Supplemental Materials on

Monocular Depth Estimation Using Relative Depth Maps

Jae-Han Lee
Korea University
jaehanlee@mcl.korea.ac.kr

Chang-Su Kim
Korea University
changsukim@korea.ac.kr

S-1. Implementation Details

Network configuration: DenseNet-BC [23], excluding the last dense block, is used as the encoder, which consists of one convolution layer, one max pooling layer, and three pairs of dense block and transition layer. The last dense block in DenseNet-BC is employed in the ten decoders in decoder part.

Each dense block is determined with hyper-parameters, such as the number n of composite functions and the growth rate k. Table S-1 lists these hyper-parameters in this work. Since each transition layer halves the feature map resolution and channels, given an 226×226 RGB image, the encoder generates an 8×8 feature map with 1,056 channels.

Also, in each decoder, a variable number (0 to 4) of WSM blocks [20] are used to expand low resolution features to higher resolution depth maps $\mathbf{D}_n$ and $\mathbf{R}_n$. Table S-2 lists hyper-parameters of each WSM block.

Matrix completion: We use the ALS algorithm to complete the comparison matrix $\mathbf{P}_{n,n-1}$ in (8). Since ALS performs element-wise operations in each step, its complexity is proportional to the size of $\mathbf{P}_{n,n-1}$, which is 2^{4n-2}. Thus, the complexity grows 16 times for every unit increase in n.

To reduce the complexity, we develop a divide-and-conquer approach. First, we divide a depth map into small regions and construct the comparison sub-matrix for each region. Specifically, for $n \geq 5$, we divide $\mathbf{D}_n$ and $\mathbf{D}_{n-1}$ into regions of sizes 16×16 and 8×8, respectively. Then, for each pair of 16×16 and 8×8 regions, we construct the comparison sub-matrix and restore missing entries. Then, from these comparison sub-matrices, we reconstruct fine detail maps $\mathbf{F}_k$ for $n - 3 \leq k \leq n$. However, since we do not consider the comparison across sub-matrices, the coarse scale components $\mathbf{F}_k$, for $1 \leq k \leq n - 4$ cannot be reconstructed. Thus, these coarse scale components are excluded from the combination in (7).

Depth component combination: We estimate up to ten depth maps, $\mathbf{D}_n$ and $\mathbf{R}_n$ for $3 \leq n \leq 7$, each of which is decomposed into components, as listed in Table 1. Since there are multiple candidates for each component, we attempt to obtain an optimal estimate by combining them. Then, we use these optimal components to generate the final depth map via (6).

For notational simplicity, let $\mathbf{C}$ be a component (*i.e.* $\mathbf{D}_0$ or $\mathbf{F}_n$, $1 \leq n \leq 7$ in Table 1), and let $\mathbf{C}_i$ be its ith candidate. Then, we combine the candidates by

$$\log \mathbf{C} = \sum_i \mathbf{w}(i) \log \mathbf{C}_i \tag{S-1}$$

where $\mathbf{w}$ is a weight vector. We reshape each $\log \mathbf{C}_i$ into a column vector $\mathbf{a}_i$, and $\log \mathbf{C}$ into $\mathbf{g}$. Then, (S-1) can be rewritten as

$$\mathbf{g} = \mathbf{A}\mathbf{w} \tag{S-2}$$

where the ith column of $\mathbf{A}$ is $\mathbf{a}_i$. We use training images

Table S-1. Hyper-parameters of dense blocks [23] in the proposed depth estimation network in Figure 2: n is the number of composite functions, k the growth rate, $H \times W$ the feature map resolution, c_i the number of input channels, and c_o the number of output channels. All dense blocks in the ten decoders have the same hyper-parameters.

	n	k	$H \times W$	c_i	c_o
Dense E2	6	48	57×57	96	384
Dense E3	12	48	29×29	192	768
Dense E4	36	48	15×15	384	2,112
Dense D	24	48	8×8	1,056	2,208

Table S-2. Hyper-parameters of WSM blocks [20] in the decoders: c_1 is the number of channels for 1×1, 3×3 and 5×5 kernels, c_2 is the number of channels for $W \times 3$ and $3 \times H$ kernels, $H \times W$ the feature map resolution, c_i the number of input channels, and c_o the number of output channels.

	c_1	c_2	$H \times W$	c_i	c_o
WSM D-1	416	208	16×16	1,664	1,664
WSM D-2	208	104	32×32	832	832
WSM D-3	104	52	64×64	416	416
WSM D-4	52	26	128×128	208	208

to determine the weight $\mathbf{w}$. We attempt to minimize the squared error $\|\mathbf{g}^{(t)} - \mathbf{A}^{(t)}\mathbf{w}\|^2$, where $\mathbf{g}^{(t)}$ is the ground truth component for the t-th training image and $\mathbf{A}^{(t)}$ is composed of the corresponding component candidates. More specifically, we solve the constrained optimization

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_t \|\mathbf{g}^{(t)} - \mathbf{A}^{(t)}\mathbf{w}\|^2 \qquad \text{(S-3)}$$

subject to $\mathbf{w} \geq \mathbf{0}$, $w_{\max} \leq \alpha w_{\min}$ and $\mathbf{1}^{\mathrm{T}}\mathbf{w} \leq \beta$. We solve this quadratic problem using the interior point method (1). For optimizing $\mathbf{D}_0$ and $\mathbf{F}_n$, $1 \leq n \leq 3$, we set $\alpha = \beta = 1.05$. On the other hand, for $\mathbf{F}_n$, $n \geq 4$, we impose only the first constraint, since there are sufficient data and is a little risk of over-fitting.

KITTI dataset [14]: We also apply our algorithm to another dataset, KITTI, which is widely used for monocular depth estimation. It provides outdoor RGB sequences and their depths, scanned by LIDAR devices mounted on driving vehicles. It provides depth labels for sparse pixels only. We adopt the split scheme [12] to extract 697 test images from 29 scenes and 22,000 training ones from the remaining 32 scenes.

We train the encoder-decoder CNN for this dataset using the same initialization methods, optimizers, and hyperparameters as those for the NYUv2 dataset, except that we set the learning rate cycle of the shifted cosine function [22, 43] to 1 epoch due to the relative small size of the KITTI dataset.

Since the depth information is provided only for very sparse pixels, we need to generate the lower resolution depth map $\mathbf{D}_{n-1}$ in (3) in a different way. Specifically, when we compute each depth in $\mathbf{D}_{n-1}$, the geometric mean is calculated in the corresponding region of $\mathbf{D}_n$, excluding pixels without depth labels. Moreover, if all depths in the region are missing, we approximate the depths with the nearest neighbor method. However, this approximation of missing depths degrades the qualities of low resolution depth maps, such as $\mathbf{D}_n$ and $\mathbf{R}_n$ for $n = 3, 4$, during the training. Therefore, it also lowers the depth estimation performance in the test phase. This is a challenge that needs to be addressed in future work.

The width of an image in KITTI is much longer than the height. Therefore, an input image to the network is resized to 161×483. For the same reason, the depth map $\mathbf{D}_n$ at each scale is also resized, as listed in Table S-3. Even if the spatial resolution of each depth map is modified, we can derive $\mathbf{R}_n$ and $\mathbf{F}_n$ in the same manner as described in Section 3.1.

S-2. Experiments on KITTI

Comparison with the state-of-the-arts: We measure depth estimation performances for the ranges of $0 \sim 50\,\mathrm{m}$ and $0 \sim 80\,\mathrm{m}$ separately. Table S-4 compares quantitative results of the proposed algorithm with those of the conventional algorithms in [12,13,66,69], [S1–S4] for the two depth ranges. Each algorithm crops depth maps differently. Hence, for common evaluation, we adopt the Garg *et al.*'s evaluation protocol used in [69], [S1–S4].

Table S-3. The spatial resolution of $\mathbf{D}_n$ for the KITTI dataset.

	$\mathbf{D}_0$	$\mathbf{D}_1$	$\mathbf{D}_2$	$\mathbf{D}_3$	$\mathbf{D}_4$	$\mathbf{D}_5$	$\mathbf{D}_6$	$\mathbf{D}_7$
height	1	1	3	6	12	24	48	96
width	1	4	8	16	32	64	128	256

Note that the proposed algorithm provides competitive results on the KITTI dataset as well. For most metrics, the proposed algorithm yields the second best performance. Figure S-1 shows qualitative comparison results on KITTI.

Notice that the performance of the proposed algorithm on KITTI is relatively low in comparison with that on NYUv2. This is because KITTI provides depth information for very sparse pixels only. Therefore, as mentioned previously, approximate and imprecise depths are used for low-resolution depth maps, such as $\mathbf{D}_n$ and $\mathbf{R}_n$ for $n = 3, 4$, which makes the training process less reliable. Reliable training in case of sparse depth information is a challenge for future work.

Ablation study: On the KITTI dataset as well, we perform experiments using various combinations of depth maps and summarize the results in Table S-5. We make the following observations:

- When using a single ordinary depth map, a higher resolution one gives better results. In particular, the performance gap between the highest and lowest resolution ones is even bigger, as compared with NYUv2, due to the characteristics of KITTI in which training labels are provided for sparse pixels only.
- It is reaffirmed that relative depth maps are more effective in reconstructing depths than ordinary depth maps. For example, combination $(\mathbf{R}_3, \mathbf{R}_4, \mathbf{R}_5, \mathbf{R}_6, \mathbf{D}_7)$ outperforms combination $(\mathbf{D}_3, \mathbf{D}_4, \mathbf{D}_5, \mathbf{D}_6, \mathbf{D}_7)$.
- Similar to NYUv2, combining one ordinary depth map and four relative depth maps shows the best performance. Combining additional ordinary depth maps does not improve performance.

S-3. Additional Experiments on NYUv2

More qualitative comparisons: Figure S-2 shows additional comparison results with [6, 12, 13, 31, 33]. In most cases, the proposed algorithm yields a more accurate estimation result than the conventional algorithms.

Depth component analysis: In Figure S-3, we show or-

Table S-4. Performance comparison on the KITTI dataset. The best results are boldfaced, and the second best ones are underlined. Note that the algorithms marked with '*' follow the different evaluation method from our algorithm. All other algorithms including the proposed algorithm follow the evaluation method of [S1].

	Cap	The lower, the better				The higher, the better		
		RMSE (lin)	RMSE (log)	ARD	SRD	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen *et al.* [12] *	0 - 80m	7.156	0.270	0.190	1.515	69.2%	89.9%	96.7%
Zhuo *et al.* [S4]	0 - 80m	6.565	0.275	0.198	1.836	71.8%	90.1%	96.0%
Yin and Shi [69]	0 - 80m	5.737	0.232	0.153	1.328	80.2%	93.4%	97.2%
Xu *et al.* [66] *	0 - 80m	4.677	-	0.122	0.897	81.8%	95.4%	98.5%
Godard *et al.* [S2]	0 - 80m	4.935	0.206	0.114	0.898	86.1%	94.9%	97.6%
Kuznietsov *et al.* [S3]	0 - 80m	4.621	0.189	0.113	0.741	86.2%	96.0%	98.6%
Fu *et al.* [13] *	0 - 80m	**2.727**	**0.120**	**0.072**	**0.307**	**93.2%**	**98.4%**	**99.4%**
Proposed	0 - 80m	4.781	0.179	0.108	0.702	86.7%	96.4%	98.8%
Garg *et al.* [S1]	0 - 50m	5.104	0.273	0.169	1.080	74.0%	90.4%	96.2%
Zhuo *et al.* [S4]	0 - 50m	4.975	0.258	0.190	1.436	73.5%	91.5%	96.8%
Yin and Shi [69]	0 - 50m	4.348	0.218	0.147	0.936	81.0%	94.1%	97.7%
Godard *et al.* [S2]	0 - 50m	3.729	0.194	0.108	0.657	87.3%	95.4%	97.9%
Kuznietsov *et al.* [S3]	0 - 50m	3.518	0.179	0.108	0.595	87.5%	96.4%	98.8%
Fu *et al.* [13] *	0 - 50m	**2.271**	**0.116**	**0.071**	**0.268**	**93.6%**	**98.5%**	**99.5%**
Proposed	0 - 50m	3.568	0.167	0.103	0.551	88.1%	97.0%	99.1%

Table S-5. Ablation study using various combinations of depth maps on the KITTI dataset.

Used ordinary depth map					Used relative depth map				Cap	The lower, the better		The higher, the better			
$\mathbf{D}_3$	$\mathbf{D}_4$	$\mathbf{D}_5$	$\mathbf{D}_6$	$\mathbf{D}_7$	$\mathbf{R}_3$	$\mathbf{R}_4$	$\mathbf{R}_5$	$\mathbf{R}_6$		RMSE (lin)	ARD	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	ρ
√	-	-	-	-	-	-	-	-	0 - 80m	7.080	0.202	68.1%	88.2%	96.1%	0.889
-	-	√	-	-	-	-	-	-	0 - 80m	5.046	0.116	84.9%	95.9%	98.7%	0.949
-	-	-	-	√	-	-	-	-	0 - 80m	4.869	0.111	86.2%	96.2%	98.8%	0.953
√	√	√	√	√	-	-	-	-	0 - 80m	4.851	0.111	86.2%	96.3%	98.8%	0.953
-	-	-	-	√	√	√	√	√	0 - 80m	4.781	0.108	86.7%	96.4%	98.8%	0.955
√	√	√	√	√	√	√	√	√	0 - 80m	4.784	0.108	86.7%	96.4%	98.8%	0.955
√	-	-	-	-	-	-	-	-	0 - 50m	5.261	0.195	68.1%	88.2%	97.0%	0.883
-	-	√	-	-	-	-	-	-	0 - 50m	3.757	0.111	86.4%	96.6%	99.0%	0.947
-	-	-	-	√	-	-	-	-	0 - 50m	3.654	0.106	87.7%	96.9%	99.0%	0.951
√	√	√	√	√	-	-	-	-	0 - 50m	3.644	0.106	87.6%	96.9%	99.0%	0.951
-	-	-	-	√	√	√	√	√	0 - 50m	3.568	0.103	88.1%	97.0%	99.1%	0.953
√	√	√	√	√	√	√	√	√	0 - 50m	3.568	0.103	88.1%	97.0%	99.1%	0.953

dinary depth maps $\mathbf{D}_n$, relative depth maps $\mathbf{R}_n$, and final combined depth maps. Compared to an ordinary depth map of the same resolution, the relative depth map reconstructs fine details more faithfully. Thus, the proposed algorithm, which combines the two kinds of depth maps, provides higher quality depth maps than the conventional algorithms, which estimate only ordinary depth maps.

To demonstrate this, we decompose depth maps, estimated by the proposed algorithm and the conventional algorithms [13, 31], into components $\mathbf{D}_0$ and $\mathbf{F}_n$ according to (6). Then, for each fine detail map $\mathbf{F}_n$, $1 \leq n \leq 6$, we evaluate the objective qualities using four metrics: Spearman's ρ, $\delta < 1.25$, RMSE (log) and ARD. Figure S-4 shows the results. For $\mathbf{F}_n$, $n \geq 2$, the proposed algorithm outperforms the other algorithms in all metrics. Figure S-5 also compares depth components. Again, we see that the proposed algorithm restores fine details more faithfully than the conventional algorithms.

Computation times: Table S-6 lists the average computation times of the NYU v2 test data. The default mode takes 0.816s to estimate a depth map. If we adopt the light mode using four maps ($\mathbf{D}_3$, $\mathbf{R}_3$, $\mathbf{R}_4$, $\mathbf{R}_5$), it takes 0.234s only, even though its performance is still competitive in Table 4.

Table S-6. Computation times (s). 'Net' means the network computation and 'Comb' means the depth component combination.

	Net	ALS	Comb	Total
Proposed (light)	0.073	0.080	0.081	0.234
Proposed (default)	0.134	0.594	0.088	0.816

References

[S1] R. Garg, G. Carneiro, and I. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016.

[S2] C. Godard, O. M. Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[S3] Y. Kuznietsov, J. Stückler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017.

[S4] T. Zhuo, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

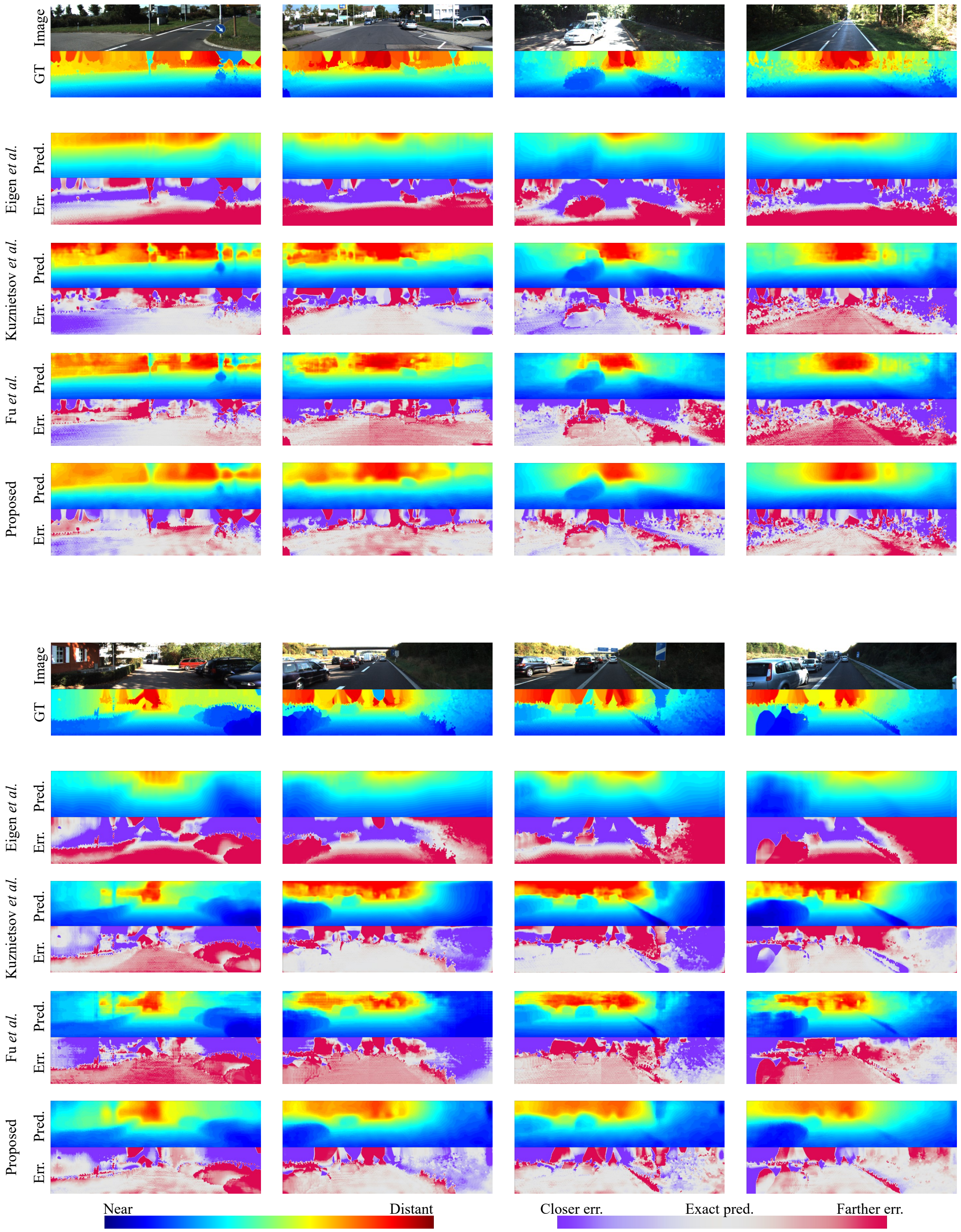

Figure S-1. Qualitative comparison of Eigen *et al*. [12], Kuznietsov *et al*. [S3], Fu *et al*. [13], and the proposed algorithm on the KITTI dataset. Predicted depth maps (Pred) and error maps (Err) are provided for easier comparison.

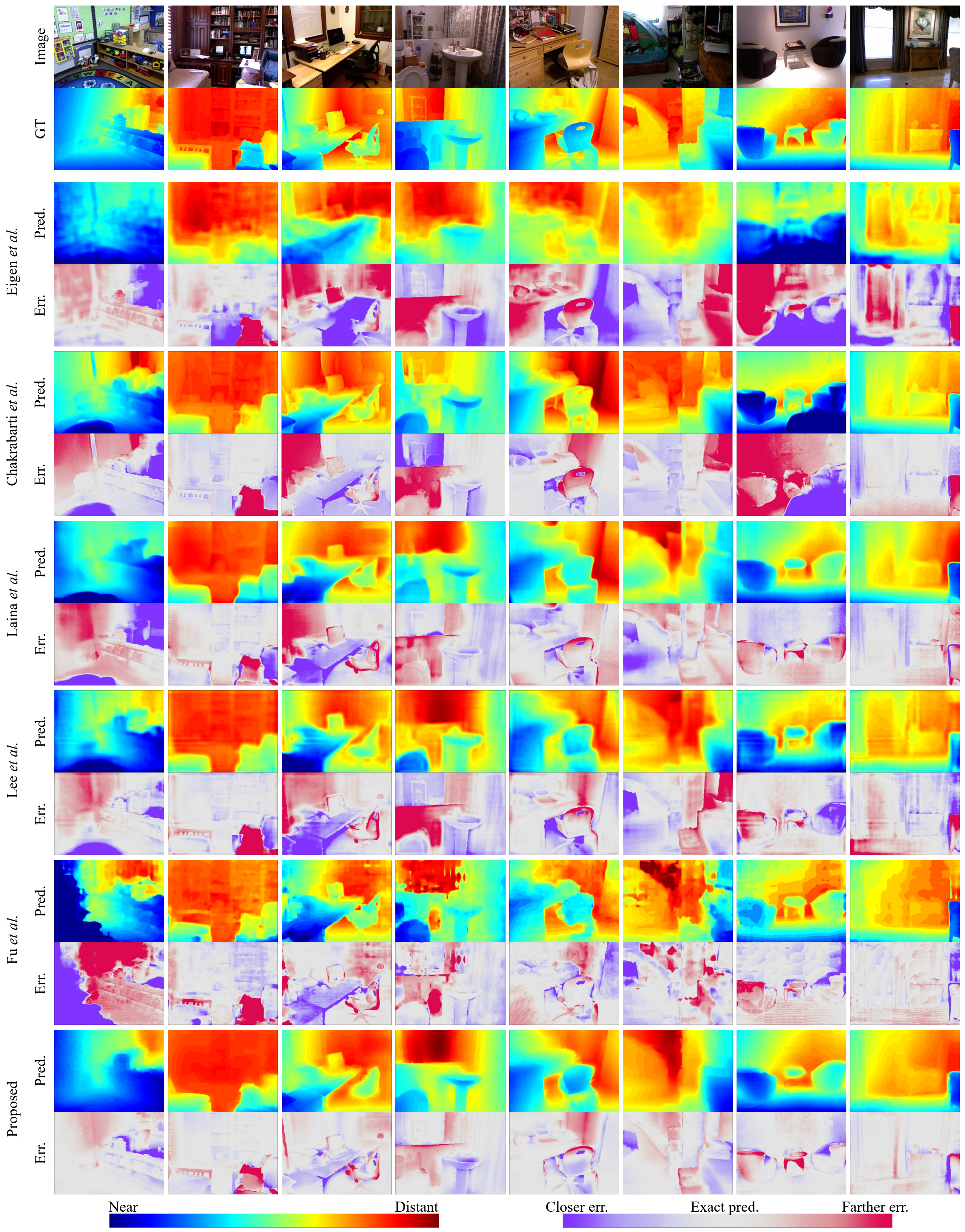

Figure S-2. Qualitative comparison of Eigen *et al*. [12], Chakrabarti *et al*. [6] Laina *et al*. [31], Lee *et al*. [33], Fu *et al*. [13], and the proposed algorithm on NYUv2 images. Predicted depth maps (Pred) and error maps (Err) are provided for easier comparison.

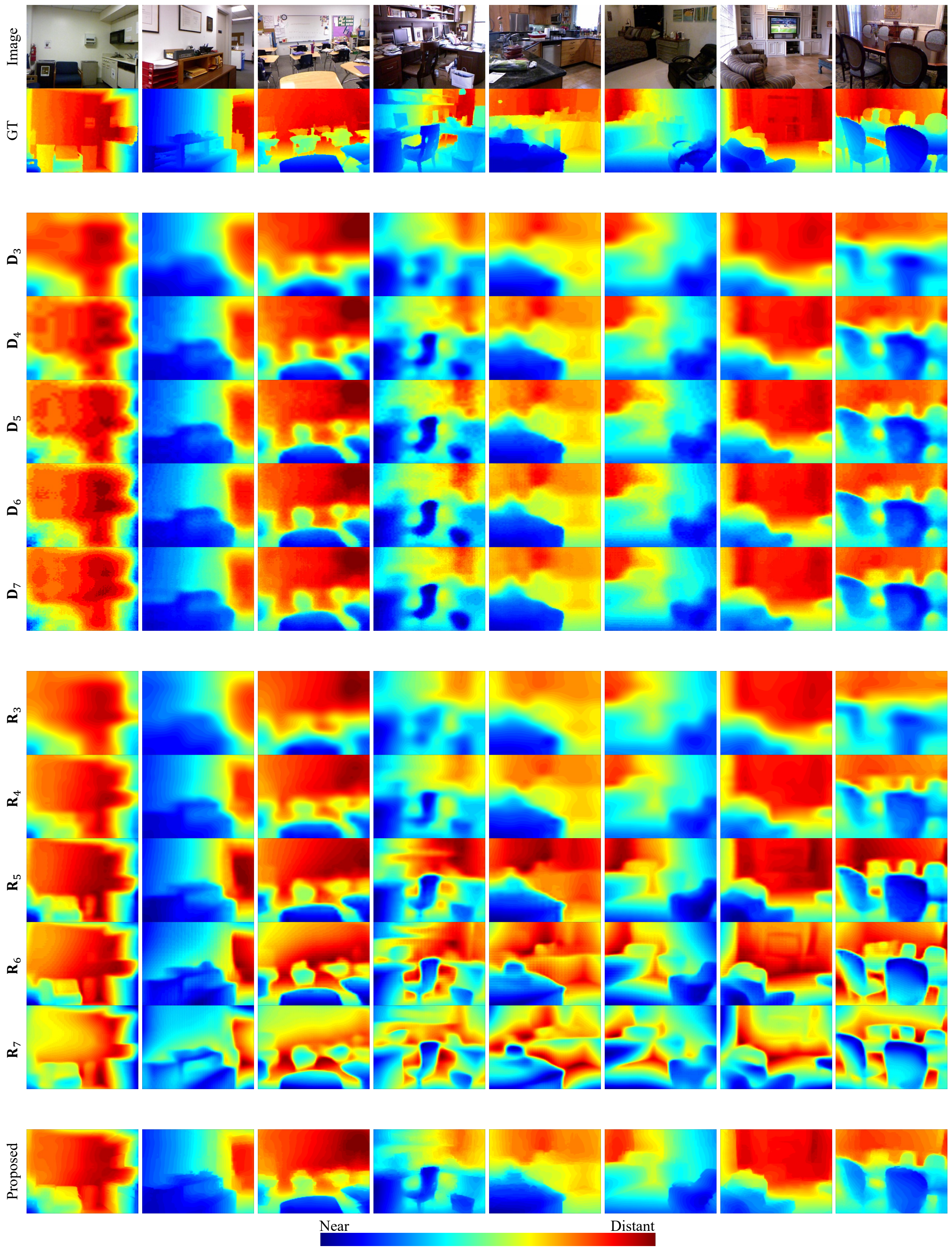

Figure S-3. Individual depth maps $\mathbf{D}_n$ and $\mathbf{R}_n$, $3 \leq n \leq 7$, and final combined depth maps using $(\mathbf{D}_3, \mathbf{R}_3, \mathbf{R}_4, \mathbf{R}_5, \mathbf{R}_6)$ in the default mode.

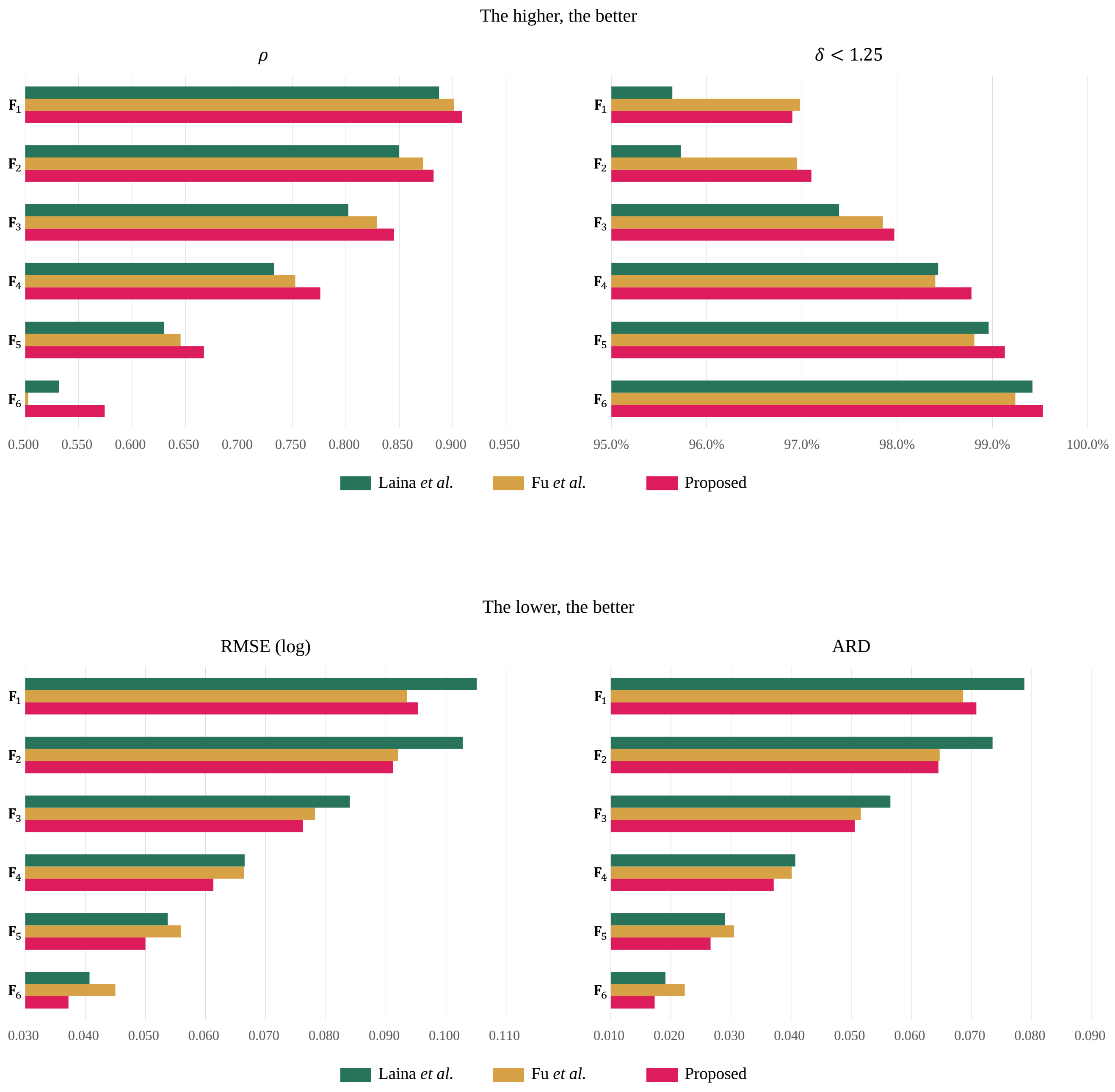

Figure S-4. Comparison of fine detail maps $\mathbf{F}_n, 1 \leq n \leq 6$, which are decomposed from depth maps, estimated by Laina *et al*. [31], Fu *et al*. [13], and the proposed algorithm, respectively.

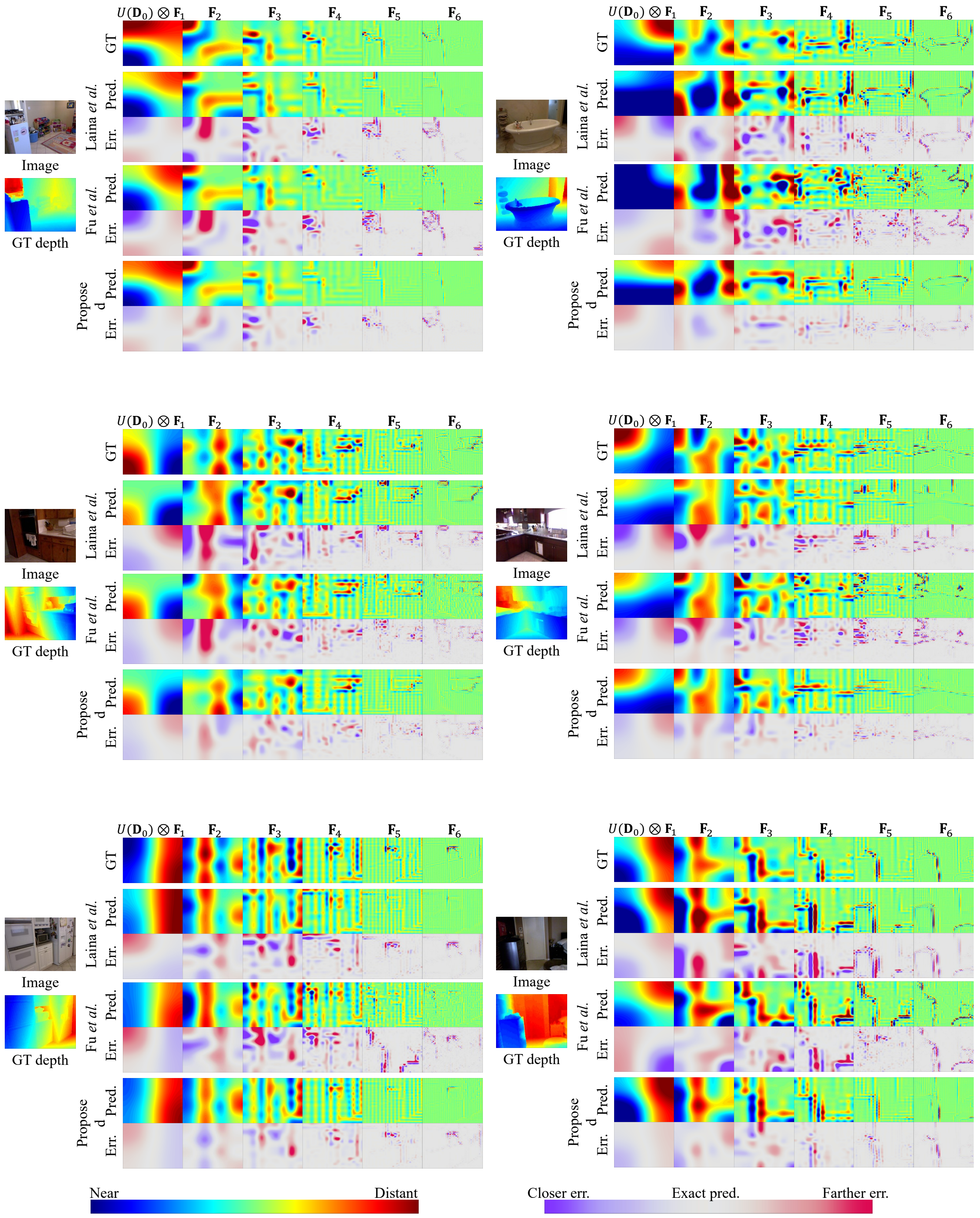

Figure S-5. Qualitative comparison of depth components $(\mathbf{D}_0, \mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3, \mathbf{F}_4, \mathbf{F}_5, \mathbf{F}_6)$, produced by Laina *et al.* [31], Fu *et al.* [13], and the proposed algorithm. Predicted depth maps (Pred) and error maps (Err) are provided for easier comparison.