Multi-Level Neural Scene Graphs for Dynamic Urban Environments

Supplementary Material

This supplementary material provides details on our method, our experimental setup, and more quantitative and qualitative results and comparisons. Please note that we also provide a **demonstration video** alongside this manuscript which showcases our view synthesis results. In Sec. A, we describe our demonstration video. In Sec. B, we provide further details on our benchmark data. In Sec. C, we provide further details on our method. In Sec. D, we provide details on our experimental setup, conduct additional experiments, and show more qualitative comparisons.

A. Demonstration Video

We provide a video of generated, novel trajectories showcasing the versatility of our representation. We use camera intrinsics with slightly lower resolution than in training to generate a standard 16:9 HD video. The video complements our evaluation by showing view angles of, *e.g.*, dynamic objects completely different from training which illustrates more challenging scenarios than the evaluation views sampled along the ego-vehicle trajectories.

In the first part of the video, we compare our method to SUDS [58] on a novel, generated trajectory in the residential area of our proposed benchmark by rendering with a single-sequence, latent vector. We also modulate the time axis with the function $f(x) = x^3$ that has a saddle point around the origin (*i.e.* the middle of the video), thus creating a slow-motion effect. Note that there are color artifacts (see Fig. 8) in the input data that propagate into our reconstruction to a small degree which results in subtle pink artifacts present in the renderings around the road markings.

In the second part of the video, we change the sequence latent vectors to highlight their influence on the scene's appearance. This results in a drastic appearance change in the rendered views. Furthermore, we illustrate in this part how varying dynamic objects can be assembled with the sequence appearances, modulating the object's appearance according to the sequence. For this, we render the same trajectory twice with different latent codes and dynamic objects. We keep the slow-motion effect in this part of the video to showcase the continuous, consistent object trajectories our method can render.

This shows the potential of our method for scenario generation, as we can shuffle objects and model varying environment conditions suitable to real-world applications like closed-loop simulation and mixed reality.

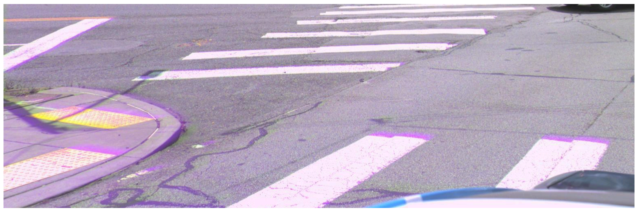

Figure 8. **Color artifacts in Argoverse 2.** We observe that color artifacts are present in the input data, illustrating the complex phenomena that need to be modeled when dealing with vehicle fleet data under varying conditions.

B. Data Details

We describe further details on our benchmark data taken from [61]. The LiDAR is sampled at 10Hz, and the 3D bounding box annotations are annotated with the LiDAR, *i.e.* they are also provided at 10Hz. The cameras are synchronized with the LiDAR which yields seven images at 10Hz for each sequence. Each camera has a resolution of 1550×2048 pixels, where all cameras besides the front camera are oriented in landscape mode. Each sequence in Argoverse 2 [61] is approximately 15 seconds long.

Since the original data contains regions where the ego-vehicle is visible in some of the cameras (cf. Fig. 7 of the main paper), we annotate each camera view with an ego-vehicle mask which we use in all experiments for all methods to constrain the ray sampling process. We release the full data splits and the sequence alignment transformations with our source code.

C. Method Details

In this section, we provide more details on our method.

Appearance and transient geometry embeddings. To condition our sequence-level appearance and transient geometry matrices $\mathbf{A}_s$ and $\mathbf{G}_s$ on the time t, we use the 1D basis function $\mathcal{F}(t)$ as mentioned in Sec. 4 of the main paper. We use six as the number of frequencies of $\mathcal{F}(t)$ for both appearance and transient geometry embeddings. The resulting vectors ω_s^t and ω_o are in $\mathbb{R}^{64}$. For ω_s^t, we learn $\mathbf{A}$ and $\mathbf{G}$ per sequence both in $\mathbb{R}^{32\times 6\cdot 2+1}$, *i.e.* desired latent vector size by output dimension of $\mathcal{F}(t)$. For ω_o, we learn separate geometry and appearance codes per object both in $\mathbb{R}^{32}$.

Transient density $\sigma_\mathbf{G}$ and color $\mathbf{c}_\mathbf{G}$. In Eq. 8 of the paper, we define the output of the transient geometry branch which is used to calculate the final static color. We blend the transient color $\mathbf{c}_\mathbf{G}$ with the predicted color $\mathbf{c}_\phi$ in Eq. 7

weighted by the densities σ_ϕ and $\sigma_{\mathbf{G}}$ analogous to Eq. 12:

$$\sigma_\phi = \sigma_\phi + \sigma_{\mathbf{G}}\,,\; \mathbf{c}_\phi = \frac{\sigma_\phi}{\sigma_\phi + \sigma_{\mathbf{G}}}\mathbf{c}_\phi + \frac{\sigma_{\mathbf{G}}}{\sigma_\phi + \sigma_{\mathbf{G}}}\mathbf{c}_{\mathbf{G}}. \tag{S1}$$

Proposal network σ_{prop}. We align with [54] and use two separate proposal networks, one for each proposal sampling iteration. These proposal networks and our final static radiance field ϕ have increasing hash table sizes, acting as a coarse-to-fine representation of the scene geometry. In contrast to previous works [2, 54], we condition the proposal networks on the sequence-specific geometry codes to account for varying transient geometry across sequences in the proposal sampling stage.

Dynamic object radiance field ψ. For our dynamic object radiance field ψ, we use separate shape and appearance latent vectors that condition the radiance field. In particular, we use a shape code at the network input that we concatenate with the input coordinate $\mathbf{x}$ and further an appearance code that we concatenate with the direction $\mathbf{d}$ at the bottleneck after density prediction. We concatenate sequence and object appearance latent vectors to propagate sequence appearance to the individual objects.

Space contraction. As mentioned in Sec. 4 of the main paper, we follow [2, 54] and contract the unbounded scene space into a unit cube. In particular, we use the following function for space contraction:

$$\chi(\mathbf{x}) = \begin{cases} \mathbf{x}, & ||\mathbf{x}||_\infty \leq 1 \\ (2 - \frac{1}{||\mathbf{x}||_\infty})\frac{\mathbf{x}}{||\mathbf{x}||_\infty}, & ||\mathbf{x}||_\infty > 1 \end{cases} .$$

Limitations. While our method sets a new state-of-the-art for radiance field reconstruction in dynamic urban environments under varying environmental conditions, the extremely challenging nature of the problem persists and further research in this area is needed. For example, we can much better represent highly dynamic, rigid objects such as cars, vans, trucks, and buses. Still, objects with highly intricate motions such as pedestrians or cyclists continue to be a challenge. Another limitation stems from the inherent problem of insufficient view coverage. For areas that were not clearly visible to the ego-car, we find that the rendering quality is significantly lower. This is particularly pronounced for dynamic objects since they are only present in a single sequence. However, we note that this problem is attenuated by the initialization of radiance field ψ with a semantic prior. Overall, views farther away from the training trajectories would constitute an interesting addition to the evaluation setup. Yet, utilizing (partial) hold-out sequences is not suitable for our task as these would contain distinct transient geometry and dynamic objects, and possibly appearance unknown to the model. Thus, a different capturing setup would be required which is outside the scope of our work but is an interesting area for future research.

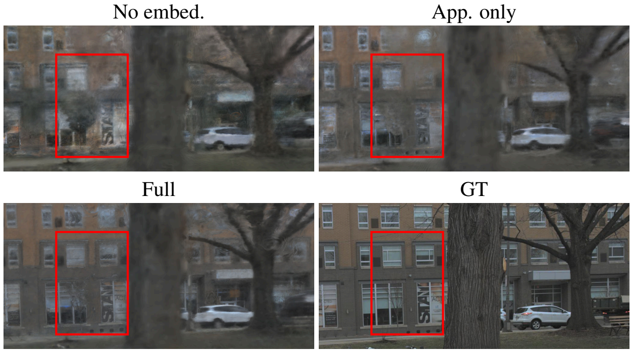

Figure 9. **Qualitative comparison of graph structure.** We show a qualitative illustration of our ablation study in Tab. 4. In particular, we show the results of our method without any sequence-dependent latent vectors ω_s^t, with only the appearance vectors $\mathbf{A}_s\mathcal{F}(t)$ and of our full method.

Split	3D Box Type	PSNR ↑	SSIM ↑	LPIPS ↓
Single Seq.	GT	27.07	0.759	0.362
	Prediction	26.71	0.756	0.365
Residential	GT	22.29	0.678	0.523
	Prediction	21.28	0.667	0.538

Table 7. **Ablation on 3D bounding boxes.** We show results on a single sequence of the residential area in our benchmark and the full residential area. We train our method with either the provided 3D bounding box annotations or predictions acquired from an off-the-shelf 3D tracker.

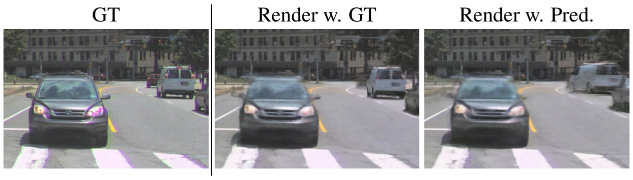

Figure 10. **Failure case of predicted 3D bounding boxes.** While the car in the foreground is well reconstructed with both ground truth and predicted 3D bounding boxes, the van in the background is rendered with incorrect orientation and is slightly too big.

Finally, while our method improves over naive pose optimization, we note that this is a challenging problem and that large pose errors are hard to correct during reconstruction. We thus tackled this issue by pre-aligning the sequences with our offline ICP procedure. We hope that our proposed benchmark can spark further research that addresses these issues.

D. Additional Experiments

We discuss further details on our experimental setup, additional experiments, and qualitative results.

Implementation details. We compute the scene bounds from the LiDAR point cloud with a maximum distance of

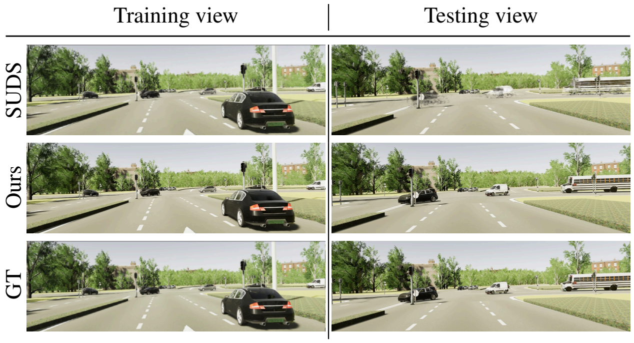

Figure 11. **Qualitative comparison on VKITTI2.** We observe that SUDS [73] can render realistic training views, but cannot properly recover the dynamic actors in the testing views. In contrast, our method shows no quality difference in rendering training and novel testing views.

80 meters per sweep, *i.e.* we filter each point cloud so that only points less than 80 meters from the ego-vehicle remain, and use the ego-vehicle poses to register all point clouds in the global world frame. With this global, world-frame point cloud we compute the scene bounds. We use the following loss weights for Eq. 13 of the main paper: $\lambda_{\text{dist}} = 0.002$, $\lambda_{\text{prop}} = 1.0$, $\lambda_{\text{dep}} = 0.05$, and $\lambda_{\text{entr}} = 0.0001$. For $\mathcal{L}_{\text{dep}}$, we use the LiDAR measurements as ground truth. We only take depth measurements at the time of the camera sensor recording to ensure dynamic objects receive valid depth supervision. Following previous works [21, 54], we optimize the evaluation camera poses during the validation phase to compensate for pose errors introduced by drifting geometry through optimized training poses that would otherwise contaminate the view synthesis quality measurement. For this, we use $\mathcal{L}_{\text{rgb}}$ only. For the depth map visualizations shown in our qualitative results, we use linear scaling with a maximum depth of 82.5 meters and a minimum depth of 1 meter. Rendering a 1920×1080 image takes $\sim$16.4 seconds. The training speed is 30K rays per second.

Baselines. We run SUDS [58] on our benchmark using their official code release. Since it requires several additional inputs such as LiDAR depth and optical flow predictions, we compute the optical flow of all sequences with RAFT [72] following the experimental setup of SUDS [58] on their City-1M benchmark. We align all other auxiliary inputs such as depth with our method. We deactivate the DINO feature distillation branch that is used in SUDS for semantic reconstruction.

Influence of graph structure on view quality. In Fig. 9, we show a qualitative comparison of our method without the node latent codes in our graph, *i.e.* $\mathbf{A}_s\mathcal{F}(t)$ and $\mathbf{G}_s\mathcal{F}(t)$, our method with only appearance codes $\mathbf{A}_s\mathcal{F}(t)$, and our full method.

We observe that without any conditioning on the sequence s, there are strong artifacts, *e.g.* the smaller tree highlighted in red is being rendered with leaves and the wall of the building behind it has a significantly different color than in the ground truth image. With the appearance embedding only, the color problem is alleviated, but the texture of the wall is highly distorted since there is no way for the model to distinguish between sequences where the tree leaves are present and the wall is occluded and where the wall is visible. In contrast, our full method recovers a faithful rendering of the tree, the wall and also the windows above.

3D bounding box predictions. While we follow previous works [35, 67] and use provided 3D bounding boxes in our experiments, we also report results using off-the-shelf algorithms to predict the 3D bounding boxes of dynamic objects. In particular, we use an off-the-shelf LiDAR-based 3D object tracker [68, 71] to generate 3D bounding box tracks. We use those tracks instead of the provided 3D bounding box annotations. Note that neither the 3D object detector nor the tracking algorithm is adjusted or fine-tuned for the Argoverse 2 [61] dataset. We take the officially provided models trained on the nuScenes dataset [70]. This dataset notably has different LiDAR sensor properties than Argoverse 2.

In Tab. 7, we compare the results of our method with annotated 3D bounding boxes and with the predicted bounding boxes. We train our method both on a single sequence, *i.e.* the same sequence as in Tab. 5 of the main paper, and the full residential area in our benchmark. When trained on a single sequence, we observe that the difference in view quality is marginal. Trained on the full residential split of our benchmark, the gap becomes slightly larger while the performance is still competitive. We analyze this more closely in Fig. 10, where we observe some failure cases when predicted boxes are inaccurate, *e.g.* when the orientation of an object is not correctly predicted and can also not be recovered through our pose optimization. Still, the synthesized views look realistic. Overall, this shows that our approach can be scaled to large vehicle fleet data without the need for manual data annotation simply through employing off-the-shelf LiDAR 3D tracking algorithms without much loss in realism.

Analysis of KITTI and VKITTI2 results. We observe a large gap between previous state-of-the-art methods and our method in terms of novel view synthesis quality in Tab. 2. At the same time, our image reconstruction quality is superior, but the gap is significantly smaller (cf. Tab. 3). Motivated by this observation, we retrain SUDS [58] on the VKITTI2 dataset and visualize its renderings for example training and testing views alongside the results of our method in Fig. 11. We observe that indeed the reconstruction quality of the training views is comparable for SUDS and our method, but SUDS fails to recover dynamic objects in the testing view properly, while our method pro-

$\mathcal{L}_{\text{rgb}}$	$\mathcal{L}_{\text{dep}}$	$\mathcal{L}_{\text{entr}}$	PSNR ↑	SSIM ↑	LPIPS ↓	AbsRel ↓
✓	-	-	22.22	0.675	0.524	0.321
✓	✓	-	22.23	0.677	**0.523**	0.219
✓	-	✓	22.20	0.676	**0.523**	0.333
✓	✓	✓	**22.29**	**0.678**	**0.523**	**0.218**

Table 8. **Loss term ablation.** We report both the view and depth quality of our model when ablating different loss terms.

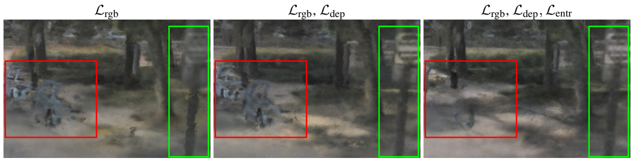
$\mathcal{L}_{\text{rgb}}$
$\mathcal{L}_{\text{rgb}}, \mathcal{L}_{\text{dep}}$
$\mathcal{L}_{\text{rgb}}, \mathcal{L}_{\text{dep}}, \mathcal{L}_{\text{entr}}$

Figure 12. **Free viewpoint renderings.** Without $\mathcal{L}_{\text{entr}}$, the separation between dynamic and static content is ambiguous (red). Without $\mathcal{L}_{\text{dep}}$, the traffic pole exhibits artifacts (green).

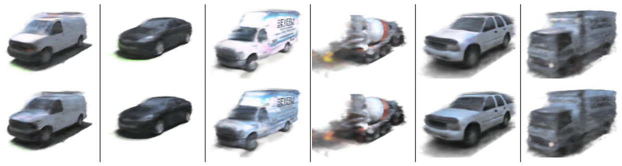

Figure 13. **Object renderings.** We illustrate object instances in sunny conditions (top) and cloudy conditions (bottom).

duces high-quality renderings also for novel views. This shows that our scene graph-based, high-level decomposition excels at representing scenes with highly dynamic objects while previous work struggles with this.

Ablation of $\mathcal{L}_{\text{dep}}$, $\mathcal{L}_{\text{entr}}$. In Tab. 8, we observe that while depth and entropy losses have a limited effect on evaluation view quality, the depth loss helps in improving geometry accuracy (AbsRel). Intuitively, this improves view quality farther from the training trajectory. The entropy loss encourages static and dynamic radiance separation, improving object renderings and scene decomposition. We illustrate these effects in Fig. 12.

Additional qualitative results of object-centric renderings. In Fig. 13, we show additional object-centric renderings conditioned on different scene appearances. In particular, we depict objects with more intricate textures, showing the ability of ψ to generalize to a wide variety of object instances. Note that the instance reconstruction quality varies with the observed training views (cf. Fig. 13 right). Yet, we note that our method models objects much better than existing works.

Additional qualitative comparison. We include further qualitative results of our method compared to the state-of-the-art in Fig. 14. We observe that, similar to Fig, 7 of the main paper, our method exhibits superior view synthesis of dynamic areas and better captures seasonal variations in terms of, for example, tree leaves. Further, the synthesized views of our method are sharper compared to prior art, and the depth maps are less noisy. We include qualitative results from both the residential and downtown areas of our benchmark.

References

[70] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 3

[71] Ziqi Pang, Zhichao Li, and Naiyan Wang. Simpletrack: Understanding and rethinking 3d multi-object tracking. *arXiv preprint arXiv:2111.09621*, 2021. 3

[72] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 3

[73] Haithem Turki, Jason Y Zhang, Francesco Ferroni, and Deva Ramanan. Suds: Scalable urban dynamic scenes. In *CVPR*, 2023. 3

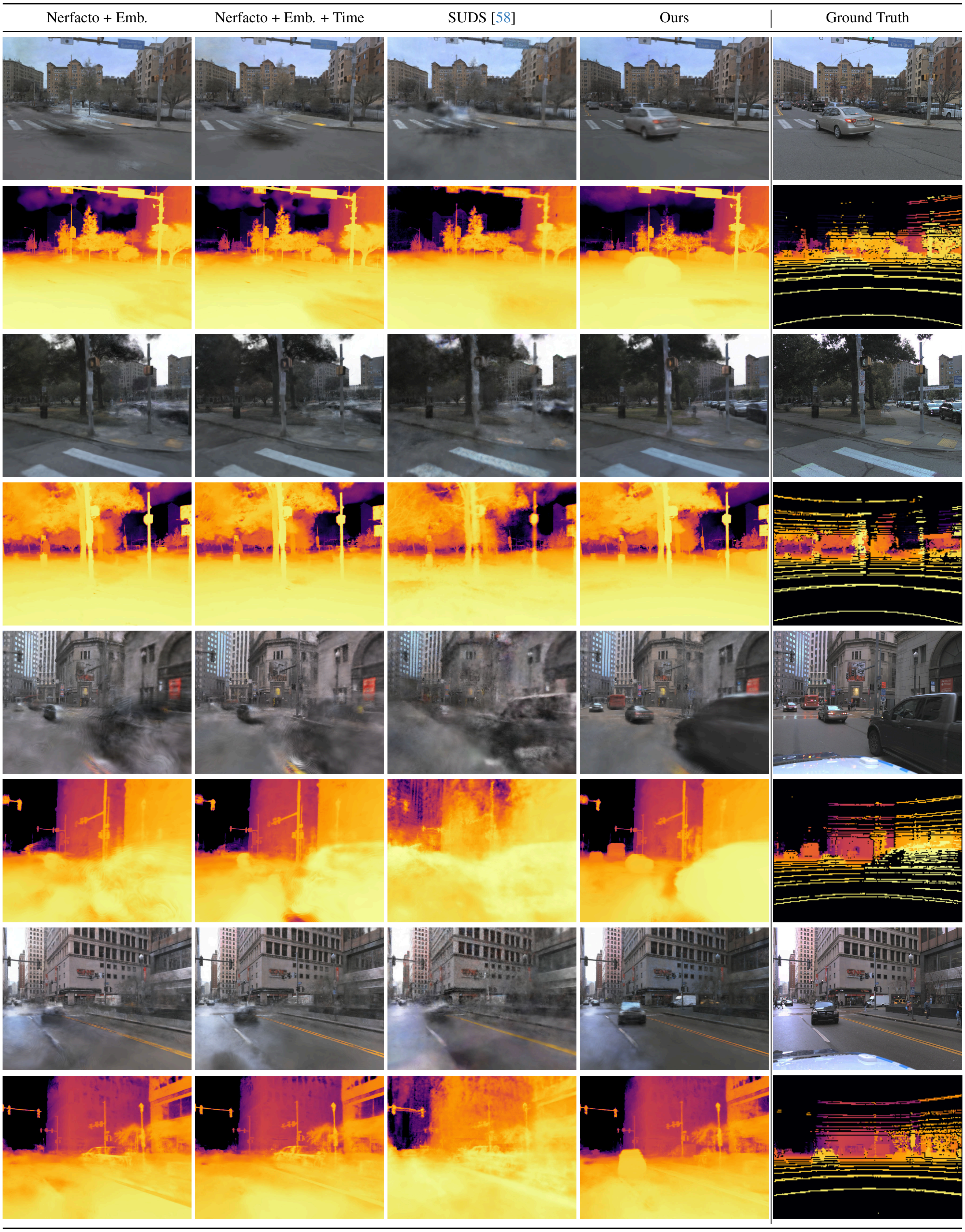

Figure 14. **Additional qualitative results on Argoverse 2.** We illustrate four examples, where the upper two are from the residential area and the lower two are from the downtown area in our benchmark. We observe that our method exhibits better view quality and cleaner depth maps, particularly in dynamic areas.