Scaling Diffusion Models to Real-World 3D LiDAR Scene Completion

Supplementary Material

This supplementary material provides further detailed information on our proposed approach. We provide detailed information on the used network architectures, and ablations over the different hyperparameters of the generation process, *i.e.*, conditioning weight s and regularization weight r. Appendix A provides detailed information about the noise predictor and refinement network architectures and more detailed information about the refinement network training. Appendix B gives further ablations over the noise predictor regularization weight r. Appendix C presents qualitative and quantitative comparisons between the scene completion with different conditioning weights s and the unconditional generation, *i.e.*, $s = 0.0$. Appendix D compares qualitatively the scene completion with different number of denoising steps. Finally, Appendix E shows further qualitative results comparing our scene completion with the evaluated baselines. Furthermore, we provide our code within this supplementary material, which we will make publicly available upon acceptance of the paper.

A. Architectures

This section shows the model architectures for the noise predictor and the refinement network with further details on the training procedure. Appendix A.1 shows the diagram of the noise predictor model together with the condition encoder and how the noise prediction is conditioned to it. Appendix A.2 presents the refinement upsample network architecture and provides further details on the refinement network training.

A.1. Noise predictor

As the noise predictor, we used a MinkUNet [1] to predict the noise over each point. For the condition encoder, we used only the encoder part of the MinkUNet with the same architecture as the noise predictor. As described in Sec. 3.6 of the main paper, before each layer l, we compute the positional embeddings τ from the denoising step t with an embedding dimension $d_t = 96$, conditioning the layer input $\mathcal{F}_l$ to $\mathcal{C}$ and t with the conditioning block. Fig. 1 depicts the noise predictor and condition encoder architecture, with each layer l features dimension d_l and the conditioning scheme.

A.2. Refinement upsample network

As the refinement network, we have used the same architecture as the noise predictor with a $\tanh$ activation as the final layer, as depicted in Fig. 2. Given that the refinement network has to predict just an offset around the diffusion generation, we use a $\tanh$ layer to limit the offset size, avoiding the model predicting too large offsets.

As mentioned in Sec. 3.5 of the main paper, we used the refinement and upsample scheme proposed by Lyu *et al.* [5]. We train the refinement model using Adam [2] optimizer, with a learning rate of 10^{-4} and decay of 10^{-4}, with a batch size equal to 8, training for 5 epochs. To generate the refinement ground truth, we aggregate 20 scans before and 20 scans after each scan in the training set, using the relative poses between the scans. We use these aggregated scans as the ground truth $\mathcal{O}_{\text{gt}}$, and as the input, we copy $\mathcal{O}_{\text{gt}}$ and add random point jittering to each point, defining the input $\mathcal{O}$. Then, the model is trained to predict $3 \times \kappa$ values for each point, corresponding to κ offsets. We add the κ offsets to each point in $\mathcal{O}$, getting the upsampled refined prediction $\mathcal{O}'$, and supervise it with the symmetric chamfer distance loss $\mathcal{L}_{\text{refine}}$ as:

$$\mathcal{L}_{\text{CD}}\left(\mathcal{A}, \mathcal{B}\right) = \frac{1}{\mid \mathcal{A} \mid} \sum_{\boldsymbol{a} \in \mathcal{A}} \min_{\boldsymbol{b} \in \mathcal{B}} \|\boldsymbol{a} - \boldsymbol{b}\|_2^2, \tag{1}$$

$$\mathcal{L}_{\text{refine}} = \mathcal{L}_{\text{CD}}(\mathcal{O}_{\text{gt}}, \mathcal{O}') + \mathcal{L}_{\text{CD}}(\mathcal{O}', \mathcal{O}_{\text{gt}}). \tag{2}$$

With Eq. (2), we train the refinement model to predict κ offsets to the input $\mathcal{O}$ such that the upsampled refined prediction $\mathcal{O}'$ gets as close as possible to the ground truth. With this refinement model, we can generate the scene completion with our diffusion model with fewer denoising steps using the DPMSolver [4] and refine it. As mentioned in Sec. 3.5 of the main paper, with fewer denoising steps, the generation quality may decrease. Therefore, with this refinement network, we can compensate for this lower generation quality while also upsampling our generated scene completion.

B. Regularization ablation

This section compares the results of the noise predictor trained with different regularization weights r. Fig. 3 compares the scene completion with the noise predictor trained with different regularization weights. With $r = 0.0$, the model can generate structural information with a noisy aspect, and, in this example, the points from the two parked cars are mixed together without a clear boundary. With $r = 1.0$, a less noisy scene completion is generated, but still, the surfaces in the structure present a noisy aspect. When comparing $r = 3.0$ and $r = 5.0$, both generated scene depicts a more detailed and less noisy scene, compared with lower regularization weights r. However, using $r = 5.0$ achieves more fine-grained structural details. The surfaces in the scene appear to have a flatter aspect, and the

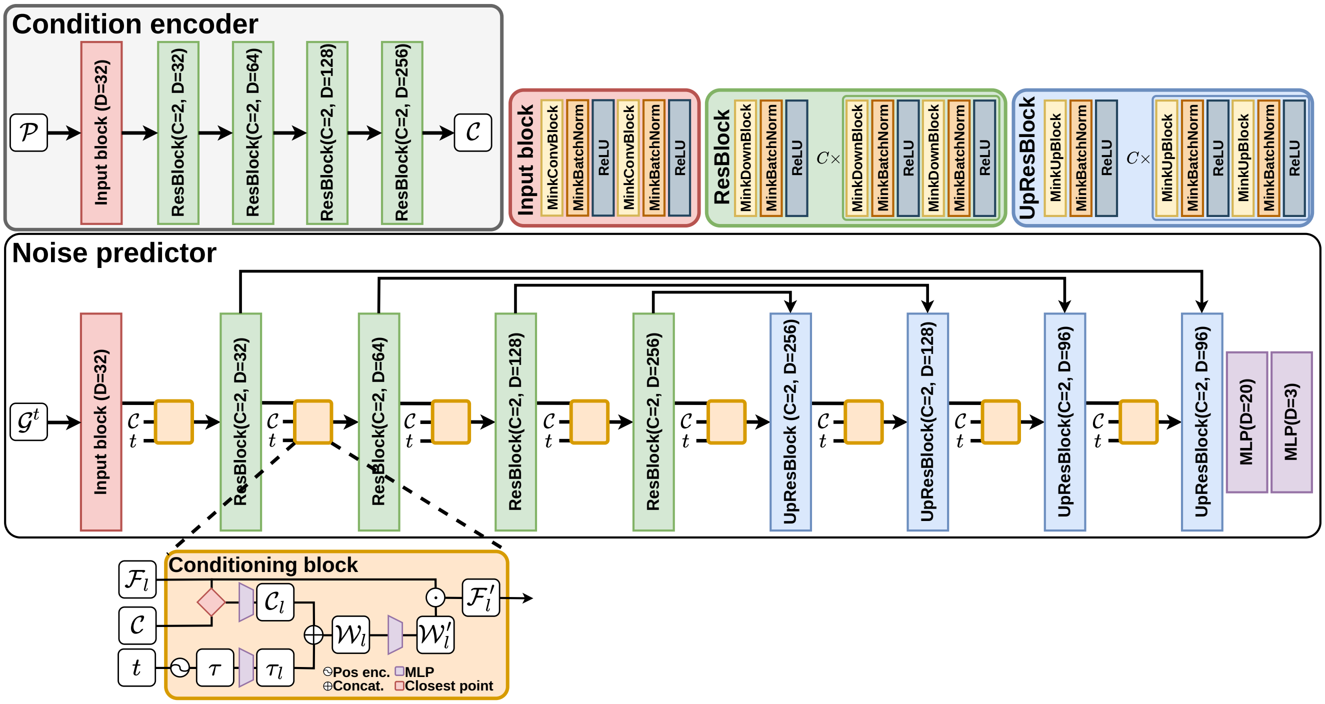

Figure 1. Noise predictor and condition encoder models architecture. The condition encoder receives the scan $\mathcal{P}$ and computes the conditioning point cloud $\mathcal{C}$. From t, we compute the positional embedding τ with a dimension $d_t = 96$. At each layer l, we give $\mathcal{C}$ and τ to the conditioning block together with the layer input features $\mathcal{F}_l$ to get $\mathcal{F}'_l$, which is then feed as input to the layer l.

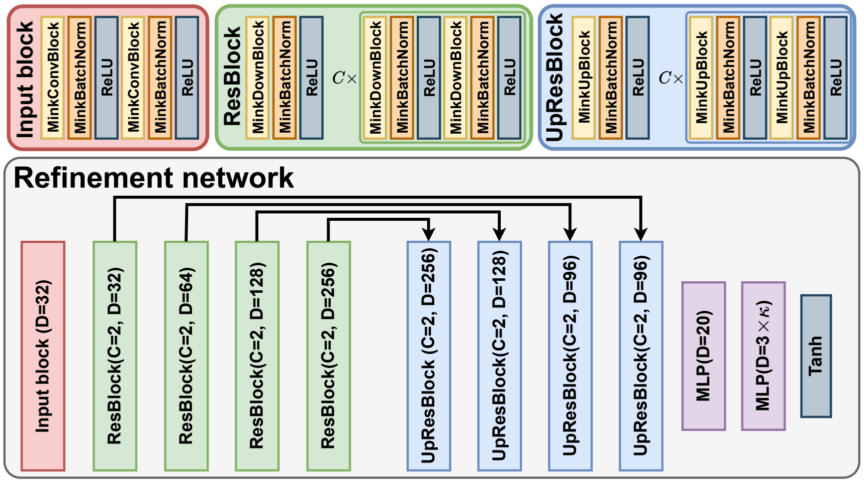

Figure 2. Refinement network architecture.

sidewalk curbs seem better defined. Also, the two parked cars retain more details, *e.g.*, the windows space. Given this analysis and the quantitative results present in Tab. 5 of the main paper, we use $r = 5.0$ in the main experiments.

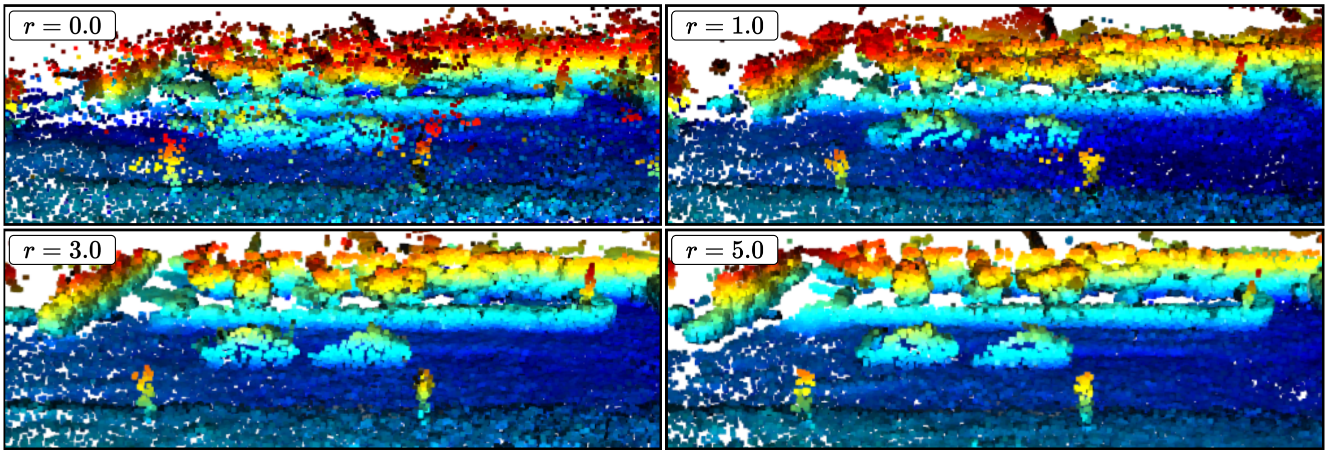

Figure 3. Comparison between results with different regularization weights r.

s	0.0	2.0	4.0	6.0	10.0	12.0	16.0
CD [m]	0.737	0.543	0.454	**0.433**	**0.432**	0.435	0.450

Table 1. Mean chamfer distance over a short sequence from the validation set of SemanticKITTI with different conditioning weights s.

C. Condition weights ablation

This section compares the scene completion quality using different condition weights s qualitatively and quantitatively. Fig. 4 shows the qualitative comparison between the scene completion with different conditioning weights. With $s = 0.0$, we have the unconditional generation. In this case, the generated scene has a flat surface distributed over the input scan borders without retaining structural information. As we increase s, the structure details are better defined. With $s = 2.0$ and $s = 4.0$ more details are generated but with a smooth aspect. With $s = 6.0$ the generation follows structural information from the input scan and defines sharper boundaries over the structures. With $s = 10.0$ and $s = 12.0$, the generated scene gets too noisy, generating artifacts over the scene.

We also evaluate the influence of the conditioning weight s in Tab. 1. As in Tab. 5 of the main paper, we compute the chamfer distance over the scene completion and the ground truth over a short sequence from the SemanticKITTI validation set, where we generate every one hundred scans. In this evaluation, having $s = 6.0$ and $s = 10.0$ achieves basically the same performance. However, from the qualitative evaluation presented in Fig. 4, we used $s = 6.0$ in the main paper since it achieved the best performance visually and numerically.

D. Denoising steps

In this section, we compare the quality of the scene completion with different number of denoising steps T. Fig. 5 shows the diffusion generation using DPMSolver [4] with the different number of denoising steps and the amount of time in seconds to generate the complete scene. Since the model was trained with $T = 1,000$, we can achieve the best quality result when using $T = 1,000$ during inference. However, inferring the $1,000$ steps demands many computational time. As we decrease T, we increase the inference speed. However, we can also notice that with lower T, the scene generation loses details. This can be seen when comparing the structures in the scene, especially the ground, where more noise can be noticed as we decrease T. Therefore, in the main paper we set $T = 50$ and take advantage of the refinement network to compensate for the lower quality generation when using smaller T.

E. Further qualitative results

In this section, we show more qualitative results, comparing our scene completion with the baselines evaluated in the paper, *i.e.*, LMSCNet [6], PVD [8], Make It Dense (MID) [7], and LODE [3]. Figs. 6 to 10 compare the results between the baselines and our method. As shown, the diffusion baseline PVD [8] fails to generate scene-scale data. The SDF baselines reconstruct the scene inheriting artifacts from the surface approximation and the voxelization. Our method achieves a more detailed representation, with a smoother generation compared to the baselines.

References

[1] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neu-

ral Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[2] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2015.

[3] Pengfei Li, Ruowen Zhao, Yongliang Shi, Hao Zhao, Jirui Yuan, Guyue Zhou, and Ya-Qin Zhang. LODE Locally Conditioned Eikonal Implicit Scene Completion from Sparse LiDAR. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[4] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022.

[5] Zhaoyang Lyu, Zhifeng Kong, Xudong XU, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2022.

[6] Luis Roldão, Raoul de Charette, and Anne Verroust-Blondet. LMSCNet: Lightweight Multiscale 3D Semantic Completion. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2020.

[7] Ignacio Vizzo, Benedikt Mersch, Rodrigo Marcuzzi, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Make it dense: Self-supervised geometric scan completion of sparse 3d lidar scans in large outdoor environments. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):8534–8541, 2022.

[8] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D Shape Generation and Completion Through Point-Voxel Diffusion. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

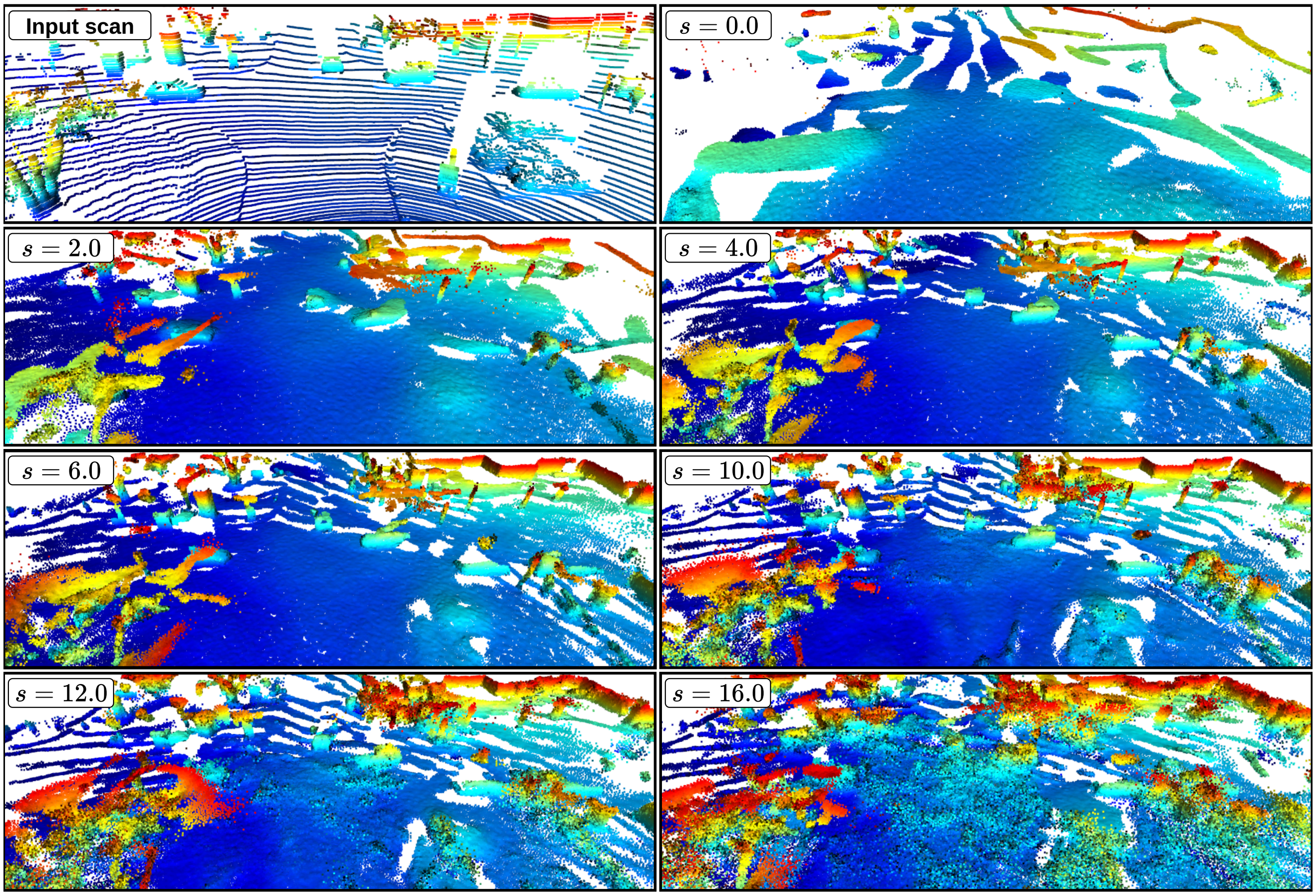

Figure 4. Comparison between results with different conditioning weights s.

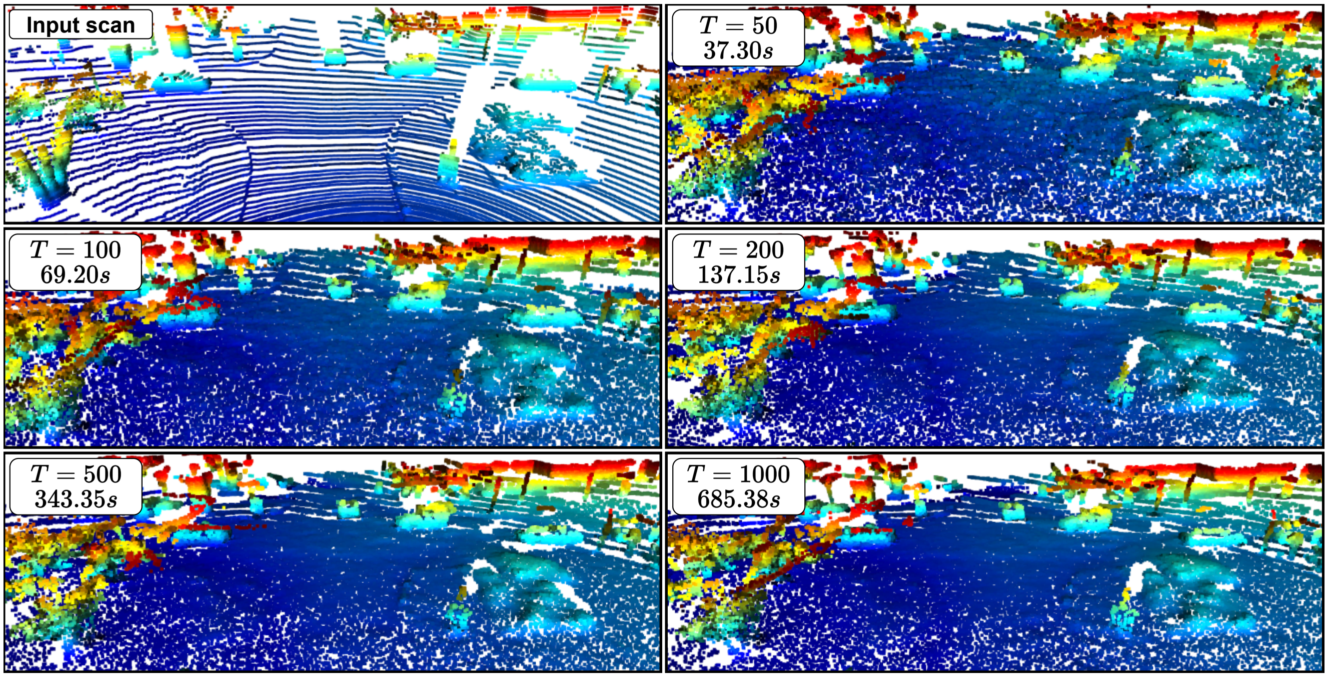

Figure 5. Comparison between results with different number of denoising steps T.

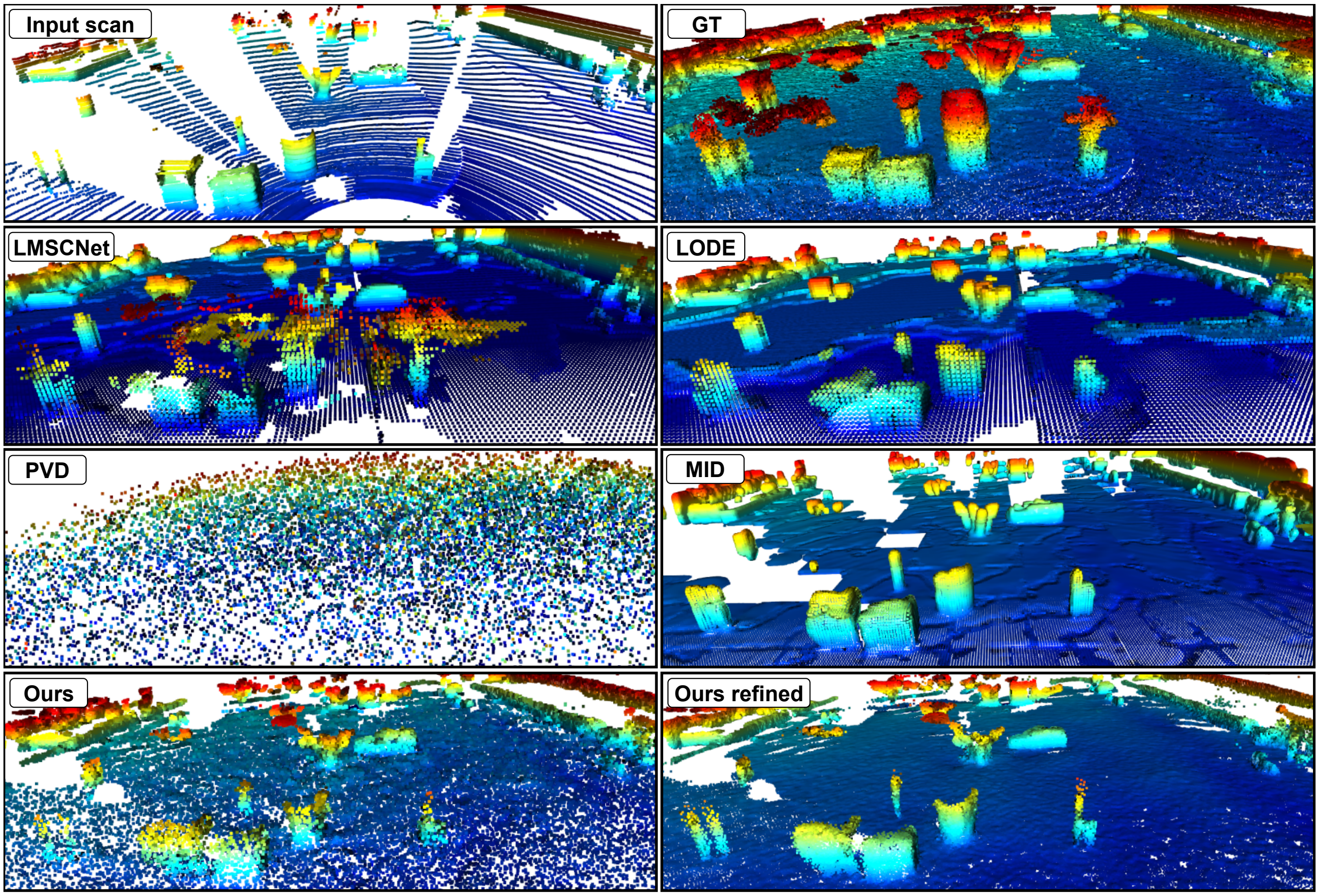

Figure 6. Qualitative results comparing the scene completion between our method and the baselines evaluated in the main paper.

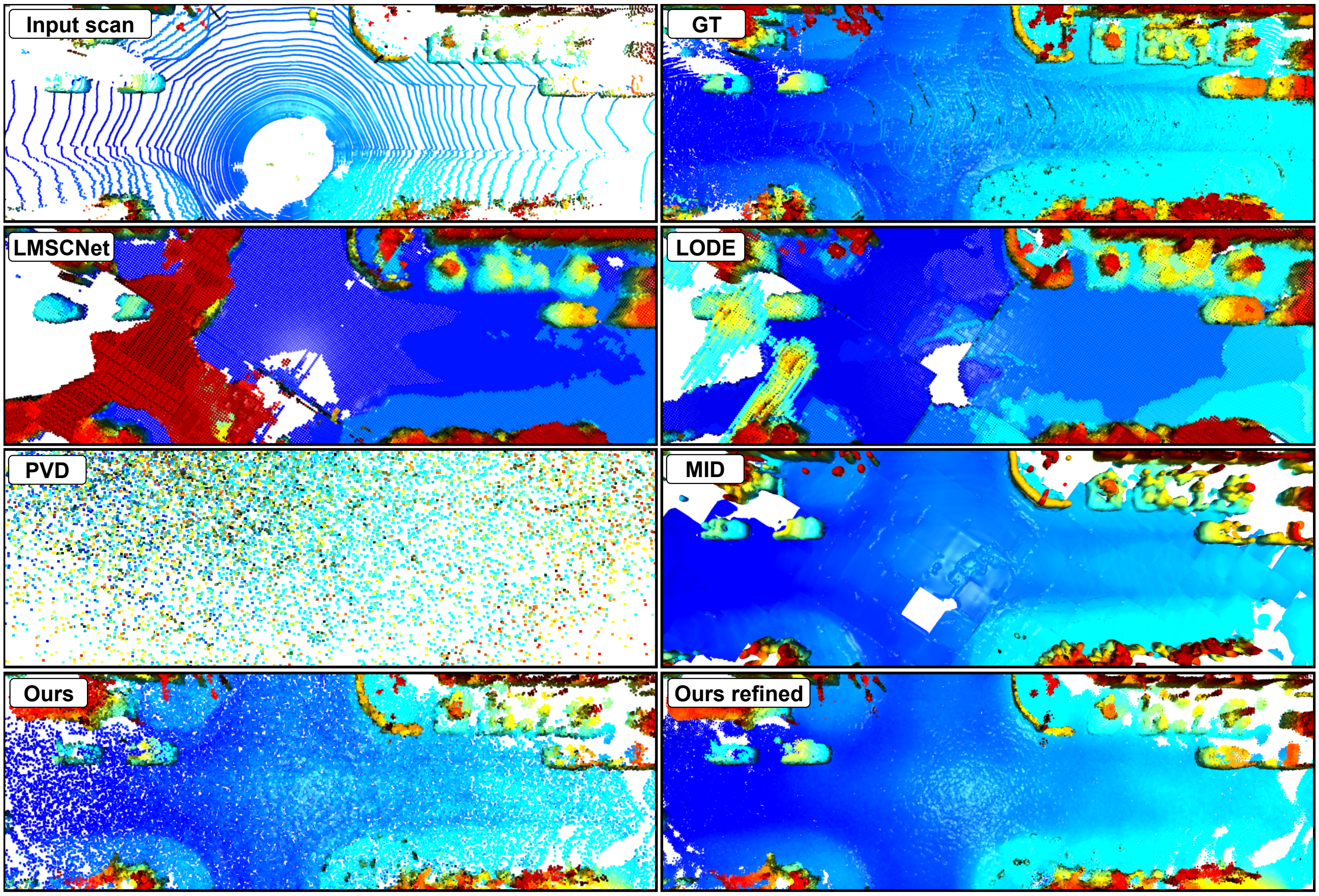

Figure 7. Qualitative results comparing the scene completion between our method and the baselines evaluated in the main paper.

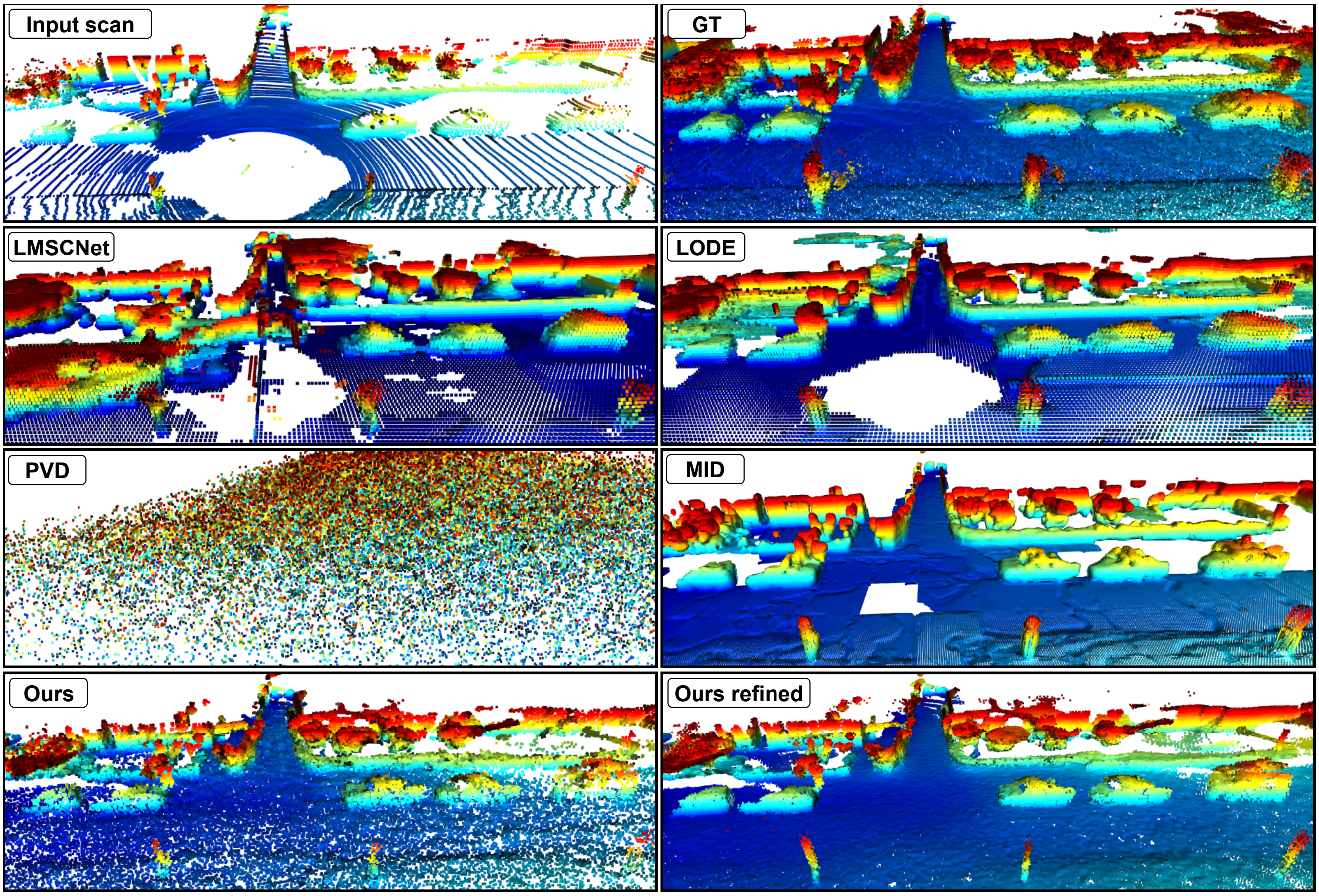

Figure 8. Qualitative results comparing the scene completion between our method and the baselines evaluated in the main paper.

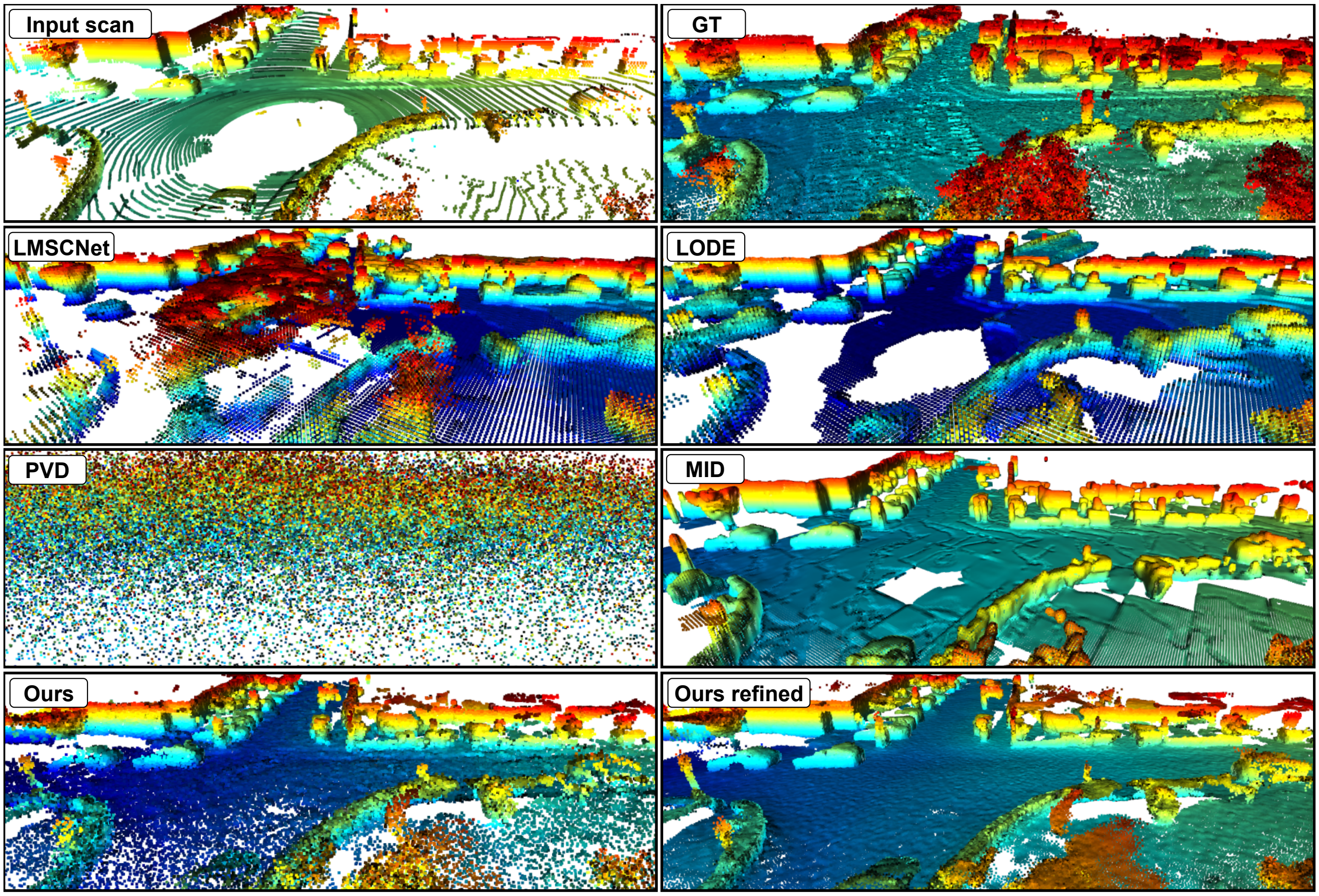

Figure 9. Qualitative results comparing the scene completion between our method and the baselines evaluated in the main paper.

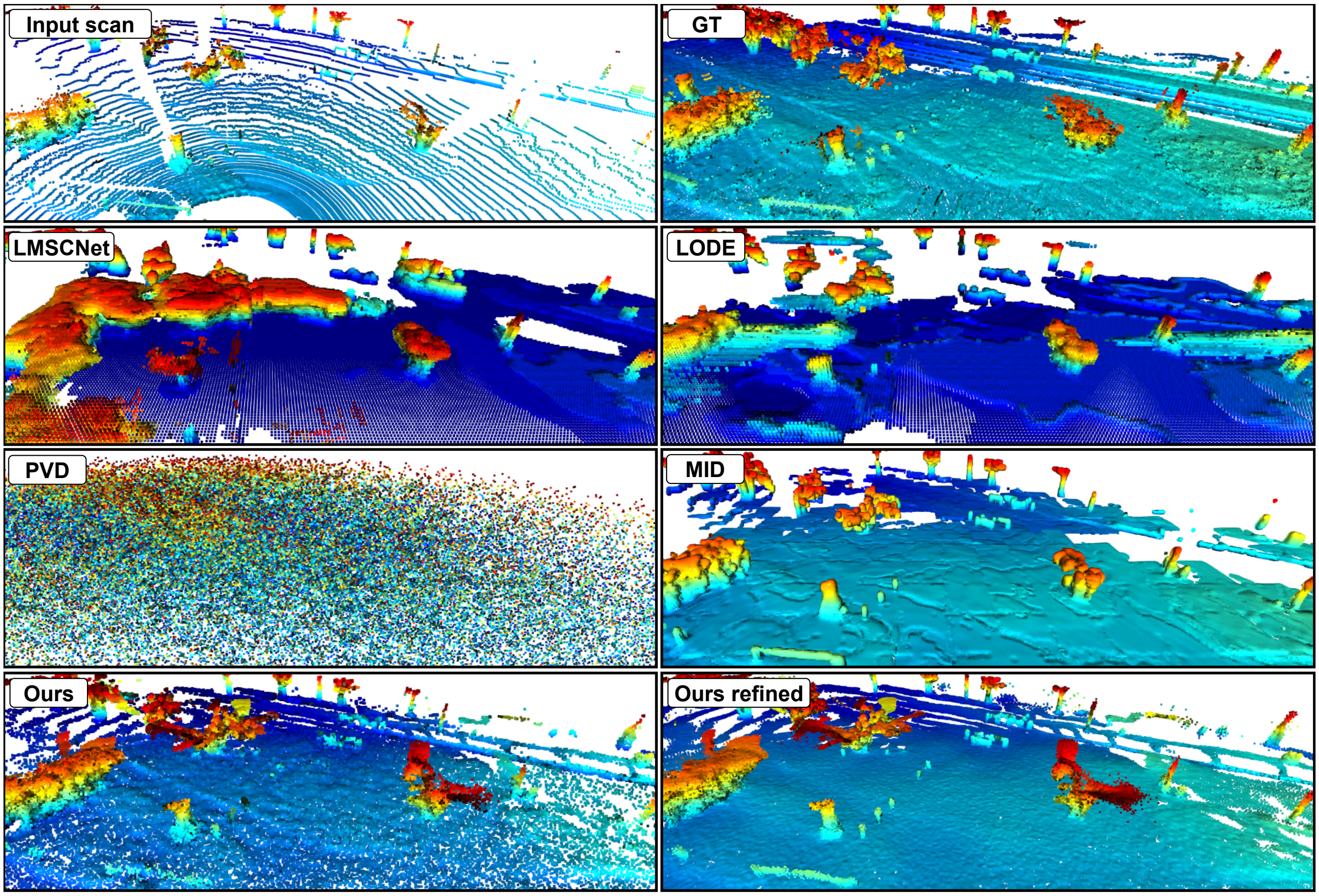

Figure 10. Qualitative results comparing the scene completion between our method and the baselines evaluated in the main paper.