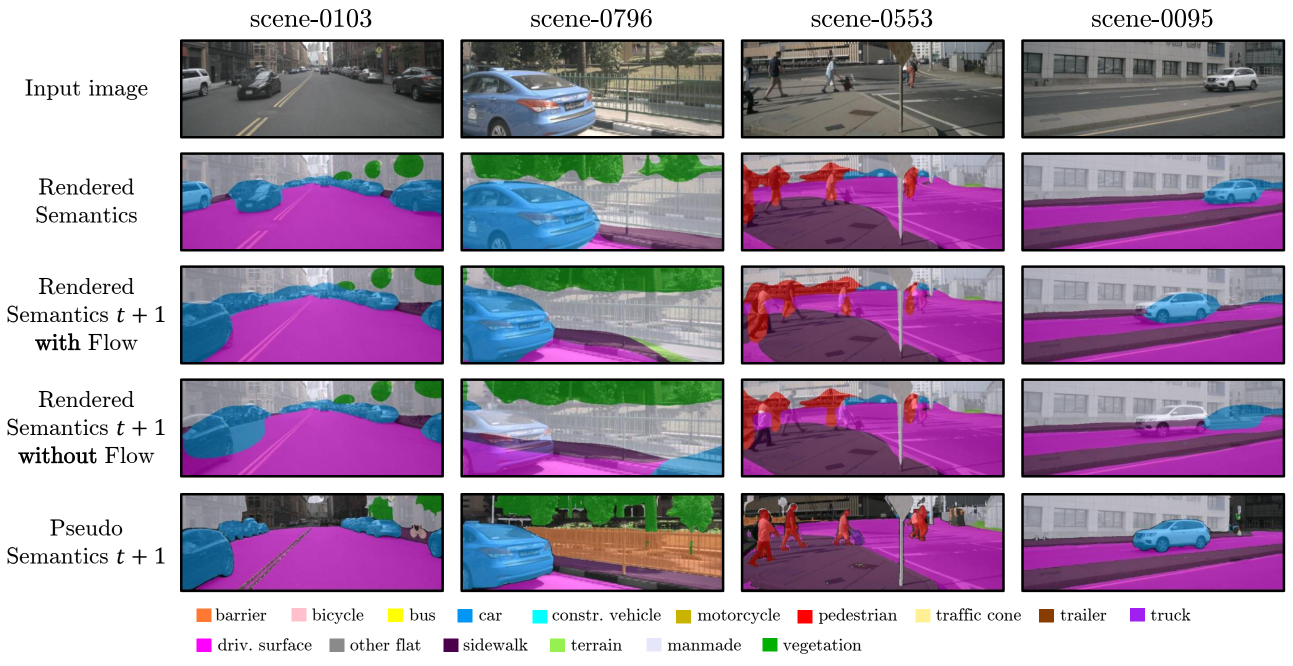

Figure B.4. **Qualitative results on the Occ3D-nuScenes validation set.** Each column shows an input image and the rendered semantics generated by using GS to render predicted Gaussians into the input camera, as well as rendered semantics when rendering the predictions into the next frame, one with using our *Temporal Module* to compensate motion and one without. The last row displays the pseudo labels of the next frame to show how correct motion compensation should look like.

in occupancy estimation performance.

B.4. Comparison to GaussianOcc

To further demonstrate the benefits of our approach, we compare the predicted occupancy between GaussianFlowOcc and GaussianOcc [15] in Fig. B.5. In contrast to GaussianOcc, our model consistently detects and models thin and small objects like trees, traffic signs and poles (*scene-0916* & *scene-0904*). In addition, GaussianFlowOcc suffers a lot less from object bleeding (*scene-0557* & *scene-0775*). It is visible that our method can much better estimate the 3D shape of objects (such as vehicles) avoiding the characteristic depth stretching that often affects 2D-supervised methods [4, 15, 20, 41, 70]. We attribute this improvement to our use of long-term temporal supervision, which enforces geometric consistency over time. This strategy is only viable with an explicit dynamic scene model—highlighting the importance of our integrated motion estimation framework.

C. Voxelization

As explained in the main paper, for benchmarking and comparison, we convert the estimated 3D Gaussian distributions into a voxelized representation. This process begins with defining a voxel grid over the scene. Each Gaussian distribution is then evaluated at the center of every voxel, where its opacity and semantic logits are accumulated to determine the final voxel values. The formulation for this voxelization is as follows:

$$\begin{aligned} v_o(p;\mathcal{G}) &= \sum_{i=1}^{P} G_i(p;\mu_i,s_i,r_i,o_i) \\ &= \sum_{i=1}^{P} \exp\left(-\frac{1}{2}\left(p-\mu_i\right)^T\Sigma_i^{-1}\left(p-\mu_i\right)\right)o_i \end{aligned} \tag{12}$$

$$\begin{aligned} v_c(p;\mathcal{G}) &= \sum_{i=1}^{P} G_i(p;\mu_i,s_i,r_i,c_i) \\ &= \sum_{i=1}^{P} \exp\left(-\frac{1}{2}\left(p-\mu_i\right)^T\Sigma_i^{-1}\left(p-\mu_i\right)\right)c_i, \end{aligned} \tag{13}$$

where Σ_i represents the covariance matrix of each Gaussian, derived from its rotation quaternion and scale parameters. To ensure efficiency in handling large numbers of Gaussians and voxels, we limit the influence of each Gaussian to a fixed local neighborhood. This is justified by the fact that the contribution of each Gaussian typically decays rapidly with distance, making it computationally unnecessary to evaluate its impact on distant voxels. As explained in Appendix A.3, this voxelization operation is fully differentiable, which enables training of our model using 3D voxel labels when available.

D. Source Code

Source code to reproduce the results presented in the paper is available at https://github.com/boschresearch/GaussianFlowOcc. Please follow the instructions in the *README.md* file to install the repository and run the code. We provide instructions to train and evaluate our models on the Occ3D-nuScenes dataset.

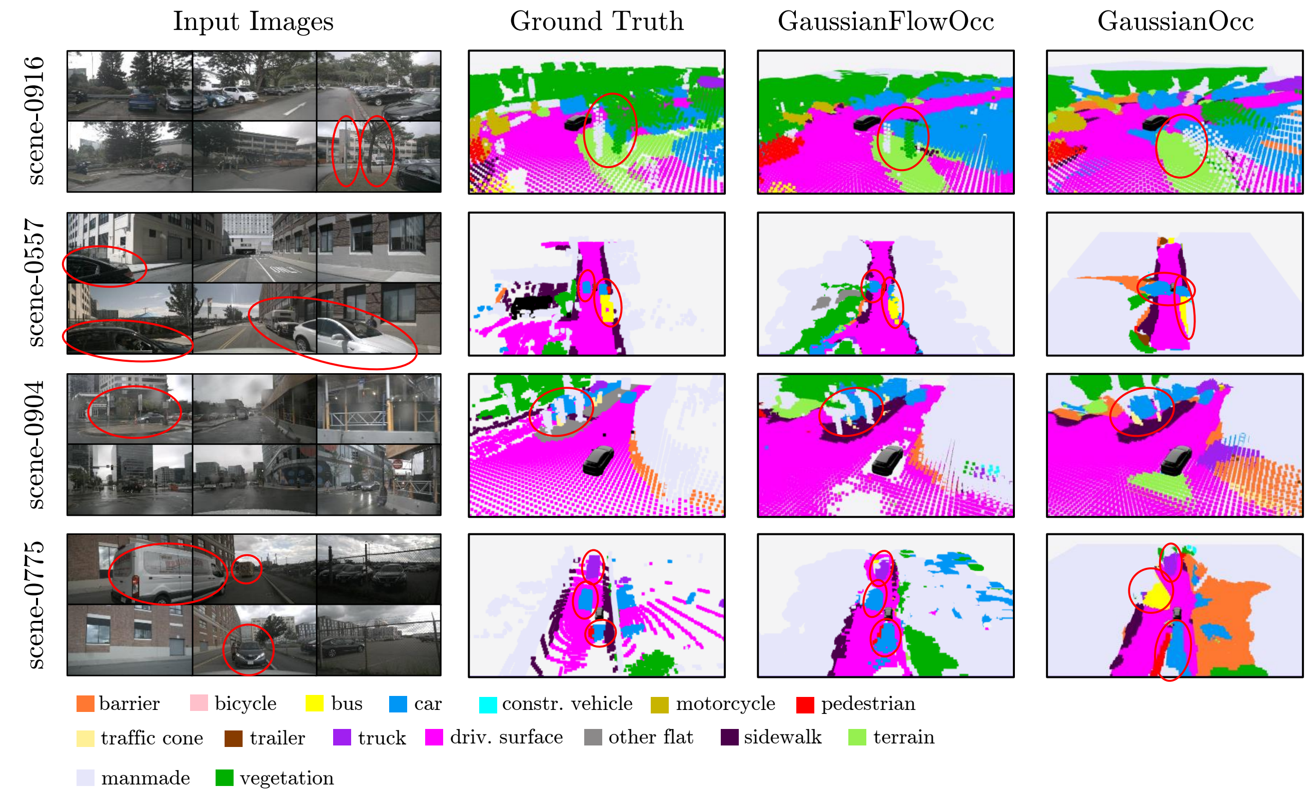

Figure B.5. **Comparing our method to GaussianOcc.** We compare the occupancy estimation results on four samples of different validation scenes of the nuScenes dataset between our method and GaussianOcc [15]. It is visible that our method can better represent thin and small objects, suffers less from object bleeding and can better estimate the complete 3D shape of scene objects.