Appendix for "OGNI-DC: Robust Depth Completion with Optimization-Guided Neural Iterations"

Yiming Zuo and Jia Deng

Department of Computer Science, Princeton University
{zuoym,jiadeng}@princeton.edu

A More Ablation Studies on Generalization

Table a: We conduct Ablation studies in the in-domain (NYUv2, 500 points), sparser inputs (NYUv2, 100 points), denser inputs (NYUv2, 10 000 points), and cross-dataset (VOID, 500 points) scenarios. All metrics are in [mm].

Test Datasets		NYUv2 [6]		NYUv2 [6]		NYUv2 [6]		VOID [11]	
Points		500		100		10 000		500	
Methods		RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
(a)	Without DDI	128.6	44.9	199.0	87.9	73.5	22.6	620.5	237.9
Ours	With DDI	**112.2**	**38.0**	**171.5**	**72.7**	**53.0**	**13.9**	**605.0**	**229.0**
(b)	1 GRU iteration	114.0	39.9	171.7	73.0	54.0	14.3	634.6	241.5
(c)	3 GRU iterations	112.4	38.2	171.6	73.0	54.1	14.3	641.5	247.1
Ours	5 GRU iterations	**112.2**	**38.0**	**171.5**	72.7	53.0	13.9	**605.0**	**229.0**
(d)	7 GRU iterations	112.3	**38.0**	172.5	**71.8**	**52.0**	**13.6**	649.2	257.5
(e)	RGB backbone	121.3	43.1	176.9	79.4	67.8	18.1	659.9	273.9
(f)	RGBD (w/o masking)	**110.1**	**36.9**	292.3	149.1	59.1	15.1	792.8	371.6
Ours	RGBD (w/ masking)	112.2	38.0	**171.5**	**72.7**	**53.0**	**13.9**	**605.0**	**229.0**
(g)	Without SPN	113.0	38.3	**170.9**	**71.3**	68.1	21.3	662.8	265.2
(h)	With NLSPN [7]	112.8	38.5	179.0	76.2	64.1	20.5	693.2	286.0
Ours	With DySPN [5]	**112.2**	**38.0**	171.5	72.7	**53.0**	**13.9**	**605.0**	**229.0**
(i)	Without $\hat{\mathbf{D}}^{\text{up}}$ loss	113.2	38.6	178.7	77.0	55.7	14.4	643.2	252.8
(j)	Without $\hat{\mathbf{G}}$ loss	112.5	38.1	**171.5**	**71.7**	54.1	14.0	634.0	252.2
Ours	With both losses	**112.2**	**38.0**	**171.5**	72.7	**53.0**	**13.9**	**605.0**	**229.0**

To better understand how the generalization of OGNI-DC is affected by different components in our depth completion pipeline, we conduct ablation studies by varying the sparsity levels and datasets. Specifically, we test models on the NYUv2 [6] validation set with sparser inputs (randomly sampled 100 points)

and denser inputs (randomly sampled 10 000 points). We further test the cross-dataset generalization by constructing a validation set for the VOID [11] dataset by randomly choosing 800 images from the VOID500 [11] training set. Finally, for easier reference, we copy the numbers from the main paper for the in-domain cases (NYUv2 [6], randomly sampled 500 points). Results are shown in Tab. a.

DDI. We train a baseline model without DDI, where the ConvGRU directly generates updates on the depth map. Comparing (a) to ours, our model with DDI outperforms the baseline model by large margins under all settings, proving that DDI is the key to achieving both in-domain accuracy and strong generalization.

Iterative Refinement. To prove the effectiveness of iterative refinement, we train different models where the ConvGRU are unrolled $1 \sim 7$ times. Compared to 1 or 3 iterations, 5 iterations consistently improve the performance under all settings. For example, when tested on VOID, 5 iterations improve MAE from 634.6mm to 605.0mm compared to 1 iteration. The benefits of further unrolling 7 iterations are not clear, as 7 iterations achieve slightly better results in the NYUv2 10 000 points case and worse results when tested on VOID. Therefore, we use 5 iterations as a good balance between performance and speed.

Backbone Inputs. We test against the baseline where we input only the RGB image to the backbone ((e), RGB backbone). We further test against a baseline where we input both the image and the sparse depth to the backbone, but do not randomly mask out the sparse depth map during training ((f), RGBD (w/o masking)). Our full model consistently outperforms the RGB baseline, proving that sparse depth inputs are helpful for depth gradients prediction. Compared to ours, the model without masking archives better in-domain performance, but has worse generalization overall. In conclusion, RGBD w/masking provides the best balance between in-domain performance and generalization.

SPN. We ablate the effect of the SPN layer in our model. Comparing (g) $\sim$ (h), the model without an SPN works best under 100 samples on NYUv2, whereas the model with DySPN [5] works best for 500 points, 10 000 points, and on VOID. The model with NLSPN [7] works worse in all cases. In conclusion, our model works best with DySPN [5], but can also work well without an SPN.

Auxiliary Losses. Comparing (i) $\sim$ (j) to ours, supervising the up-sampled depth $\hat{\mathbf{D}}^{\text{up}}$ and the depth gradients $\hat{\mathbf{G}}$ both contribute to better performance under all test setting. The contribution of the supervision on $\hat{\mathbf{D}}^{\text{up}}$ is more significant. That's probably because supervising the output of the SPN layer is not enough for regularizing its input, and therefore intermediate supervisions are necessary.

B KITTI Sparsity Genealization with Retrained Models

To analyze our model's performance more thoroughly on sparser inputs, we run experiments under the setup of CFormer [12], where all models are *retrained* on sub-sampled lidar lines and tested under the same sparsity. To be consistent with CFormer [12], instead of using the entire training set, we use the file list they

provide, where they randomly sample 10 000 training images. We test under the sparsity of 8, 16, 32, and 64 lines. We don't test even sparser inputs because no autonomous driving vehicles are equipped with Lidar sparser than 8 lines.

Results are shown in Tab. b. The 64-Lines and 16-Lines numbers are copied from CFormer [12]. The 32-Lines and 8-Lines numbers are reproduced by ourselves with their official training code. Our model still consistently outperforms all baselines under all sparsity levels in this retrain setting.

Table b: Robustness to the number of Lidar lines on the KITTI [10] validation dataset. All methods are *retrained* under the corresponding sparsity levels with 10 000 images. All metrics are in [mm]. Our model consistently outperforms baselines.

Lidar Scans	64-Lines		32-Lines		16-Lines		8-Lines	
Methods	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
NLSPN [7]	889.4	238.8	1052.2	285.0	1288.9	377.2	1584.0	501.9
DySPN [5]	878.5	228.6	-	-	1274.8	366.4	-	-
CFormer [12]	848.7	215.9	994.8	265.8	1218.6	337.4	1513.2	457.7
Ours	**813.7**	**205.4**	**967.9**	**252.3**	**1196.7**	**324.3**	**1510.6**	**444.0**

C Details of the Differentiable Depth Integrator (DDI)

C.1 Backward Pass Details

Let $\mathcal{L}$ be the loss of the network. The input to the backward function is its gradient on $\hat{\mathbf{D}}$, *i.e.*, $\partial\mathcal{L}/\partial\hat{\mathbf{D}}$. We want to compute $\partial\mathcal{L}/\partial\hat{\mathbf{G}}$. Recall from the paper that we have:

$$\frac{\partial\overline{\hat{\mathbf{D}}}}{\partial\mathbf{b}} = (\mathbf{A}^\intercal\mathbf{A})^{-1}\mathbf{A}^\intercal,\ \frac{\partial\mathbf{b}}{\partial\overline{\hat{\mathbf{G}}}} = \begin{pmatrix} \mathbf{I}_{H(W-1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{(H-1)W} & \mathbf{0} \end{pmatrix}^\intercal. \tag{1}$$

Therefore,

$$\frac{\partial\mathcal{L}}{\partial\overline{\hat{\mathbf{G}}}} = \frac{\partial\mathcal{L}}{\partial\overline{\hat{\mathbf{D}}}} \cdot \frac{\partial\overline{\hat{\mathbf{D}}}}{\partial\mathbf{b}} \cdot \frac{\partial\mathbf{b}}{\partial\overline{\hat{\mathbf{G}}}} \tag{2}$$

$$= \left((\mathbf{A}^\intercal\mathbf{A})^{-1} \cdot \left(\frac{\partial\mathcal{L}}{\partial\overline{\hat{\mathbf{D}}}}\right)^\intercal\right)^\intercal \cdot \mathbf{A}^\intercal \cdot \begin{pmatrix} \mathbf{I}_{H(W-1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{(H-1)W} & \mathbf{0} \end{pmatrix}^\intercal. \tag{3}$$

The first part can be computed using the same conjugate gradient solver as in the forward pass. Since $\mathbf{A}^\intercal$ and $\partial\mathbf{b}/\partial\overline{\hat{\mathbf{G}}}$ are sparse, the rest of the matrix multiplications can be done efficiently.

C.2 Speed Optimizations

Stopping Conditions. We stop the conjugate gradient solver when the optimal solution is found, *i.e.*, when the relative residual is smaller than $1e-5$ or the residual has no improvement more than 1% for 10 steps. This avoids unnecessary conjugate gradient steps compared to optimizing for a fixed number of steps.

Initialization from Current Solution. We assume the refinements on depth gradients to be small for each iteration, therefore the correct solution can be used as an initial guess for the next round. We store $\hat{\mathbf{D}}_t$ as a dummy variable and use it to be the initial value of $\hat{\mathbf{D}}_{t+1}$ in the conjugate gradient iterations. We do a similar thing for the backward pass by storing $(\mathbf{A}^\intercal\mathbf{A})^{-1}\cdot\partial\mathcal{L}/\partial\overline{\hat{\mathbf{D}}}$, but this time using the result from the $(t+1)$-th step to initialize the t-th step.

C.3 Confidence on Sparse Depth Observations

The sparse depth observations in the KITTI [10] dataset contain bleeding artifacts due to the baseline between the camera and the Lidar sensor (see Fig. a (g) for an example). This has been observed in several previous works [2, 7, 8]. This issue is especially critical for OGNI-DC, since the noisy inputs directly affect the outputs of DDI and the error cannot be corrected. Therefore, we predict a confidence map $\hat{\mathbf{C}} \in [0,1]^{\frac{H}{4}\times\frac{W}{4}}$ for the sparse depth observations from the $1/4$ resolution feature with a Conv-Sigmoid layer. $\hat{\mathbf{C}}$ works by down-weighting the observation energy term for the noisy pixels in the optimization problem, *i.e.*,

$$\overline{\hat{\mathbf{D}}} = \arg\min_{\overline{\mathbf{D}}} \left\| \underbrace{\begin{pmatrix} \overline{\boldsymbol{\Delta}_{\mathbf{x}}} \\ \overline{\boldsymbol{\Delta}_{\mathbf{y}}} \\ \operatorname{diag}\left(\sqrt{\alpha}\cdot\overline{\sqrt{\hat{\mathbf{C}}}}\cdot\overline{\mathbf{M}}\right) \end{pmatrix}}_{\mathbf{A}} \overline{\mathbf{D}} - \underbrace{\begin{pmatrix} \overline{\hat{\mathbf{G}}^{\mathbf{x}}} \\ \overline{\hat{\mathbf{G}}^{\mathbf{y}}} \\ \sqrt{\alpha}\cdot\overline{\sqrt{\hat{\mathbf{C}}}}\cdot\overline{\mathbf{M}}\cdot\overline{\mathbf{O}} \end{pmatrix}}_{\mathbf{b}} \right\|_2^2 . \tag{4}$$

Applying $\hat{\mathbf{C}}$ in the forward pass is straightforward. However, since we don't have ground truth for $\hat{\mathbf{C}}$, we must learn $\hat{\mathbf{C}}$ directly from the loss on the integrated depth map. Therefore, we have to compute the gradient of $\overline{\hat{\mathbf{D}}}$ with respect to $\hat{\mathbf{C}}$. This can be done by applying the chain rule (remember $\overline{\hat{\mathbf{D}}} = (\mathbf{A}^\intercal\mathbf{A})^{-1}\mathbf{A}^\intercal\mathbf{b}$):

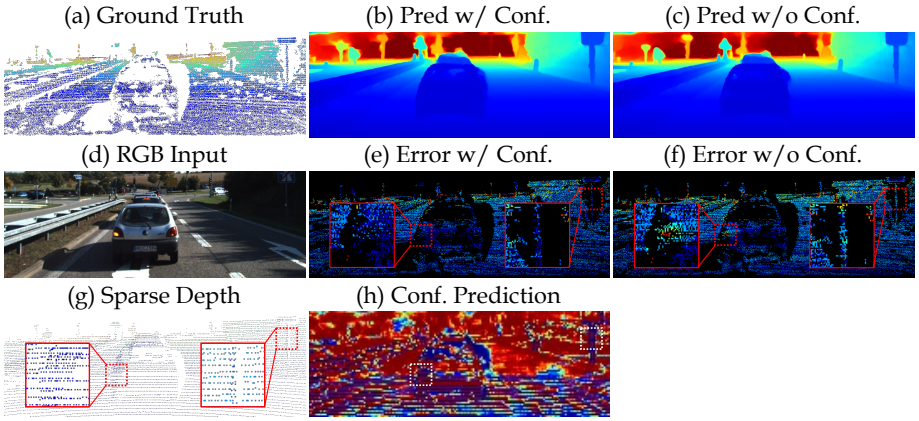

Fig. a: Our model learns a confidence map (h) to filter out the noisy sparse observations in (g). Compare (e) to (f), the model with confidence prediction is more accurate.

$$(\mathbf{A}^\intercal\mathbf{A})\cdot\overline{\hat{\mathbf{D}}}=\mathbf{A}^\intercal\mathbf{b}\;\Rightarrow\;\frac{\partial(\mathbf{A}^\intercal\mathbf{A})}{\partial\overline{\hat{\mathbf{C}}}}\cdot\overline{\hat{\mathbf{D}}}+(\mathbf{A}^\intercal\mathbf{A})\cdot\frac{\partial\overline{\hat{\mathbf{D}}}}{\partial\overline{\hat{\mathbf{C}}}}=\frac{\partial(\mathbf{A}^\intercal\mathbf{b})}{\partial\overline{\hat{\mathbf{C}}}} \tag{5}$$

$$\Rightarrow\frac{\partial\overline{\hat{\mathbf{D}}}}{\partial\overline{\hat{\mathbf{C}}}}=(\mathbf{A}^\intercal\mathbf{A})^{-1}\left(\frac{\partial(\mathbf{A}^\intercal\mathbf{b})}{\partial\overline{\hat{\mathbf{C}}}}-\frac{\partial(\mathbf{A}^\intercal\mathbf{A})}{\partial\overline{\hat{\mathbf{C}}}}\cdot\overline{\hat{\mathbf{D}}}\right), \tag{6}$$

$$\text{where}\quad\frac{\partial(\mathbf{A}^\intercal\mathbf{b})}{\partial\overline{\hat{\mathbf{C}}}}=\operatorname{diag}\left(\alpha\cdot\overline{\mathbf{M}}\cdot\overline{\mathbf{O}}\right),\quad\left[\frac{\partial(\mathbf{A}^\intercal\mathbf{A})}{\partial\overline{\hat{\mathbf{C}}}}\right]_{ijk}=\alpha\cdot\overline{\mathbf{M}}_i\cdot\mathbb{1}_{i=j=k}. \tag{7}$$

We demonstrate the effectiveness of the predicted confidence map in Fig. a. Fig. a (h) shows that confidence is high (red) in most areas while being low (blue) at the noisy regions on the car and the traffic sign. Comparing Fig. a (e) with Fig. a (f), the errors are greatly reduced in those areas with predicted confidence map. The results show that our model successfully learned to filter out the noisy inputs even without the ground truth.

D Network Architecture

We use the CompletionFormer [12] as our backbone. CompletionFormer is a U-Net-like architecture with a series of down-sample and up-sample layers. The architecture of our update block is illustrated in Fig. b. We use the same ConvGRU as RAFT [9]. Please refer to [9] for details.

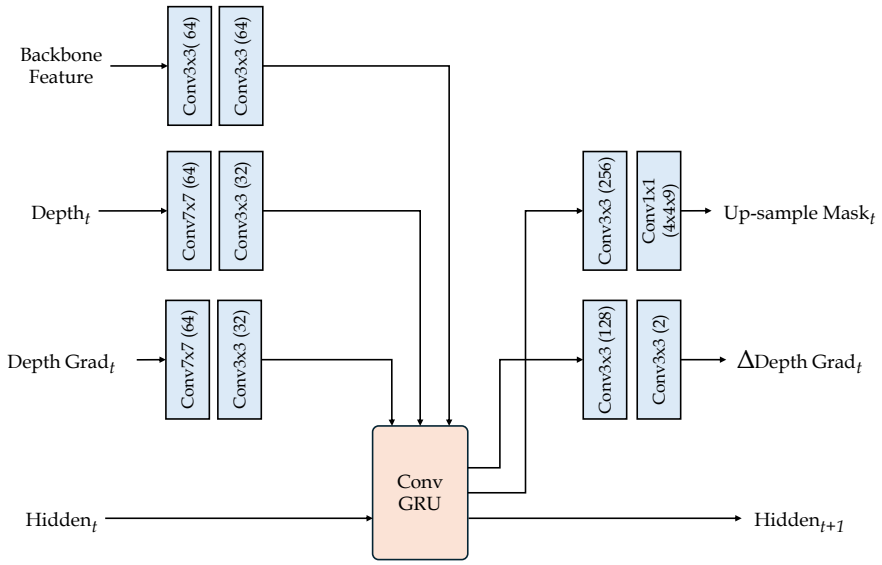

Fig. b: The detailed architecture of our update unit.

E More Visualizations

E.1 NYUv2 Sparsity Generalization

We qualitatively compare our method's generalization ability to different sparsity levels on the NYUv2 [6] dataset with NLSPN [7], CFormer [12], and SpAgNet [3]. Results are shown in Fig. c and Fig. d. This image is the first one in the NYUv2 [6] test set and is **not cherry-picked**. Our model works better than baselines under all sparsity levels.

E.2 KITTI Sparsity Generalization

Results are shown in Fig. e and Fig. f. This is the first image in the KITTI validation set and is **not cherry-picked**. While all methods do equally well with 64-Lines input, our method is significantly better when the inputs are sparser.

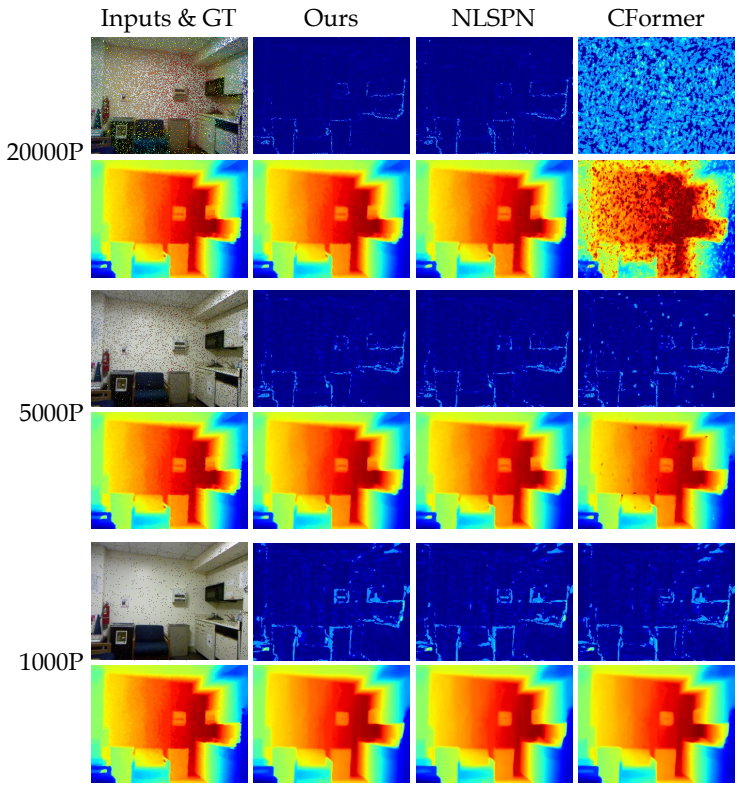

Fig. c: Generalization to denser inputs on NYUv2 [6]. For each sparsity level, the first row is the inputs/error maps, and the second row is the gt/predicted depths. Our model works better than baselines although not trained on these sparsity levels.

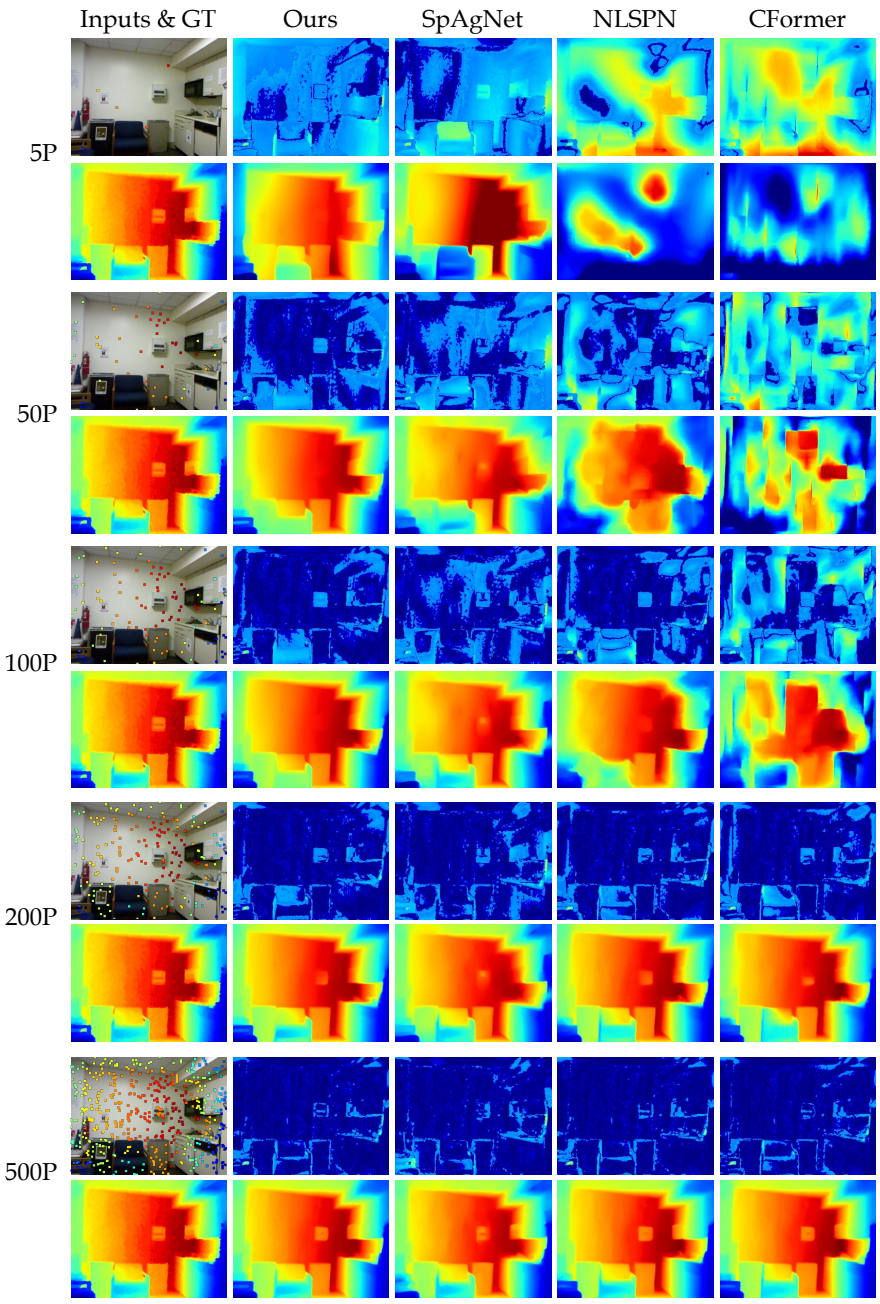

Fig. d: Generalization to sparser inputs on NYUv2 [6]. For each sparsity level, the first row is the inputs/error maps, and the second row is the gt/predicted depths. Our model consistently outperforms baselines under all sparsity levels.

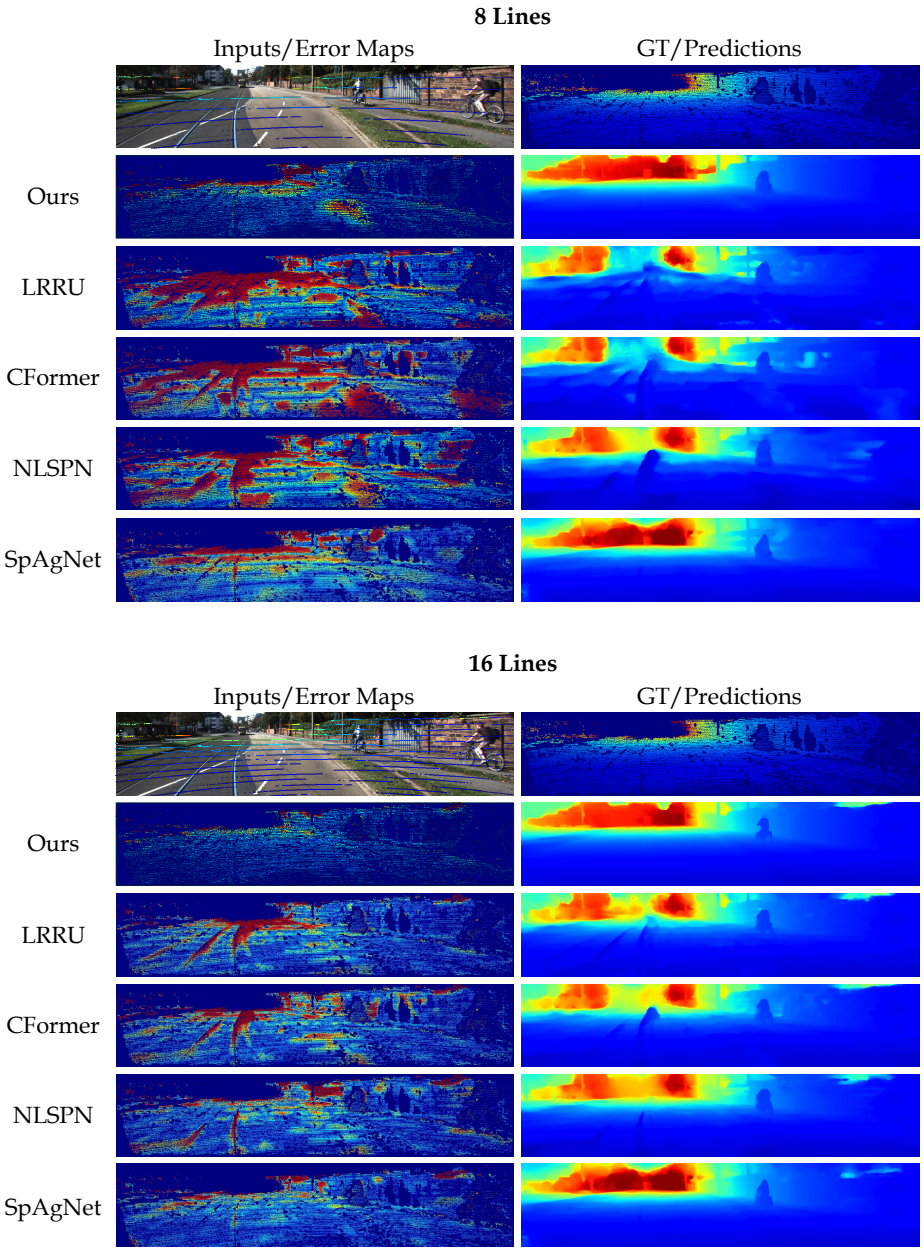

Fig. e: KITTI [10] results with 8 and 16 lines inputs. Red colors mean larger errors.

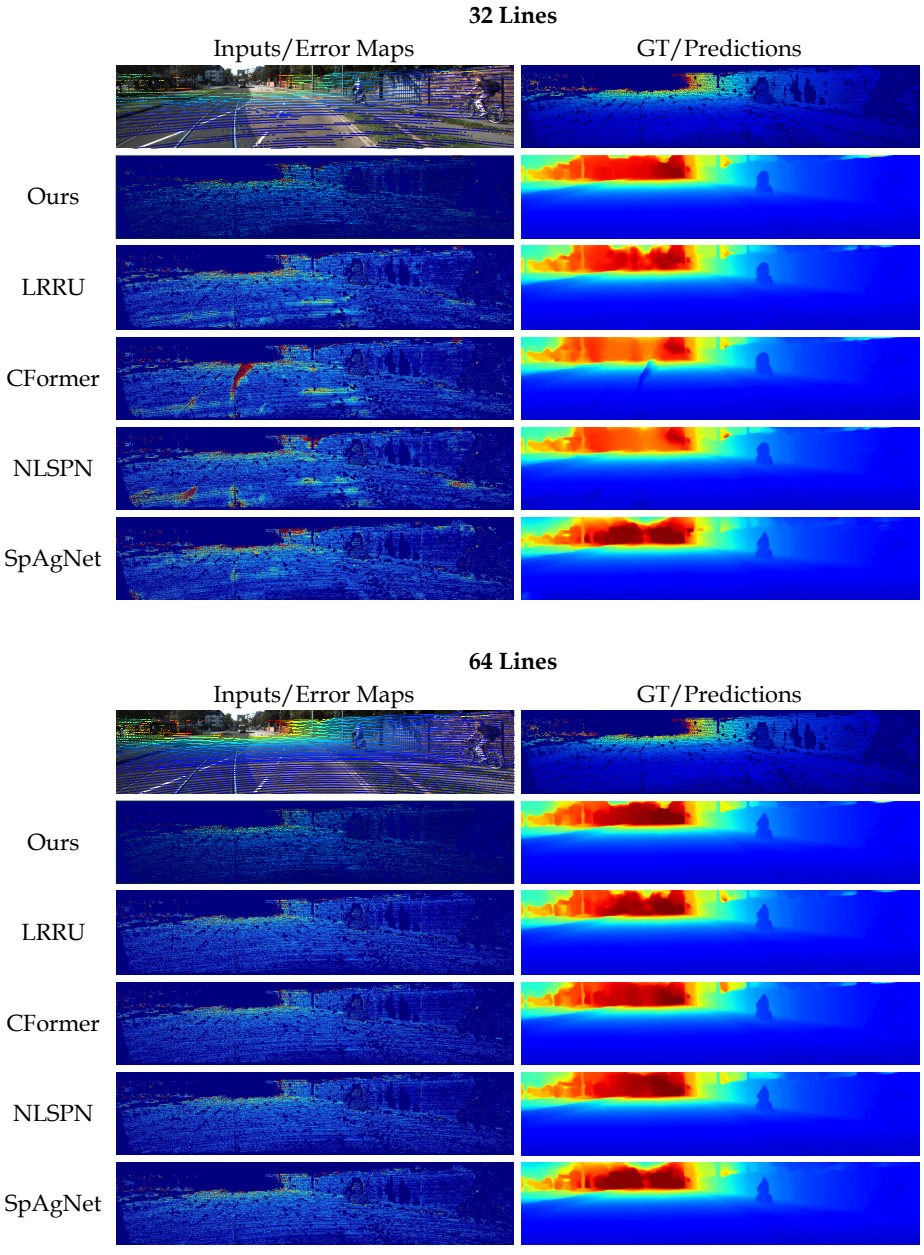

Fig. f: KITTI [10] results with 32 and 64 lines inputs. Red colors mean larger errors.

F Dataset Descriptions

NYUv2. NYUv2 [6] contains 45 205 training images, 2 379 validation images, and 654 test images from 464 indoor scenes. Dense depth maps are collected with the Microsoft Kinect sensor, and 500 points are randomly sampled to provide sparse observations. Following previous works [7, 12], we resize the original 480×640 images to 240×320 and then center-crop to 228×304.

KITTI. The KITTI depth completion dataset [10] contains 86 898 training images, 1 000 selected validation images, and 1 000 online test images. Images and depths are collected from an autonomous driving vehicle with a Velodyne HDL-64E Lidar sensor. All images have resolution 352×1216. Following previous works [7, 12], during training and validation, the images are bottom-cropped to 240×1216 as no Lidar points are available in the sky areas.

VOID. The VOID [11] dataset contains 56 sequences of indoor scenes. Depth ground truths are collected with an Intel RealSense D435i camera, and sparse observations are from a visual odometry system at 3 different sparsity levels, *i.e.*, 1 500, 500, and 150 points, corresponding to 0.5%, 0.15%, and 0.05% density. Each test split contains 800 images at 480×640 resolution.

DDAD. DDAD [4] is an autonomous driving dataset with depth ground truth captured by a long-range, high-resolution Luminar-H2 Lidar. Following the split and pre-processing of VPP4DC [1], we evaluate on 3 950 images under 1216×1936 resolution captured by the front-viewing camera. We randomly sample about 20% points as the input sparse depth, resulting in $\sim 0.21\%$ density.

References

1. Bartolomei, L., Poggi, M., Conti, A., Tosi, F., Mattoccia, S.: Revisiting depth completion from a stereo matching perspective for cross-domain generalization. arXiv preprint arXiv:2312.09254 (2023)
2. Conti, A., Poggi, M., Aleotti, F., Mattoccia, S.: Unsupervised confidence for lidar depth maps and applications. In: IROS. pp. 8352–8359. IEEE (2022)
3. Conti, A., Poggi, M., Mattoccia, S.: Sparsity agnostic depth completion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5871–5880 (2023)
4. Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., Gaidon, A.: 3d packing for self-supervised monocular depth estimation. In: CVPR (2020)
5. Lin, Y., Cheng, T., Zhong, Q., Zhou, W., Yang, H.: Dynamic spatial propagation network for depth completion. In: AAAI (2022)
6. Nathan Silberman, Derek Hoiem, P.K., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV (2012)
7. Park, J., Joo, K., Hu, Z., Liu, C.K., So Kweon, I.: Non-local spatial propagation network for depth completion. In: ECCV (2020)
8. Qiu, J., Cui, Z., Zhang, Y., Zhang, X., Liu, S., Zeng, B., Pollefeys, M.: Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In: CVPR (2019)
9. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: ECCV. pp. 402–419. Springer (2020)
10. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: 3DV (2017)
11. Wong, A., Fei, X., Tsuei, S., Soatto, S.: Unsupervised depth completion from visual inertial odometry. IEEE Robotics and Automation Letters (2020)
12. Zhang, Y., Guo, X., Poggi, M., Zhu, Z., Huang, G., Mattoccia, S.: Completionformer: Depth completion with convolutions and vision transformers. In: CVPR (2023)