GEM: Generating LiDAR World Model via Deformable Mamba

Supplementary Material

A. Appendix

In this Appendix, we provide additional information. In Sec. A.1, we introduce the process of projecting LiDAR point clouds onto range maps; In Sec. A.2, we provide a more detailed explanation of how we utilize the Mamba algorithm and the diffusion algorithm; In Sec. A.3, we introduce the detailed architectures of discriminator; In Sec. A.4, we provide additional visualizations of future prediction results generated by our GEM; In Sec. A.5, we present reconstruction results from our LiDAR scene tokenizer, including both 32-line and 64-line LiDAR data; In Sec. A.6, we show more results on controllable generation, including counterfactual reasoning and BEV-layouts controlled generation; In Sec. A.7, we analyze GEM's capability for long-term future prediction and present visualizations of the results; In Sec. A.8, we discuss the limitations and broader impacts of GEM and provide possible solutions.

Additionally, we also upload videos in the supplementary materials demonstrating the quality of reconstructed data and generated sequences.

A.1. LiDAR Data Representation

LiDAR point clouds captured in outdoor environments typically exhibit high sparsity and irregularity. To address these challenges, the range map has emerged as an effective representation that successfully mitigates the issues of sparsity and irregularity in LiDAR data, as demonstrated in prior studies [2, 7, 8]. While efficient feature extraction for range maps remains under-explored, their adoption in LiDAR generation research has become mainstream [9, 10, 13, 16].

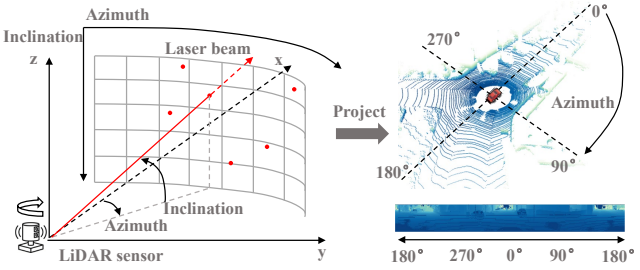

Figure 1. The conversion process from the LiDAR point cloud to the range map.

Compared to voxel and BEV representations that suffer from information loss caused by limited resolution and computational cost [15], a key advantage of the range map representation is its ability to preserve nearly all original information from the raw point cloud data while maintaining low computational cost [8]. As shown in Figure 1, we transform the coordinate space of each point, projecting it from Cartesian coordinates $\mathbf{r} = (x, y, z) \in \mathbb{R}^3$ to spherical coordinates $\mathbf{s} \in (\theta, \phi, d)$:

$$\begin{aligned} d &= \sqrt{x^2 + y^2 + z^2}, \\ \theta &= \texttt{arccos}(z/\sqrt{x^2 + y^2 + z^2}), \\ \phi &= \texttt{atan2}(y, x), \end{aligned} \tag{1}$$

where θ is the inclination, ϕ is the azimuth, d is the depth. For a 64-line LiDAR, we set the dimensions of range map to 64×1024, and for a 32-line LiDAR, we set the dimensions of range map to 32×1024. A row in the range map corresponds to a beam circle obtained from a full rotation scan by the LiDAR sensor, and a column in the range map corresponds to the central ray acquired from the emission of the LiDAR sensor [14].

A.2. More Detailed Algorithmic Procedures

In this section, we first elaborate on the specific implementation of Mamba in our framework, followed by a detailed explanation of how the diffusion algorithm operates within our GEM.

Each row of a range map corresponds to one full rotation of LiDAR scanning points. Partitioning range maps into patches lacks clear physical significance, whereas each row naturally preserves the structural information of LiDAR data. Therefore, unlike previous methods that apply Mamba by scanning image patches, we perform scanning point-by-point along the LiDAR beams, which better aligns with the inherent characteristics of point clouds.

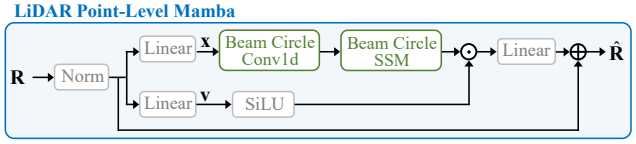

Figure 2. The architecture of the Mamba we employ. Unlike patch-level Mamba used in image processing, our Mamba performs scanning circle-by-circle directly on LiDAR points, effectively preserving the structural information of LiDAR data.

Specifically, taking the range map $\mathcal{R}$ as input, to align with the imaging characteristics of LiDAR data, we use convolutional layers for preliminary feature modeling, then flatten it pixel by pixel along rows to obtain $\mathbf{R} \in \mathbb{R}^{K \times J}$, where K and J represent the number of points and channels. As shown in Fig. 2, within the Mamba process, the input $\mathbf{R}$ is first projected to embeddings $\mathbf{x}$ and $\mathbf{v}$ via linear

layers:

$$\mathbf{x} = \texttt{Linear}_{\mathbf{x}}(\mathbf{R}), \mathbf{v} = \texttt{Linear}_{\mathbf{v}}(\mathbf{R}). \tag{2}$$

Then, to enable input-independent reasoning, we introduce selective parameters $\mathbf{B}$, $\mathbf{C}$ and a discretization parameter $\mathbf{\Delta} \in \mathbb{R}^{K\times D}$:

$$\begin{aligned} \mathbf{B} &= \texttt{Linear}_{\mathbf{B}}(\mathbf{x}), \mathbf{C} = \texttt{Linear}_{\mathbf{C}}(\mathbf{x}) \\ \mathbf{\Delta} &= \texttt{log}(1 + \texttt{exp}(\texttt{Linear}_{\mathbf{\Delta}}(\mathbf{x}) + \texttt{bias}_{\mathbf{\Delta}})), \end{aligned} \tag{3}$$

where $\texttt{bias}_{\mathbf{\Delta}}$ is a learnable bias parameter of $\mathbf{\Delta}$. Next, to apply the concept of continuous dynamic systems to discrete inputs, we use the zero-order hold to obtain discretization $\overline{\mathbf{A}}, \overline{\mathbf{B}} \in \mathbb{R}^{K\times D}$:

$$\overline{\mathbf{A}} = \texttt{exp}(\mathbf{A}\cdot\mathbf{\Delta}), \overline{\mathbf{B}} = (\mathbf{\Delta}\cdot\mathbf{A})^{(-1)}\cdot(\texttt{exp}(\mathbf{\Delta}\cdot\mathbf{A})-\mathbf{I}), \tag{4}$$

where $\mathbf{A} \in \mathbb{R}^{K\times D}$ is the learnable parameters. Subsequently, we perform the Mamba scan for each step $k \in [1, \cdots, K]$ until $\hat{\mathbf{R}} \in \mathbb{R}^{K\times D}$ is obtained:

$$\begin{aligned} &h_0 = 0, h_k = \overline{\mathbf{A}}_k \cdot h_{k-1} + \overline{\mathbf{B}}_k \cdot \mathbf{x}_k, \hat{\mathbf{R}}_k = \mathbf{C}_k \cdot h_k, \\ &\hat{\mathbf{R}} = \texttt{MLP}([\hat{\mathbf{R}}_1, \cdots, \hat{\mathbf{R}}_L]^\top) \cdot \texttt{SiLU}(\mathbf{v}) + \texttt{Norm}(\mathbf{R}), \end{aligned} \tag{5}$$

where MLP consists of linear and normalization layers. Finally, after several channel modification operations, we can obtain the compressed latent $\mathbf{z} \in \mathbb{R}^{h\times w\times C}$, where h, w, C means the number of height, width and channels.

Give a latent feature $\mathcal{Z} \in \mathbb{R}^{\tau\times h\times w\times C}$ with τ frames of $\mathbf{z}$. We construct GEM following the latent diffusion paradigm [11, 12], which mainly contains two parts. (1) Forward process: given LiDAR series $\mathcal{Z} = [\mathcal{Z}^p, \mathcal{Z}^f]$, this process transforms the future frames $\mathcal{Z}^f$ into pure Gaussian noise in an iterative manner. The addition of noise at timestep t is defined as: $\mathcal{Z}^f_t = \sqrt{\alpha_t}\mathcal{Z}^f_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t$, where $\alpha_t = 1 - \beta_t$, β_t is the noise weight, and the $\boldsymbol{\epsilon}_t$ is the added Gaussian noise. Through iterative derivation [5], we can obtain $\mathcal{Z}^f_t = \sqrt{\bar{\alpha}_t}\mathcal{Z}^f_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t$, where $\bar{\alpha}_t = \alpha_t \cdot \alpha_{t-1} ... \alpha_1$, and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This process can be further expressed as: $q(\mathcal{Z}^f_t|\mathcal{Z}^f_0) = \mathcal{N}(\mathcal{Z}^f_t; \sqrt{\bar{\alpha}_t}\mathcal{Z}^0_f, (1-\bar{\alpha}_t)\mathbf{I})$. (2) Reverse process: we need to recover the data from the noise distribution.

$$q(\mathcal{Z}^f_{t-1}|\mathcal{Z}^f_t, \mathcal{Z}^f_0) = \mathcal{N}(\mathcal{Z}^f_{t-1}; \widetilde{\mu}_t(\mathcal{Z}^f_t, \mathcal{Z}^f_0), \widetilde{\beta}_t\mathbf{I}), \tag{6}$$

in which, $\widetilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t, \widetilde{\mu}_t(\mathcal{Z}^f_t, \mathcal{Z}^f_0) = \frac{1}{\sqrt{\alpha_t}}(\mathcal{Z}^f_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_t)$, and $\hat{\boldsymbol{\epsilon}}_t$ is predicted by our LiDAR world model.

The training objective of timestamp t is to estimate the noise $\hat{\boldsymbol{\epsilon}}_t$ in Eq. (6) based on $\mathcal{Z}_t = [\mathcal{Z}^p, \mathcal{Z}^f_t]$. After training, given an initial noise latent $\hat{\mathcal{Z}}^f_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ we can progressively recover it over T steps to produce the final prediction, where T is the predefined denoising step.

A.3. Detailed Architectures of Discriminator

During the training of the LiDAR scene tokenizer, the MSE loss function tends to generate blurred and averaged results, as it minimizes pixel-level errors while overlooking the plausibility of local structures [4]. Relying solely on MSE loss makes it challenging to reconstruct high-frequency details in range maps, which are critical for features such as object contours.

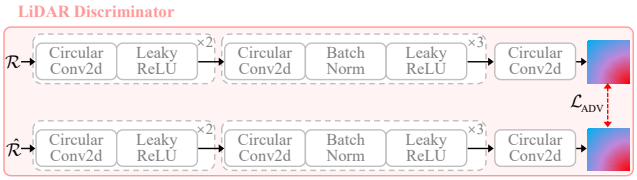

Figure 3. The architecture of the discriminator we employ. The application of a discriminator assists the LiDAR scene tokenizer in reconstructing high-frequency details, which is crucial for range maps since these details correspond to object contours, road structures, and other critical information.

To address this, we introduce a discriminator to improve the model's reconstruction accuracy. As illustrated in Fig. 3, the discriminator comprises circular convolutions, normalization layers, and activation functions. Since the reconstructed $\hat{\mathcal{R}}$ has already undergone Mamba processing, which effectively preserves LiDAR structural information during feature extraction and reconstruction, the discriminator focuses on assessing the perceptual realism of $\hat{\mathcal{R}}$, thus eliminating the need for Mamba operations that are primarily geared toward precision reconstruction. The discriminator emphasizes perceptual "realism" rather than pixel-wise alignment, thereby compelling the tokenizer to learn the true distribution on the data manifold instead of merely fitting the average behavior of the training samples.

A.4. More Visualizations of World Modeling

In this section, we present more visualizations for 3s predictions and compare them with 4D-Occ [6]. On nuScenes [1] dataset, the 3s future comprises 6 frames, while on KITTI Odometry [3], it consists of 5 frames.

On nuScenes. Fig. 4 supplements the main-text visualizations by presenting the complete sequence from 0.5s to 3.0s at 0.5s intervals alongside the historical 6 frames observations, which have omitted the 0.5s, 1.5s, and 2.5s future frames due to space constraints before. Additionally, Figure 5 provides further visualizations on the nuScenes dataset. It can be observed that our method achieves more accurate predictions of objects, with clearer contours. In contrast, competing methods exhibit blurred or even missing object boundaries. Additionally, our method provides more precise predictions of LiDAR beams, while competing methods show significant noise. These visualizations further demonstrate our superiority on nuScenes.

On KITTI Odometry. Fig. 6 supplements the main-text visualizations by presenting the complete sequence from 0.6s to 3.0s at 0.6s intervals alongside the historical 5 frames observations, which have omitted the 1.2s, and 2.4s future frames due to space constraints before. Additionally, Figure 7 provides further visualizations on the KITTI Odometry dataset. These visualizations demonstrate our superiority on KITTI Odometry. It is evident that our method produces clearer object predictions with fewer noisy points. As the distance increases, our approach maintains superior LiDAR beam accuracy, whereas competing methods exhibit progressively more blurring.

A.5. Visualizations of Reconstruction

This section demonstrates our LiDAR scene tokenizer's reconstruction capability, achieving compression ratios of 2.67 (32-line) and 5.32 (64-line) while maintaining high-fidelity reconstruction. As shown in Figs. 8 and 9, our LiDAR scene tokenizer can accurately reconstruct scene structures and object details from LiDAR data, maintaining high reconstruction accuracy even in challenging scenarios such as rainy conditions. Specifically, in rows 1 to 6 of Figure 8, we demonstrate diverse road conditions, including straight roads, turns, and intersections. It can be seen that our method accurately reconstructs small objects such as cars and pedestrians, as well as road structures. In rows 7 to 10 of Fig. 8, we present the reconstruction results of our LiDAR scene tokenizer for rainy scenes. It can be observed that, owing to our precise capture of LiDAR data structural information, our method maintains high reconstruction accuracy even in sparser and noisier scenarios. Fig. 9 demonstrates the reconstruction results of 64-line LiDAR data under various road conditions. It can be observed that our method achieves exceptionally high fidelity in reconstructing both object details and road structures.

A.6. More Visualizations of Controlled Generation

This section presents additional visualizations to demonstrate our model's capabilities in controlled generation, covering both counterfactual reasoning and BEV-guided generation. As shown in Figs. 10 and 11, we present the current and past frames alongside a comparison between original predictions and counterfactual reasoning. Fig. 10 illustrates the transition from a left turn to emergency braking, while Fig. 11 demonstrates the shift from stationary waiting to a right turn. The results indicate that our method thoroughly comprehends the driving world, possesses strong imaginative capabilities, and can generate plausible scenarios based on given counterfactual actions. Fig. 12 demonstrates the editing of future frames under BEV layout control, including object addition, deletion, and repositioning. It can be observed that GEM accurately accomplishes these editing tasks.

A.7. Visualizations of Long-Sequence Generation

Generating long-term sequences is a ultimate goal for world models. In this section, we present visualizations that suggests the promising potential of GEM for long-horizon prediction. In detail, GEM first generates a future sequence based on observed past frames and subsequently uses this output as input, thereby iteratively expanding the prediction horizon. As shown in Fig. 13, GEM remarkably maintains considerable accuracy for up to 38 seconds before a noticeable performance decline at 46.8s. Notably, under the extreme challenge of the 58 seconds horizon, the model preserves only basic structural information. It is worth emphasizing that GEM is the only existing LiDAR world model capable of supporting minute-level future observations prediction with meaningful accuracy.

A.8. Limitations and Broader Impacts

Despite GEM achieving excellent results in LiDAR world modeling, it still has some shortcomings. Firstly, controlled generation including BEV-guided generation and counterfactual reasoning is still not mature enough for real world application. We attribute this limitation primarily to insufficient training data, and believe that scaling the dataset could potentially enhance the capacity. Secondly, despite demonstrating an ability for long-term generation which is not observed in previous methods, we observe a growing discrepancy between predictions and ground truth as time grows. We attribute this to compounding errors accumulated during iterative generation. Addressing the impact of these compounding errors on long-term performance will be a key focus of our future research. Thirdly, GEM currently cannot predict sudden environmental changes, such as abrupt rain or entering foggy areas. Collecting more diverse long-sequence data and data that includes weather variations would help alleviate the aforementioned issues.

In terms of broader impacts, GEM may potentially be utilized by technical personnel in economically developed regions for customized urban data simulation, indirectly replacing some labor-intensive data collection jobs. This may inadvertently widen the wealth gap, disrupt regional development balance, and lead to certain social inequities. As a potential countermeasure, broader social impact assessments should be conducted, and relevant authorities ought to establish comprehensive technical application frameworks to better harness technology for societal benefit.

CVPR
#9273

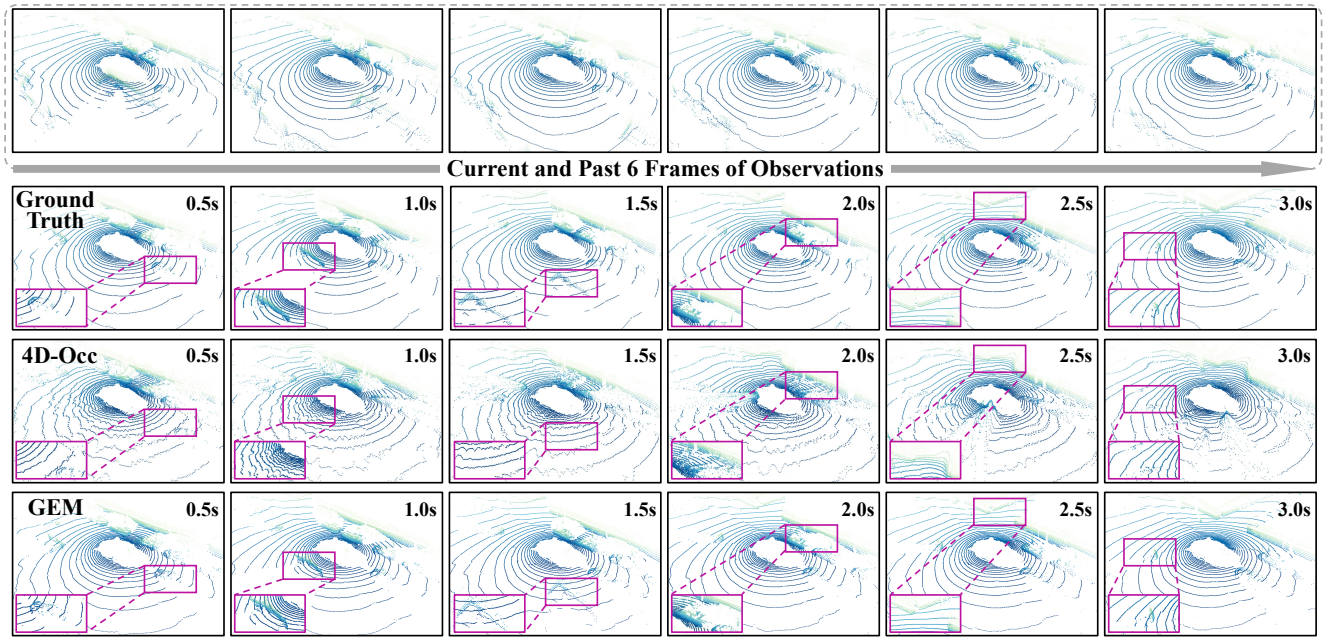

Figure 4. More visual comparisons with competitive LiDAR world models on nuScenes [1].

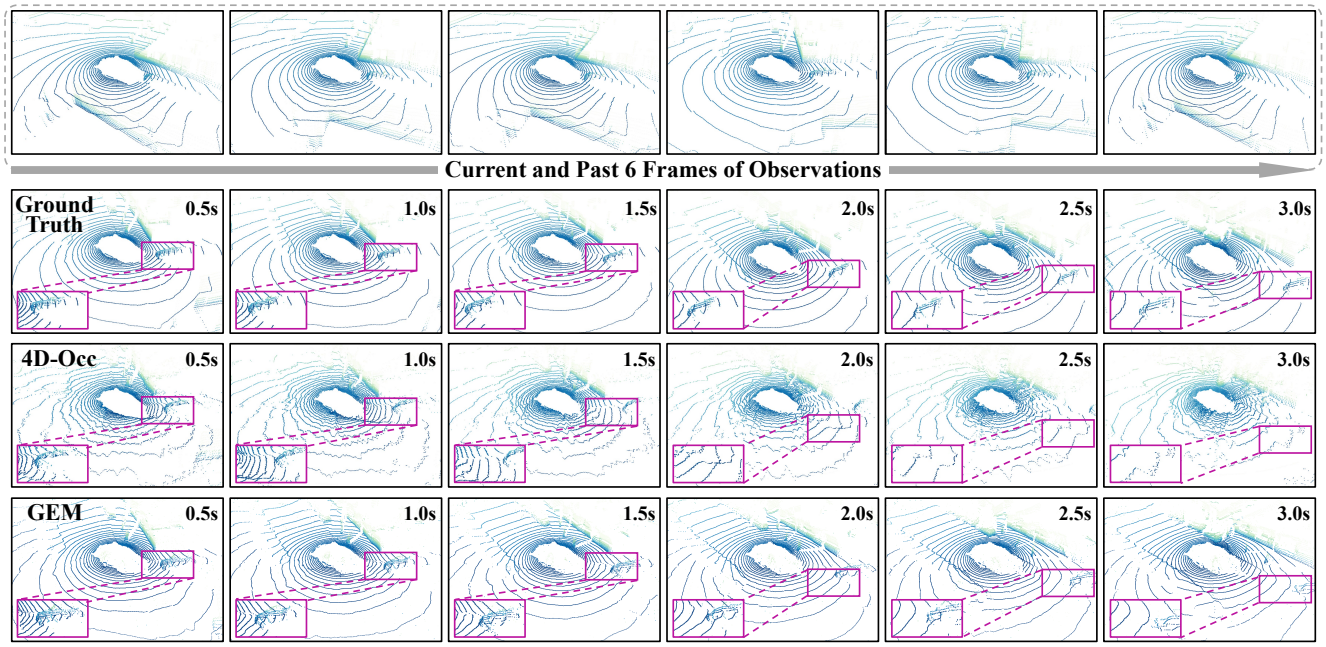

Figure 5. More visual comparisons with competitive LiDAR world models on nuScenes [1].

CVPR
#9273

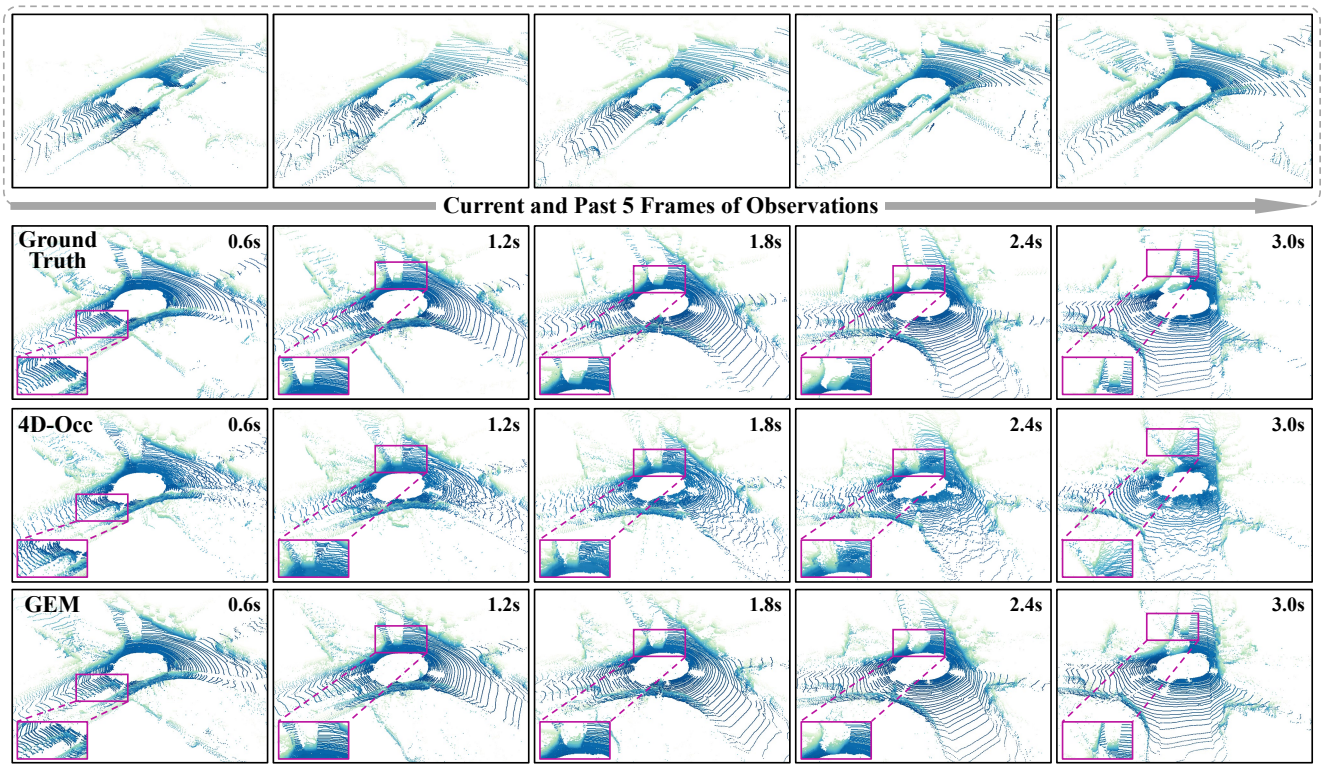

Figure 6. More visual comparisons with competitive LiDAR world models on KITTI Odometry [3].

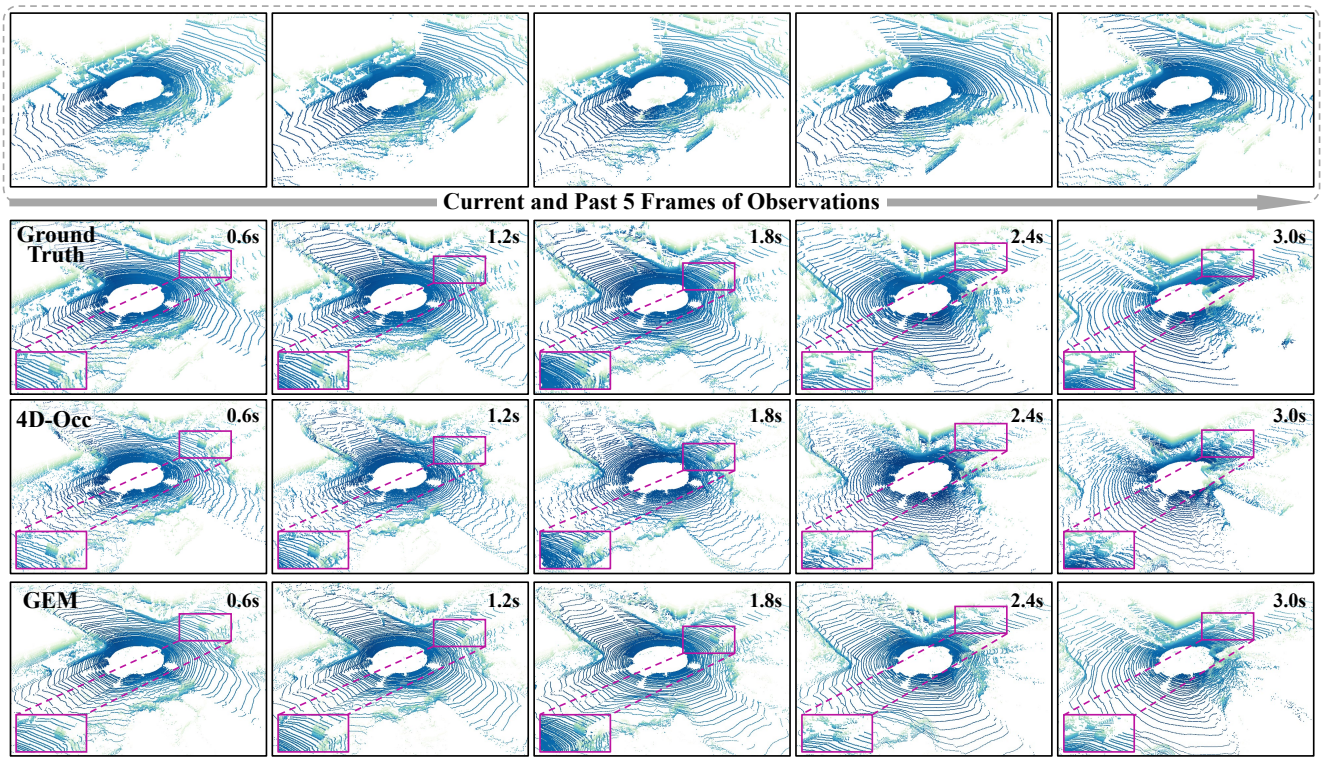

Figure 7. More visual comparisons with competitive LiDAR world models on KITTI Odometry [3].

Figure 8. Visualization of 32-line LiDAR data reconstruction by the LiDAR scene tokenizer..

CVPR
#9273

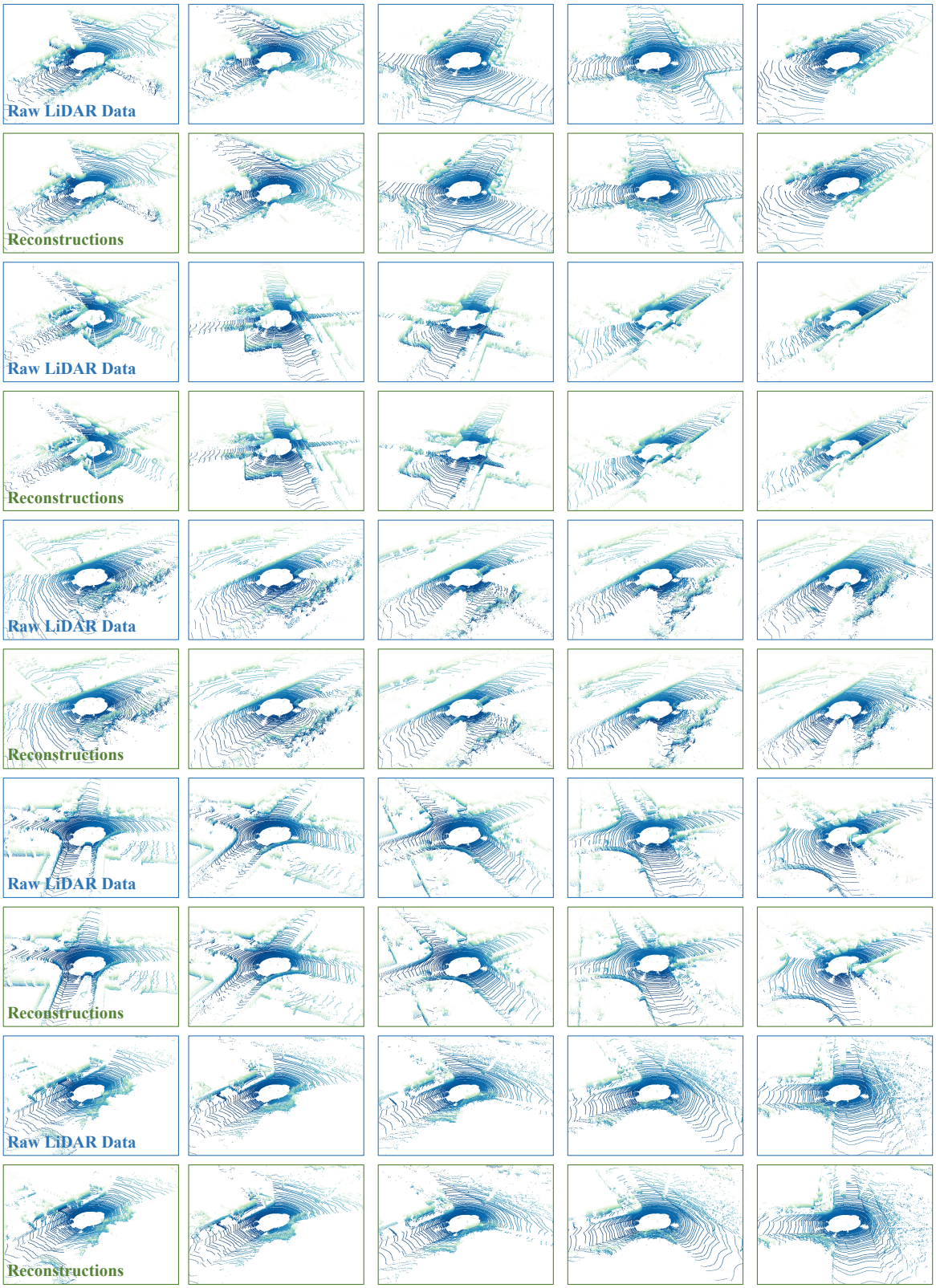

Figure 9. Visualization of 64-line LiDAR data reconstruction by the LiDAR scene tokenizer.

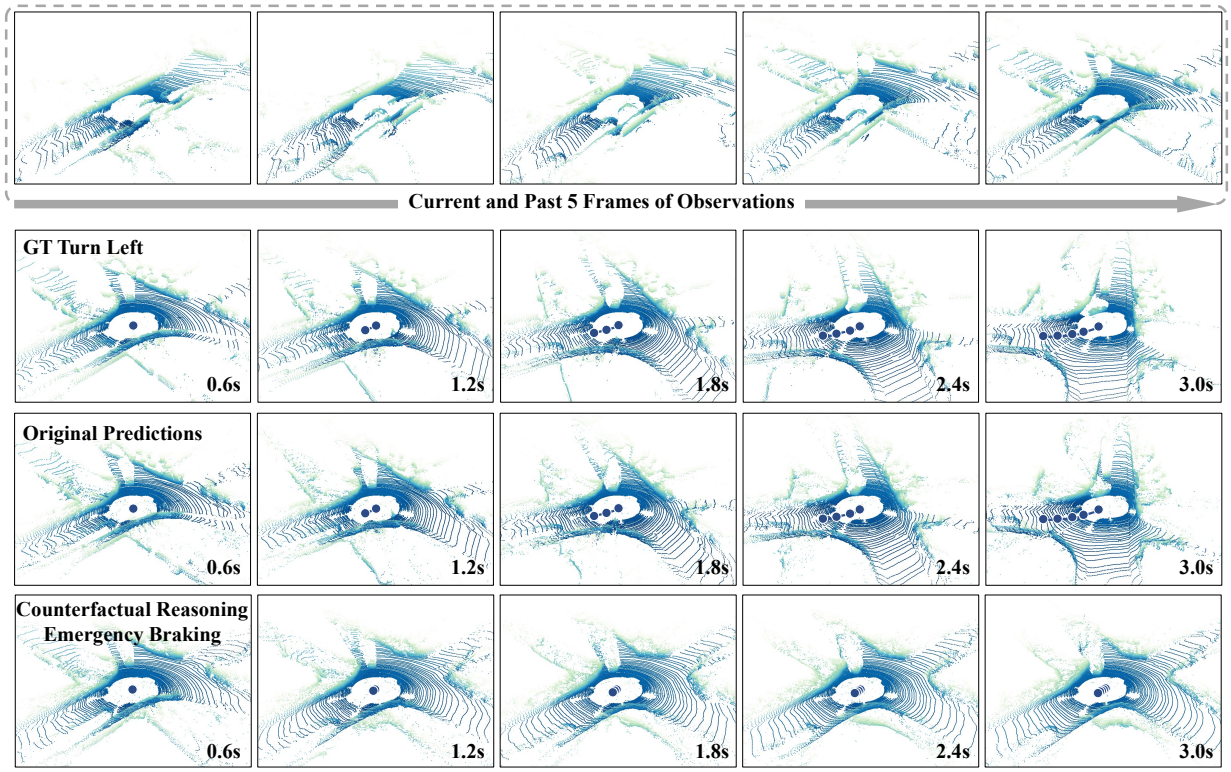

Figure 10. More visualizations of counterfactual reasoning.

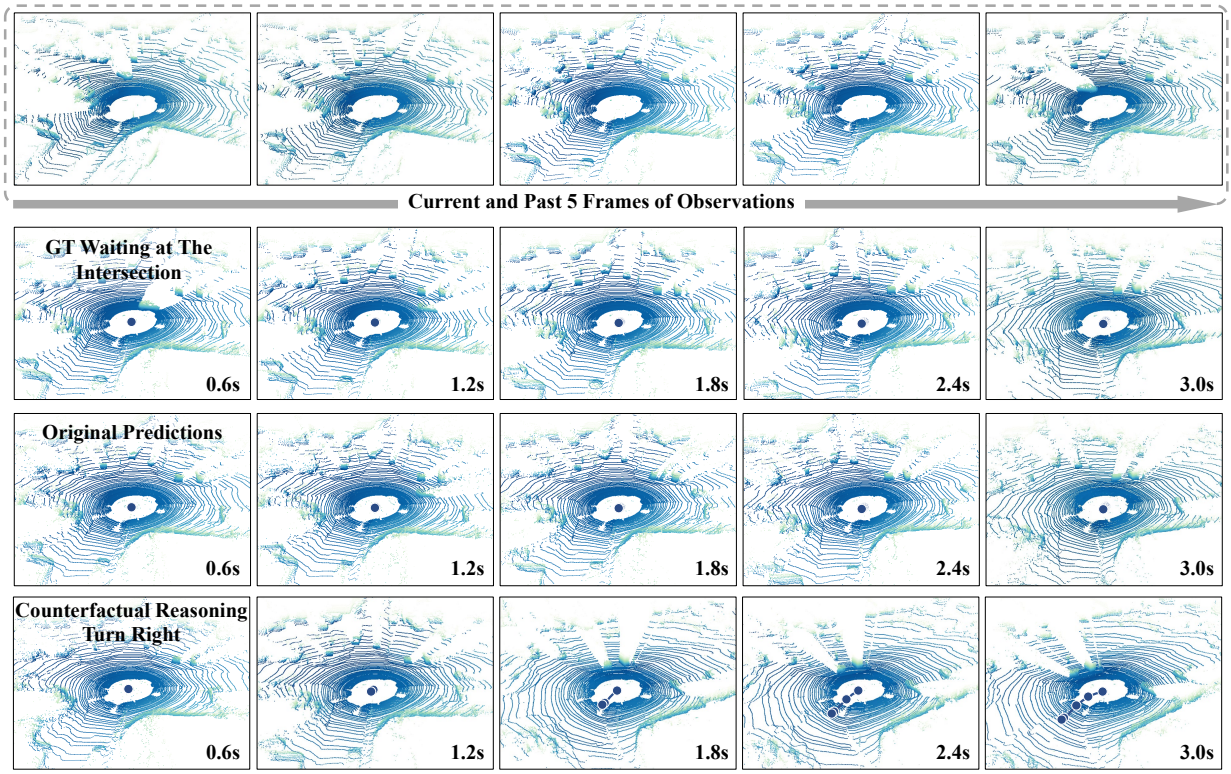

Figure 11. More visualizations of counterfactual reasoning.

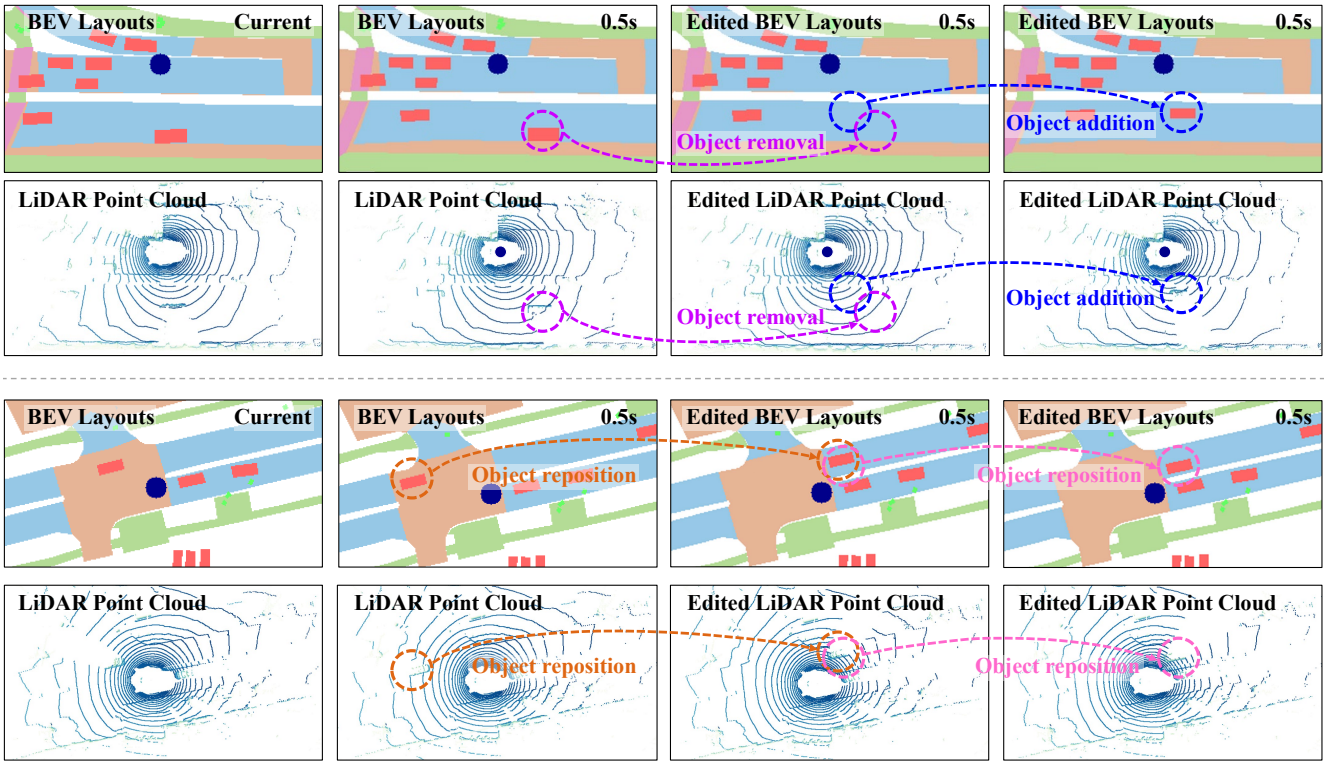

Figure 12. More visualization of bev layouts-controlled generation.

CVPR
#9273

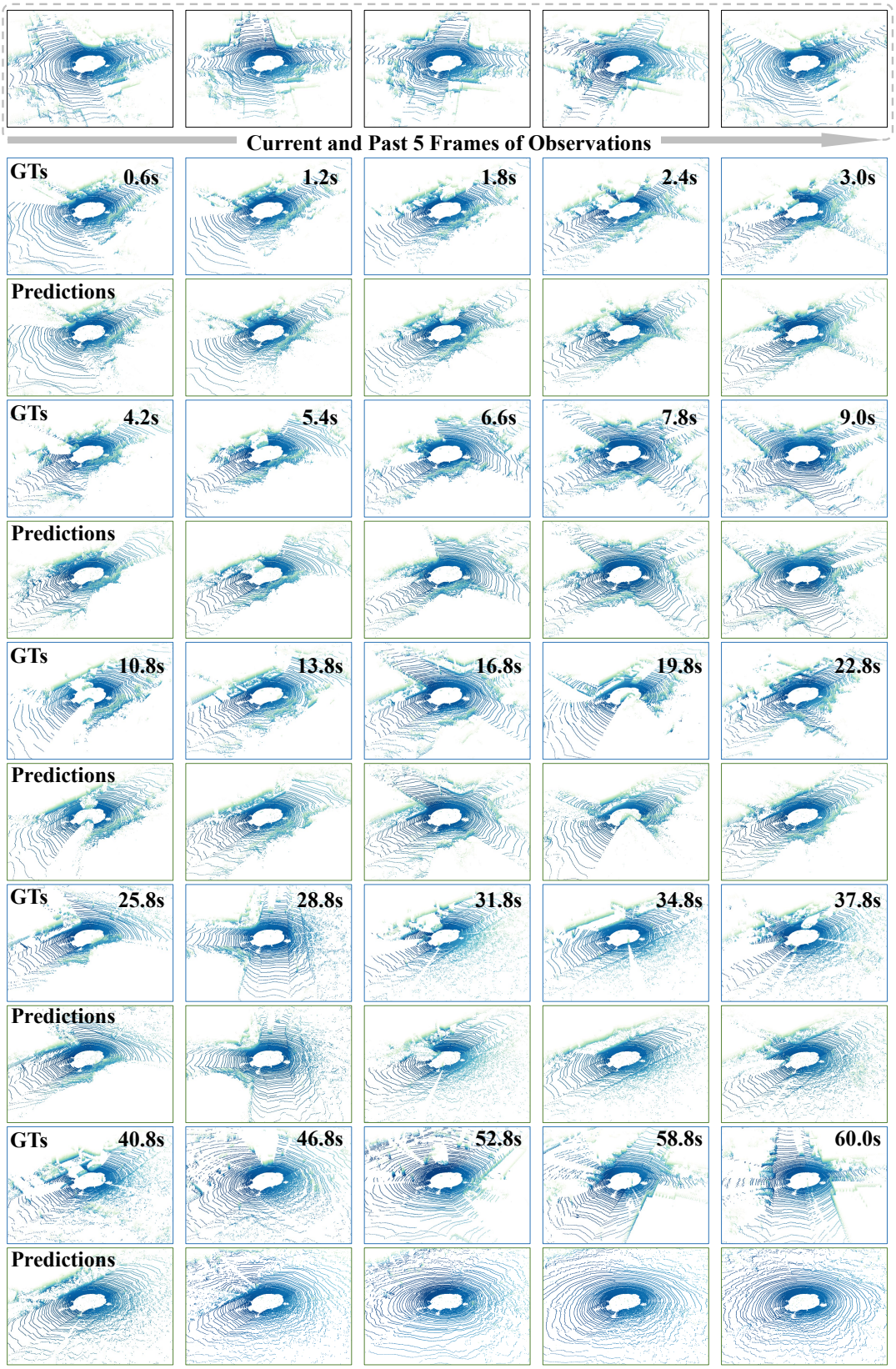

Figure 13. Visualizations of GEM's potential for minute-level future observations prediction.

References

[1] 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*, pages 11621–11631, 2020. 2, 4

[2] Yuning Chai, Pei Sun, Jiquan Ngiam, Weiyue Wang, Benjamin Caine, Vijay Vasudevan, Xiao Zhang, and Dragomir Anguelov. To the point: Efficient 3d object detection in the range image with graph convolution kernels. In *CVPR*, pages 16000–16009, 2021. 1

[3] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012. 2, 5

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 27, 2014. 2

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020. 2

[6] Tarasha Khurana, Peiyun Hu, David Held, and Deva Ramanan. Point cloud forecasting as a proxy for 4d occupancy forecasting. In *CVPR*, pages 1116–1124, 2023. 2

[7] Gregory P Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *CVPR*, pages 12677–12686, 2019. 1

[8] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *IROS*, pages 4213–4220. IEEE, 2019. 1

[9] Kazuto Nakashima and Ryo Kurazume. Learning to drop points for lidar scan synthesis. In *IROS*, pages 222–229. IEEE, 2021. 1

[10] Kazuto Nakashima and Ryo Kurazume. Lidar data synthesis with denoising diffusion probabilistic models. *arXiv preprint arXiv:2309.09256*, 2023. 1

[11] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023. 2

[12] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 2

[13] Yang Wu, Kaihua Zhang, Jianjun Qian, Jin Xie, and Jian Yang. Text2lidar: Text-guided lidar point cloud generation via equirectangular transformer. *arXiv preprint arXiv:2407.19628*, 2024. 1

[14] Yang Wu, Yun Zhu, Kaihua Zhang, Jianjun Qian, Jin Xie, and Jian Yang. Weathergen: A unified diverse weather generator for lidar point clouds via spider mamba diffusion. In *CVPR*, pages 17019–17028, 2025. 1

[15] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018. 1

[16] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to generate realistic lidar point clouds. In *ECCV*, pages 17–35. Springer, 2022. 1