Supplementary Material for Connecting the Dots: Learning Representations for Active Monocular Depth Estimation

Gernot Riegler[1,*] Yiyi Liao[2,*] Simon Donne[2] Vladlen Koltun[1] Andreas Geiger[2]
[1]Intel Intelligent Systems Lab [2]Autonomous Vision Group, MPI-IS / University of Tübingen
{firstname.lastname}@intel.com {firstname.lastname}@tue.mpg.de

Abstract

In this supplementary document, we first present details on the network architectures of our disparity decoder in Section 1. Section 2 discusses the runtime of our method and Section 3 provides additional details on the edge decoder used in our work. In Section 4 we give additional ablation studies for our method and the used baselines. Finally, we show additional results in Section 5.

1. Disparity Decoder

The disparity decoder is based on the DispNetS architecture presented in [5]. Table 1 shows the detailed disparity decoder architecture, where k denotes kernel size, s stride and p padding. C_{in} and C_{out} are the number of input and output channels. $up(\cdot)$ denotes bilinear up-sampling. We use a ReLU layer after each convolutional layer and a scaled sigmoid after each prediction layer. We do not show them in the table for clarity.

2. Runtime

We evaluated the individual parts of the network for multiple forward passes in Table 2. Note that the network evaluation (edge decoder and disparity decoder) is the most time consuming part. The geometric loss is only slightly slower than the photometric loss and adds no drastic overhead in network training.

3. Edge Decoder

We exploit a shallow U-Net architecture for our edge decoder. It is also based on the DispNet architecture [5], but with fewer layers. Table 3 shows the detailed edge decoder architecture, with k denoting kernel size, s stride, and p padding. C_{in} and C_{out} are the number of input and output channels. $up(\cdot)$ denotes bilinear up-sampling. There is a ReLU layer after each convolutional layer which we do not show in the table for clarity.

As mentioned in the main paper, we train an edge decoder to predict $\mathbf{A}' = \text{LCN}_\epsilon(|\nabla \mathbf{A}|)$, the local contrast normalized gradient magnitude of the ambient image $\mathbf{A}$. While the edge decoder requires the ambient image $\mathbf{A}$ as supervision, this is easily obtained in practice (recording the scene with the laser projector turned on and off). In particular, our shallow U-Net generalizes from few training samples: In the experiment, we assume that we have ambient images for $1,024$ short sequences among the full rendered training dataset of $8,192$ sequences.

Fig. 1 shows an example of an input image $\mathbf{I}$ from the test dataset, the edge image $\mathbf{A}'$ and our prediction. We observe that the edge decoder clearly separates edge from non-edge regions.

4. Additional Ablation Studies

In this section we present additional ablation studies for our method and the utilized baselines. Section 4.1 demonstrates the correlation between the photometric loss and some of the metrics that we evaluated on. In Section 4.2 we show an

[*] Joint first authors with equal contribution.

layer name	$k \times k, p, s$	C_{in}	C_{out}	input name	output name
conv1	$7 \times 7, 3, 2$	2	32	-	-
	$7 \times 7, 3, 1$	32	32	-	out_conv1
conv2	$5 \times 5, 2, 2$	32	64	out_conv1	-
	$5 \times 5, 2, 1$	64	64	-	out_conv2
conv3	$3 \times 3, 1, 2$	64	128	out_conv2	-
	$3 \times 3, 1, 1$	128	128	-	out_conv3
conv4	$3 \times 3, 1, 2$	128	256	out_conv3	-
	$3 \times 3, 1, 1$	256	256	-	out_conv4
conv5	$3 \times 3, 1, 2$	256	512	out_conv4	-
	$3 \times 3, 1, 1$	512	512	-	out_conv5
conv6	$3 \times 3, 1, 2$	512	512	out_conv5	-
	$3 \times 3, 1, 1$	512	512	-	out_conv6
conv7	$3 \times 3, 1, 2$	512	512	out_conv6	-
	$3 \times 3, 1, 1$	512	512	-	out_conv7
upconv7	$3 \times 3, 1, 2$	512	512	out_conv7	out_upconv7
concat	-	-	-	out_upconv7, out_conv6	concat7
iconv7	$3 \times 3, 1, 1$	1024	512	concat7	out_iconv7
upconv6	$3 \times 3, 1, 2$	512	512	out_conv6	out_upconv6
concat	-	-	-	out_upconv6, out_conv5	concat6
iconv6	$3 \times 3, 1, 1$	1024	512	concat6	out_iconv6
upconv5	$3 \times 3, 1, 2$	512	256	out_conv5	out_upconv5
concat	-	-	-	out_upconv5, out_conv4	concat5
iconv5	$3 \times 3, 1, 1$	512	256	concat5	out_iconv5
upconv4	$3 \times 3, 1, 2$	256	128	out_conv4	out_upconv4
concat	-	-	-	out_upconv4, out_conv3	concat4
iconv4	$3 \times 3, 1, 1$	256	128	concat4	out_iconv4
predict_disp4	$3 \times 3, 1, 1$	128	1	out_iconv4	disp4
upconv3	$3 \times 3, 1, 2$	128	64	out_conv3	out_upconv3
concat	-	-	-	out_upconv3, out_conv2, up(disp4)	concat3
iconv3	$3 \times 3, 1, 1$	129	64	concat3	out_iconv3
predict_disp3	$3 \times 3, 1, 1$	64	1	out_iconv3	disp3
upconv2	$3 \times 3, 1, 2$	64	32	out_iconv3	out_upconv2
concat	-	-	-	out_upconv2, out_conv1, up(disp3)	concat2
iconv2	$3 \times 3, 1, 1$	65	32	concat2	out_iconv2
predict_disp2	$3 \times 3, 1, 1$	32	1	out_iconv2	disp2
upconv1	$3 \times 3, 1, 2$	32	16	out_iconv2	out_upconv1
concat	-	-	-	out_upconv1, out_conv1, up(disp2)	concat1
iconv1	$3 \times 3, 1, 1$	33	16	concat1	out_iconv1
predict_disp1	$3 \times 3, 1, 1$	16	1	out_iconv1	disp1

Table 1: **Architecture of Disparity Decoder.** k denotes kernel size, s stride and p padding. C_{in} and C_{out} are the number of input and output channels

LCN	Edge Dec.	Disp. Dec.	$\mathcal{L}_P$	$\mathcal{L}_D$	$\mathcal{L}_G$
3.19	15.00	21.51	4.58	1.33	6.85

Table 2: **Timings [ms].** Average of single sample forward passes.

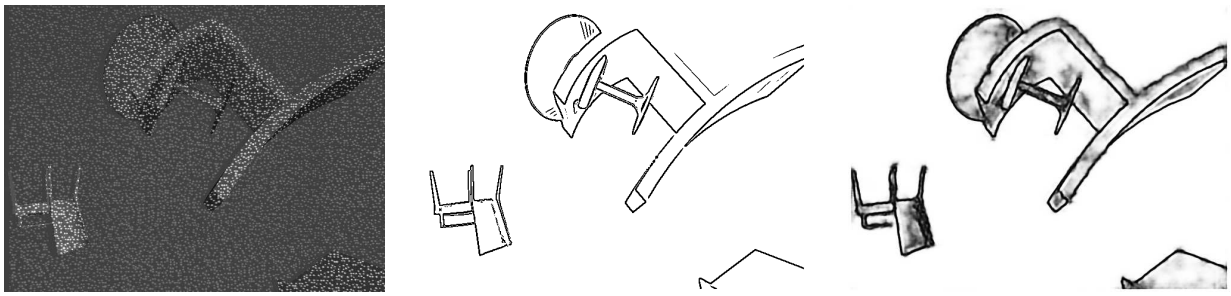
ablation study of the disparity decoder. In Section 4.3 we evaluate our method and the baselines when applied with smooth post-processing. Finally, Section 4.4 provides hyper-parameter tuning experiments and the influence of post-processing on the HyperDepth [2] baseline used in our evaluations.

4.1. Correlation of Photometric Loss

Note that our self-supervised approach does not have access to ground truth geometry. We thus select the network parameters from all training epochs by minimizing the average photometric error on a validation set with 512 samples. In this section, we demonstrate that the photometric loss is well correlated to our evaluation metrics and is thus suitable as a proxy metric. Fig. 2 illustrates the photometric loss and $o(t)$ when training our model using only the photometric loss $\mathcal{L}_P$. Each dot represents the photometric loss (x-axis) and the corresponding metric (y-axis) at one particular epoch. We fit a line to this data using linear regression, and show the correlation coefficient above each sub-figure (here 1 represents maximal positive

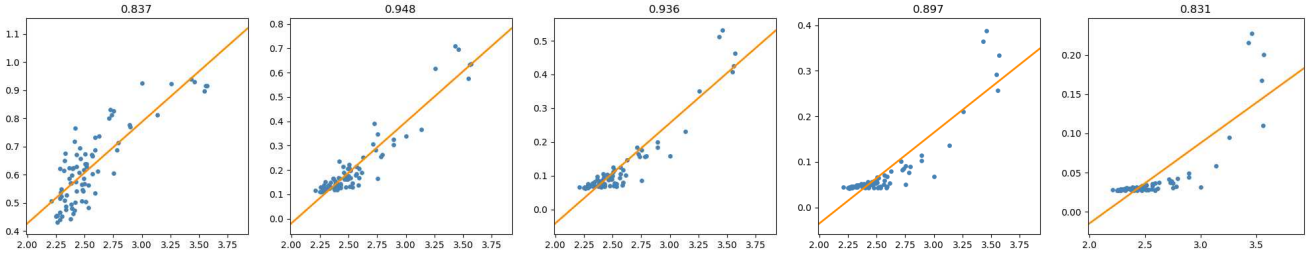
layer name	$k \times k, p, s$	C_{in}	C_{out}	input name	output name
conv1	$7 \times 7, 3, 2$	2	32	-	-
	$7 \times 7, 3, 1$	32	32	-	out_conv1
conv2	$5 \times 5, 2, 2$	32	64	out_conv1	-
	$5 \times 5, 2, 1$	64	64	-	out_conv2
conv3	$3 \times 3, 1, 2$	64	128	out_conv2	-
	$3 \times 3, 1, 1$	128	128	-	out_conv3
upconv3	$3 \times 3, 1, 2$	128	64	out_conv3	out_upconv3
concat	-	-	-	out_upconv3, out_conv2	concat3
iconv3	$3 \times 3, 1, 1$	128	64	concat3	out_iconv3
predict_edge3	$3 \times 3, 1, 1$	64	1	out_iconv3	edge3
upconv2	$3 \times 3, 1, 2$	64	32	out_iconv3	out_upconv2
concat	-	-	-	out_upconv2, out_conv1, up(edge3)	concat2
iconv2	$3 \times 3, 1, 1$	65	32	concat2	out_iconv2
predict_edge2	$3 \times 3, 1, 1$	32	1	out_iconv2	edge2
upconv1	$3 \times 3, 1, 2$	32	16	out_iconv2	out_upconv1
concat	-	-	-	out_upconv1, out_conv1, up(edge2)	concat1
iconv1	$3 \times 3, 1, 1$	33	16	concat1	out_iconv1
predict_edge1	$3 \times 3, 1, 1$	16	1	out_iconv1	edge1

Table 3: **Architecture of Edge Decoder.** k denotes kernel size, s stride and p padding. C_{in} and C_{out} are the number of input and output channels, respectively.



(a) Input $\mathbf{I}$ (b) Ground truth $\mathbf{A}'$ (c) Prediction

Figure 1: **Qualitative Results of Edge Decoder.**



(a) $o(0.1)$ (b) $o(0.5)$ (c) $o(1)$ (d) $o(2)$ (e) $o(5)$

Figure 2: **Correlation of Photometric Loss and Evaluation Metrics.** For each threshold t, we show the photometric loss (x-axis) wrt. the corresponding metric (y-axis) across epochs as data points. We show the correlation coefficient above each figure, where 1 denotes maximal positive linear correlation.

linear correlation and 0 denotes no linear correlation). Fig. 2 suggests that the photometric loss is highly correlated with $o(t)$ across different thresholds. Therefore, it is a viable criterion for model selection.

depth	o(0.1)	o(0.5)	o(1)	o(2)	o(5)
1	0.9231	0.6717	0.5303	0.4071	0.1989
2	0.8935	0.5768	0.4477	0.3468	0.1684
3	0.5646	0.2742	0.2299	0.1976	0.1502
4	0.5196	0.1999	0.1393	0.1064	0.0804
5	**0.3636**	0.1179	0.0663	0.0400	0.0224
6	0.4151	0.1207	0.0645	0.0361	0.0169
7	0.3745	**0.1084**	**0.0575**	**0.0320**	**0.0154**

Table 4: **Ablation of U-Net architecture.** A higher depth value indicates a larger receptive field of the network, with depth $= 7$ being the architecture presented in Table 1.

	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$	$o_u(1)$	$o_u(5)$
Block Matching	7.84	7.20	7.06	6.83	4.44	4.23
Block Matching + MF	6.78	6.15	5.98	5.71	3.57	3.32
FastMRF	12.07	8.36	6.71	5.14	5.25	3.57
FastMRF + MF	12.00	8.34	6.69	5.12	5.23	3.57
HyperDepth	15.01	12.63	11.83	11.49	7.39	6.73
HyperDepth + MF	9.09	7.41	6.68	6.20	3.97	3.31
Ours	6.77	3.88	2.57	1.63	1.75	0.70
Ours + MF	**6.28**	**3.62**	**2.46**	**1.58**	**1.64**	**0.69**

Table 5: **Quantitative Results on Synthetic Data with Median Filtering.**

4.2. Ablation of Disparity Decoder

In the following evaluation we show that a large receptive field as present in our disparity decoder (Table 1) is needed for accurate disparity estimation. For this evaluation we consecutively remove lower resolution parts of the network. We train the network on a 100 rows crop of our train dataset and evaluate on the same 100 rows of the test dataset. The results are summarized in Table 4. We can observe that the network performance gradually improves by adding layers at lower resolutions that capture a larger receptive field. We also tried to train the network with CoordConvs [4] and BatchNorm [3], but did not observe any improvements on the test metrics.

4.3. Smooth Post-processing

We observe that some of our baselines produces a random noise in the prediction, where the results might be improved with a smooth post-processing. For a fair comparison, we apply 5×5 median filtering to all methods in Tab. 3 of the main paper. The results are shown in Table 5. Note that the smooth post-processing improves the performance of all compared methods, while our method still performs the best.

4.4. HyperDepth Hyper-Parameters

The most related work to our method is the random forest based HyperDepth [2]. Unfortunately, there is no implementation available and therefore, we had to implement this method ourselves. We tried to replicate the method as closely as possible based on the original paper and also communicated with the authors regarding the details. For a fair comparison, we cross-validated the hyper-parameters on a validation set. In Table 6 we show results varying the total tree depth, the number of random samples for split node optimization and the tree depth at which we switch from pixel to sub-pixel accuracy. Note that in our experiments, we obtained better results with deeper trees.

Another set of hyper-parameters involve the post-processing in HyperDepth. In Fig. 3 we show quantitatively and qualitatively results for different settings. Note how with different hyper-parameters we can trade accuracy for completeness. In the evaluation in our main paper we used the hyper-parameters that lead to the smallest harmonic mean of accuracy and completeness.

	$o(0.1)$	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
depth=12, samples=1024, switch=6	0.7821	0.3123	0.2212	0.2017	0.1972
depth=12, samples=1024, switch=8	0.7817	0.3156	0.2256	0.2063	0.2015
depth=12, samples=1024, switch=10	0.7801	0.3130	0.2252	0.2061	0.2018
depth=12, samples=4096, switch=6	0.7750	0.3186	0.2350	0.2158	0.2113
depth=12, samples=4096, switch=8	0.7692	0.3028	0.2215	0.2042	0.2002
depth=12, samples=4096, switch=10	0.7694	0.3056	0.2290	0.2133	0.2094
depth=12, samples=16384, switch=6	0.7694	0.3112	0.2361	0.2201	0.2160
depth=12, samples=16384, switch=8	0.7674	0.3177	0.2433	0.2281	0.2237
depth=12, samples=16384, switch=10	0.7670	0.3152	0.2422	0.2273	0.2233
depth=14, samples=1024, switch=8	0.7411	0.2685	0.1991	0.1843	0.1802
depth=14, samples=1024, switch=10	0.7301	0.2567	0.1915	0.1780	0.1741
depth=14, samples=1024, switch=12	0.7423	0.2627	0.1943	0.1790	0.1750
depth=14, samples=4096, switch=8	0.7299	0.2570	0.1921	0.1783	0.1748
depth=14, samples=4096, switch=10	0.7264	0.2546	0.1923	0.1794	0.1760
depth=14, samples=4096, switch=12	0.7309	0.2571	0.1956	0.1830	0.1797
depth=14, samples=16384, switch=8	0.7275	0.2615	0.1987	0.1850	0.1813
depth=14, samples=16384, switch=10	0.7278	0.2691	0.2088	0.1952	0.1920
depth=14, samples=16384, switch=12	0.7349	0.2746	0.2134	0.1997	0.1961
depth=16, samples=1024, switch=10	0.7287	0.2519	0.1887	0.1750	0.1712
depth=16, samples=1024, switch=12	0.7275	0.2490	0.1852	0.1711	0.1674
depth=16, samples=1024, switch=14	0.7311	0.2448	0.1775	0.1632	0.1596
depth=16, samples=4096, switch=10	**0.7078**	**0.2269**	0.1704	0.1588	0.1554
depth=16, samples=4096, switch=12	0.7124	0.2293	**0.1703**	**0.1579**	**0.1548**
depth=16, samples=4096, switch=14	0.7197	0.2319	0.1716	0.1586	0.1553
depth=16, samples=16384, switch=10	0.7083	0.2346	0.1784	0.1665	0.1630
depth=16, samples=16384, switch=12	0.7116	0.2360	0.1810	0.1690	0.1656
depth=16, samples=16384, switch=14	0.7147	0.2342	0.1768	0.1644	0.1611

Table 6: **Hyper-Parameter Tuning for HyperDepth [2].** We train a random forest with four trees on a single row of the synthetic dataset. *depth* denotes the maximal tree depth, *samples* is the maximal number of training instances sampled to optimize a given split node, and *switch* is the tree depth where we switch from integer accuracy to sub-pixel accuracy.

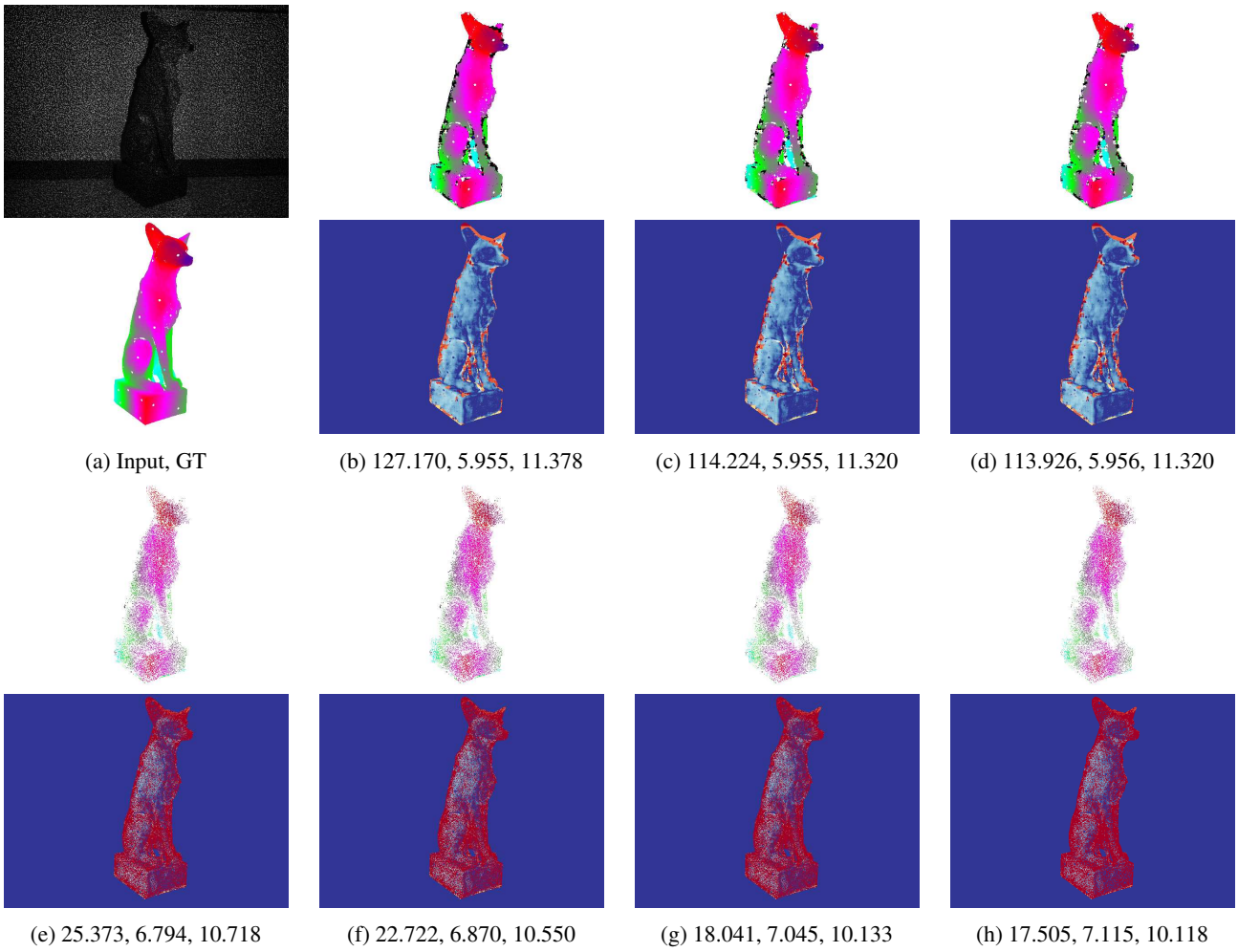
5. Additional Results

5.1. Rendered Data

In this section we show additional qualitative results on our synthetic dataset and the dataset provided by [1]. Fig. 4 depicts additional qualitative results as depth maps on our synthetic dataset for our method, block matching, FastMRF [1], and HyperDepth [2]. We also show results as 3D point-clouds from Fig. 5 to Fig. 7. For each method we show two point clouds from different perspectives. Odd rows show the point cloud from the estimated depth map, with green indicating accurate predictions, yellow are points with a distance of $1cm$ distance to the closest 3D ground-truth point, and red points have a distance $> 2cm$ to the nearest 3D ground-truth point. Even rows depict the ground-truth point cloud with the same color coding, but indicating the distance to the closest estimated 3D point. Hence, odd rows show the accuracy of a given method, whereas even rows depict their completeness. Note that we don't apply post-processing to HyperDepth on our synthetic data, as the evaluation metrics – percentage of outliers remain the same regardless of the post-processing.

5.2. Real Data

Fig. 8 shows additional depth map results on the dataset provided by [1]. We compare our method to the same set of baselines as in the previous experiment. In addition we show results as 3D point-clouds in Fig. 9 for the model *Angel*, in Fig. 10 for the model *Arch*, and in Fig. 11 for the model *Gargoyle*. For each method we show two point clouds from different perspectives. Odd rows show the point cloud from the estimated depth map, with green indicating accurate predictions, yellow are points with a distance of $5mm$ distance to the closest 3D ground-truth point, and red points have a distance



(a) Input, GT (b) 127.170, 5.955, 11.378 (c) 114.224, 5.955, 11.320 (d) 113.926, 5.956, 11.320

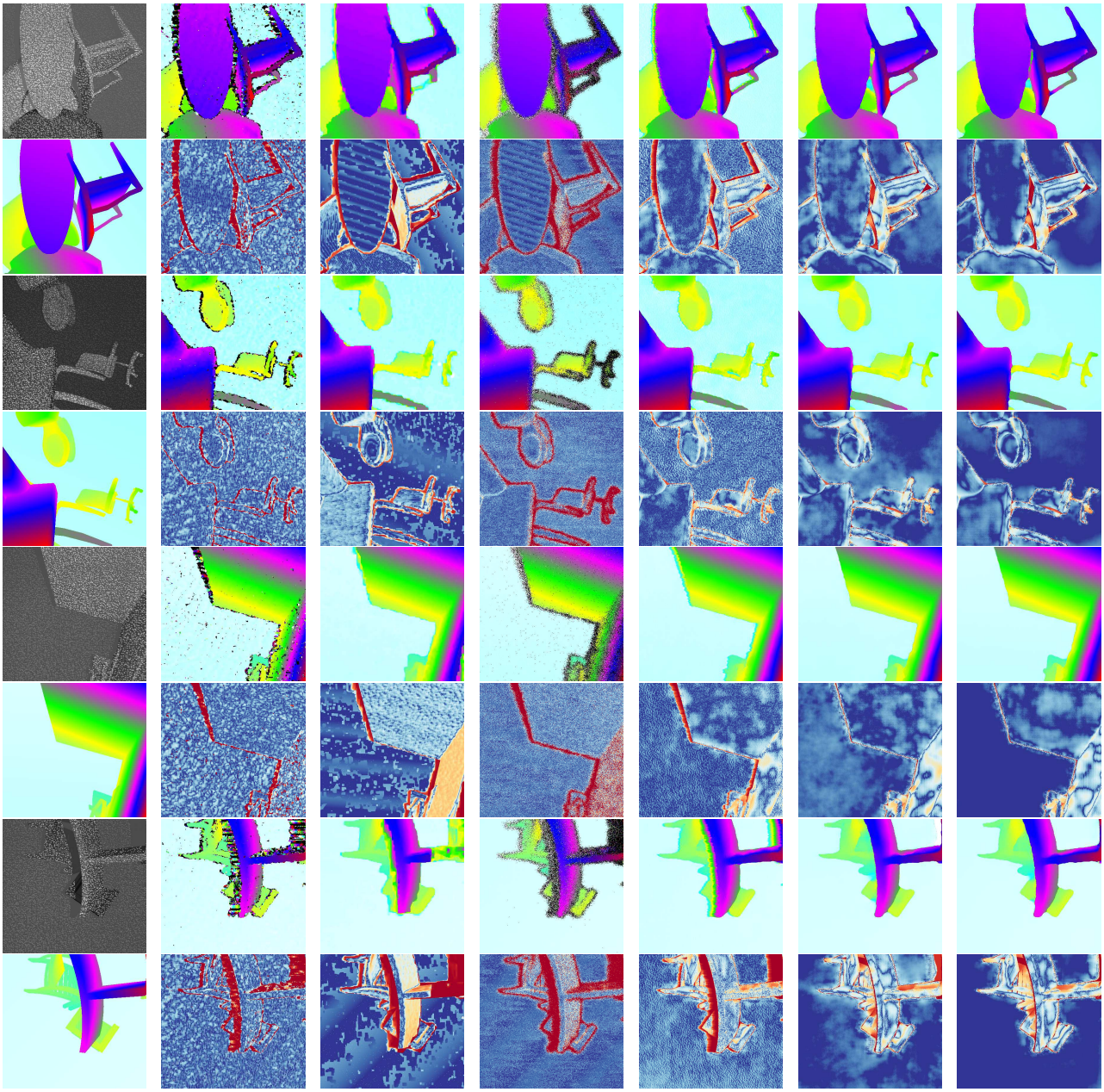
(e) 25.373, 6.794, 10.718 (f) 22.722, 6.870, 10.550 (g) 18.041, 7.045, 10.133 (h) 17.505, 7.115, 10.118

Figure 3: **Influence of HyperDepth Post-Processing.** (a) Input IR image and projected ground-truth model. (b) No post-processing, raw random forest output. (c) Mask out disparity values $\notin [0, d_{\max}]$. (d) Additionally mask out disparity values with likelihood from forest < 0.1. (e) Additionally mask out disparity values, where best and second best predicted disparity difference is $>$ 10pixels. (f) Additionally mask out disparity values, where best and second best predicted disparity difference is $>$ 5pixels. (g) Additionally mask out disparity values, where best and second best predicted disparity difference is $>$ 2pixels. (h) Additionally mask out disparity values, where best and second best predicted disparity difference is $>$ 1pixels. The numbers in the sub-caption are accuracy, completeness and harmonic mean of those two numbers in mm.

$> 1cm$ to the nearest 3D ground-truth point. Even rows depict the ground-truth point cloud with the same color coding, but indicating the distance to the closest estimated 3D point. Hence, odd rows show the accuracy of a given method, whereas even rows depict their completeness.

5.3. Real Data in Complex Real-World Scenarios

In addition to the dataset provided by [1], we further trained and evaluated our network in more complex real-world scenarios with IR images collected by a Microsoft Kinect v1 ($3,191$ for training and 623 for testing). Fig. 12 shows qualitative 3D results on the test set with a human in motion and an indoor scene respectively, demonstrating that our method generalizes well to complex real-world scenarios.



(a) Input, GT (b) Block M. (c) FastMRF [1] (d) HyperD. [2] (e) Ours $\mathcal{L}_P$ (f) Ours $+\mathcal{L}_D$ (g) Ours $+\mathcal{L}_G$

Figure 4: **Additional Qualitative Results on Synthetic Data.**

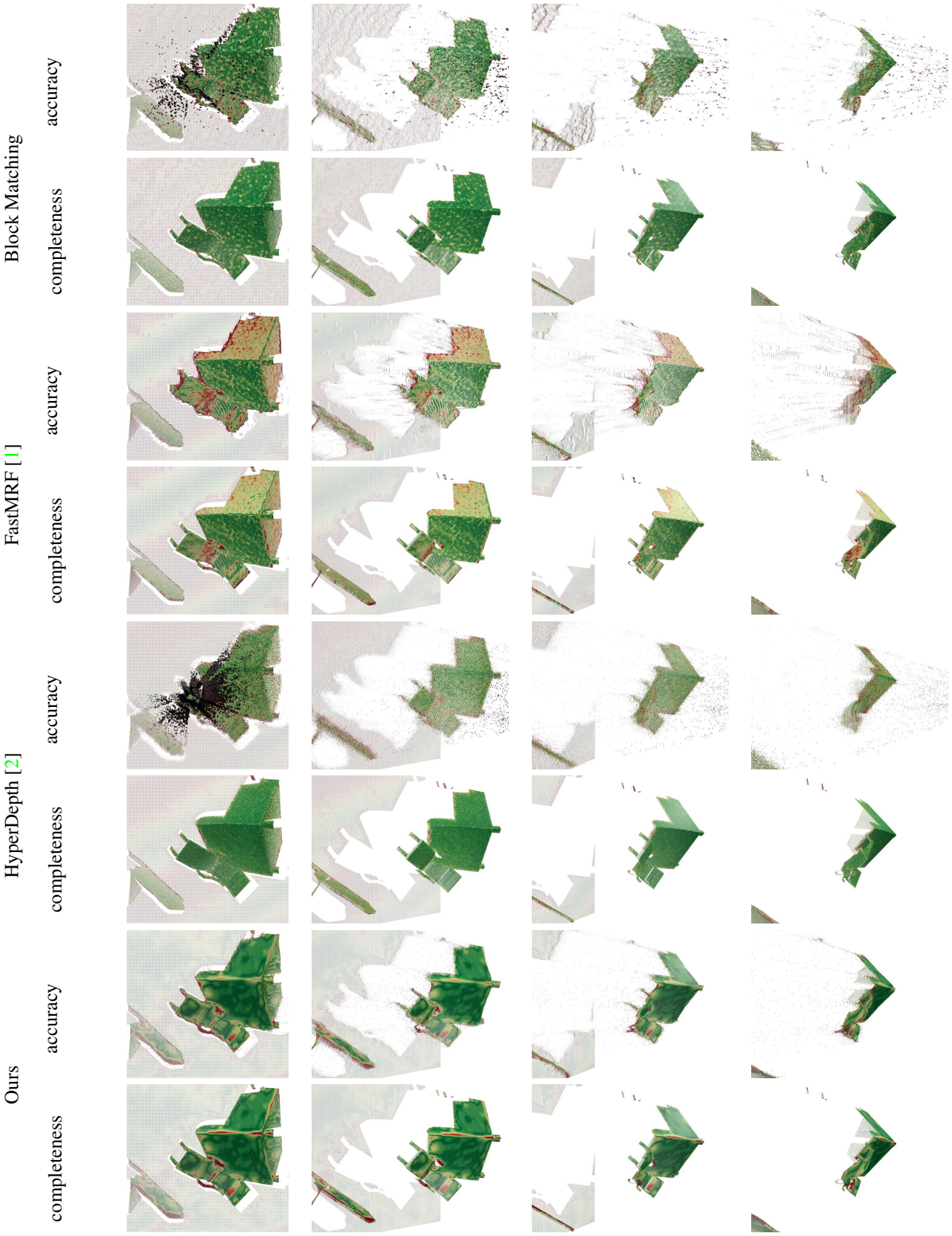


Figure 5: **Additional Qualitative 3D Results on Synthetic Data.** Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0cm$, over yellow $= 1cm$, to red $\geq 2cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point.

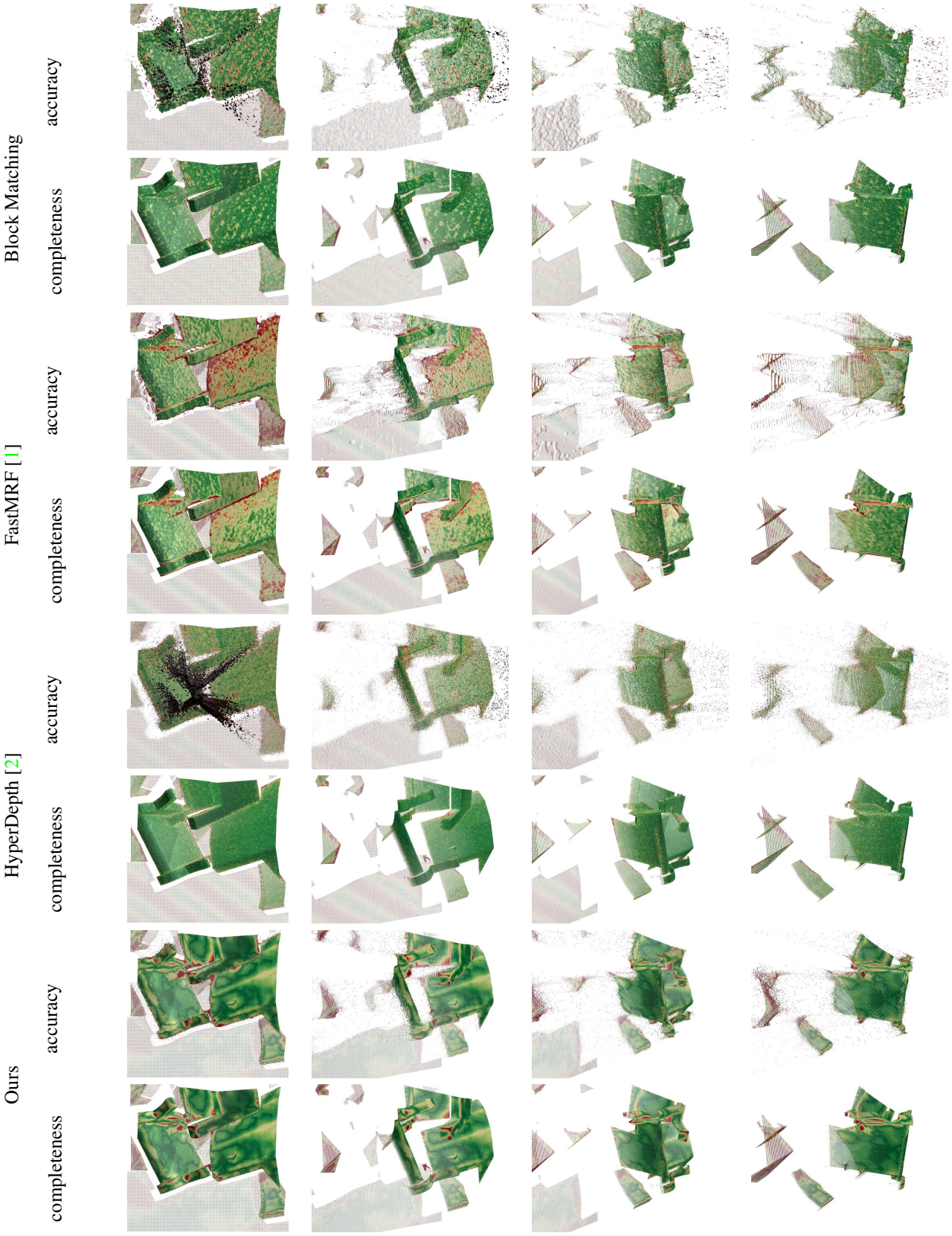
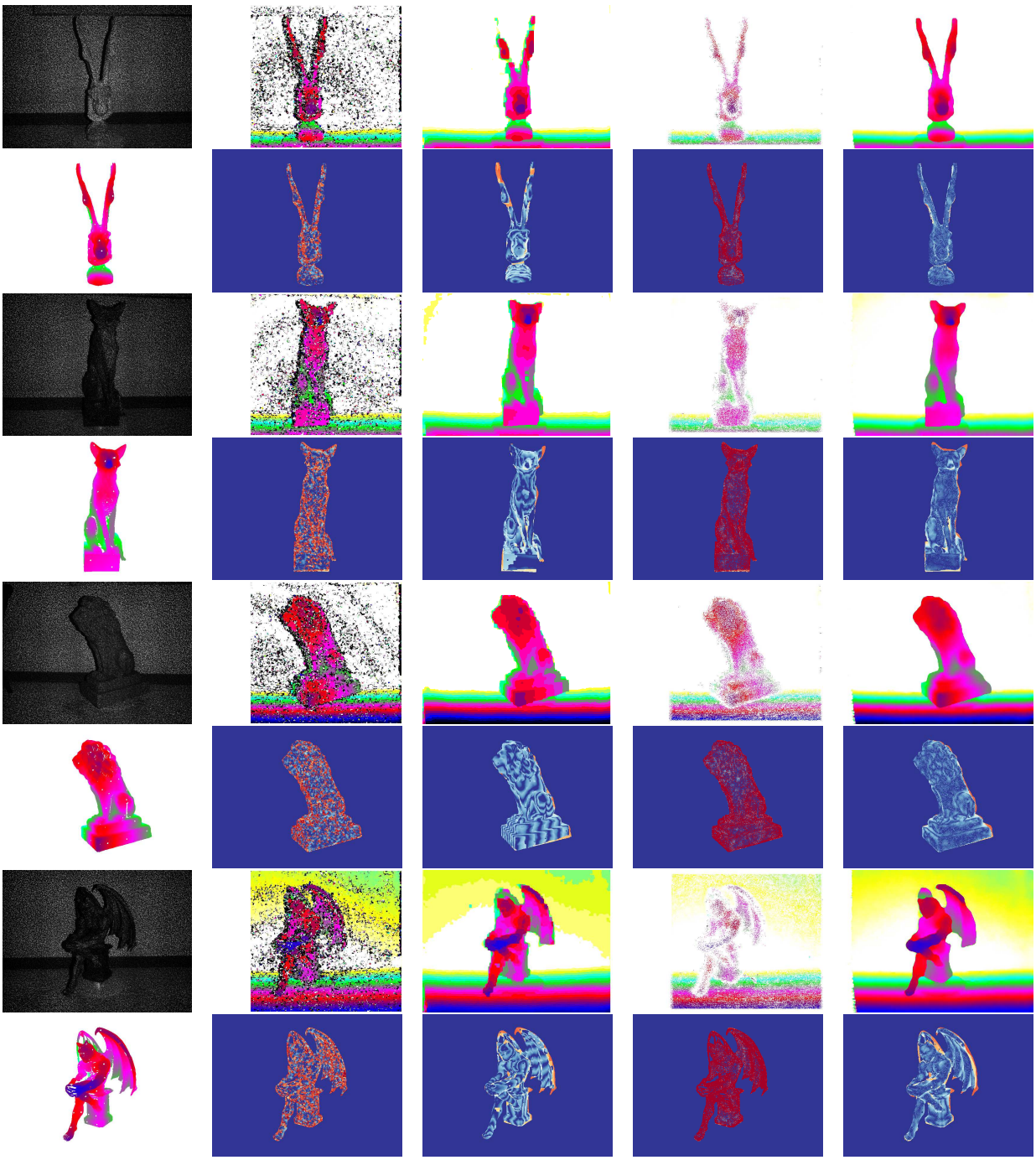


Figure 6: **Additional Qualitative 3D Results on Synthetic Data.** Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0cm$, over yellow $= 1cm$, to red $\geq 2cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point.

Figure 7: **Additional Qualitative 3D Results on Synthetic Data.** Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0cm$, over yellow $= 1cm$, to red $\geq 2cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point.



(a) Input, GT (b) Block M. (c) FastMRF [1] (d) HyperD. [2] (e) Ours

Figure 8: **Additional Qualitative Results on Real Data.**

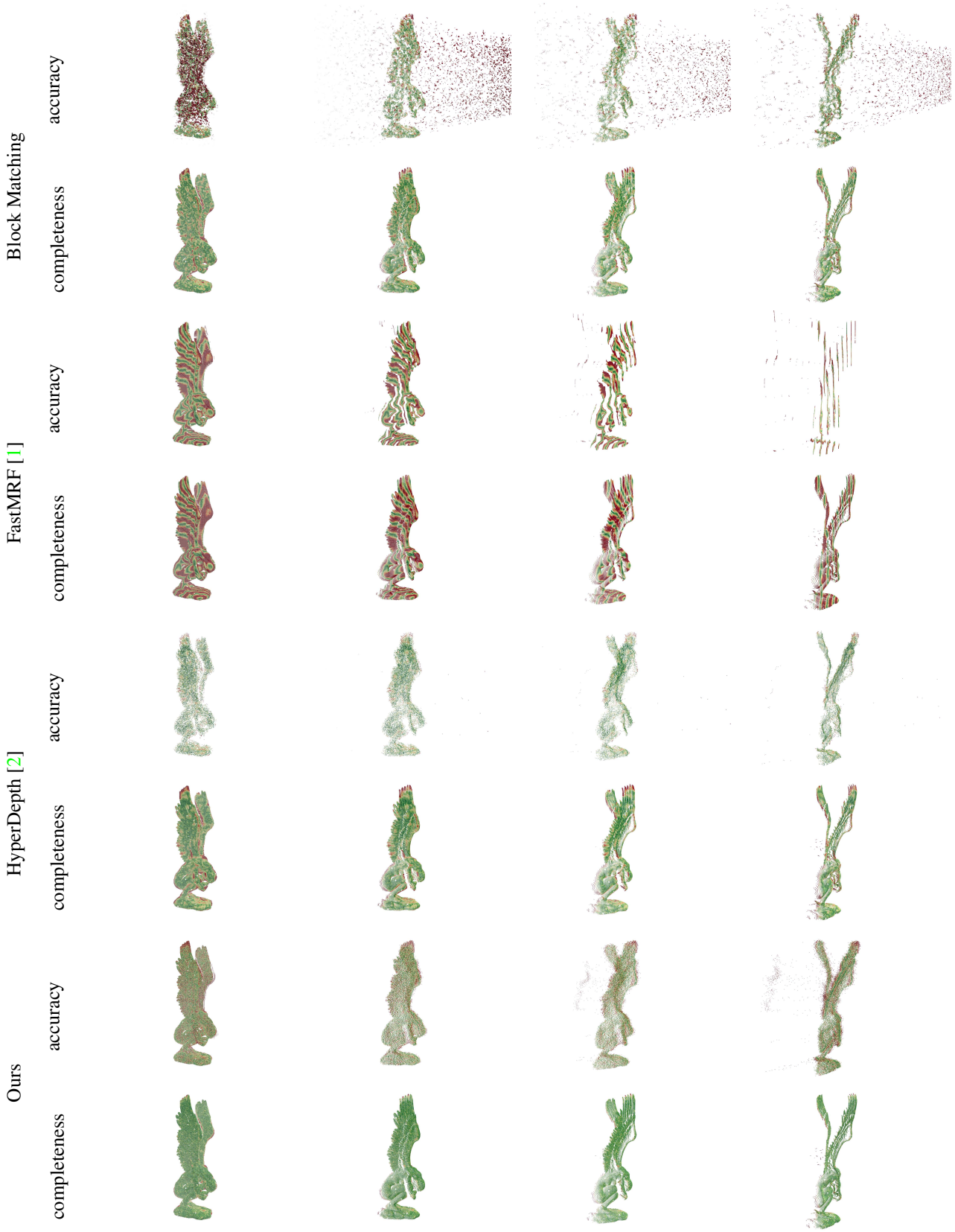


Figure 9: **Additional Qualitative 3D Results on Real Data.** *Angel*. Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0mm$, over yellow $= 5mm$, to red $\geq 1cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point.

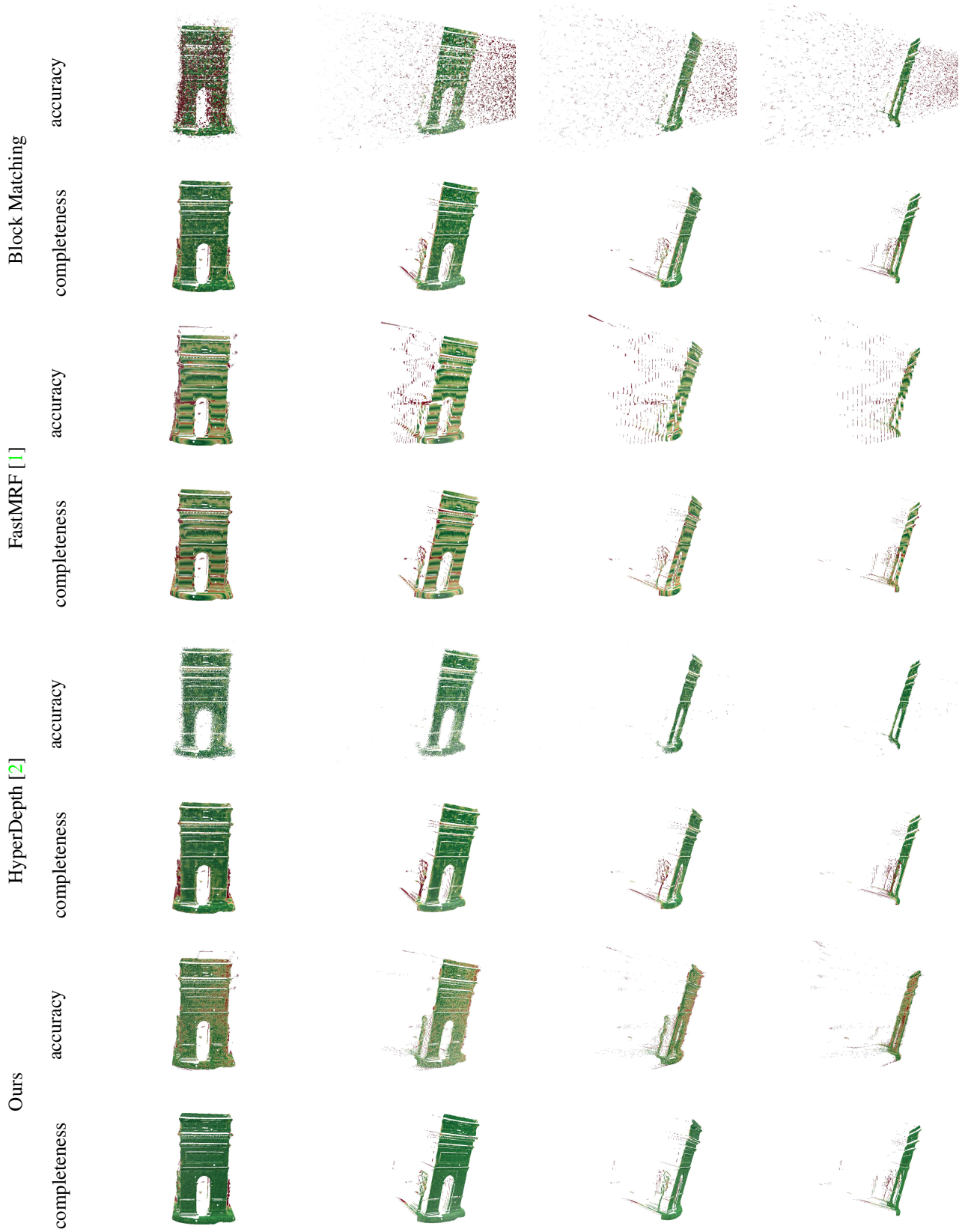


Figure 10: **Additional Qualitative 3D Results on Real Data.** *Arch*. Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0mm$, over yellow $= 5mm$, to red $\geq 1cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point. Arch.

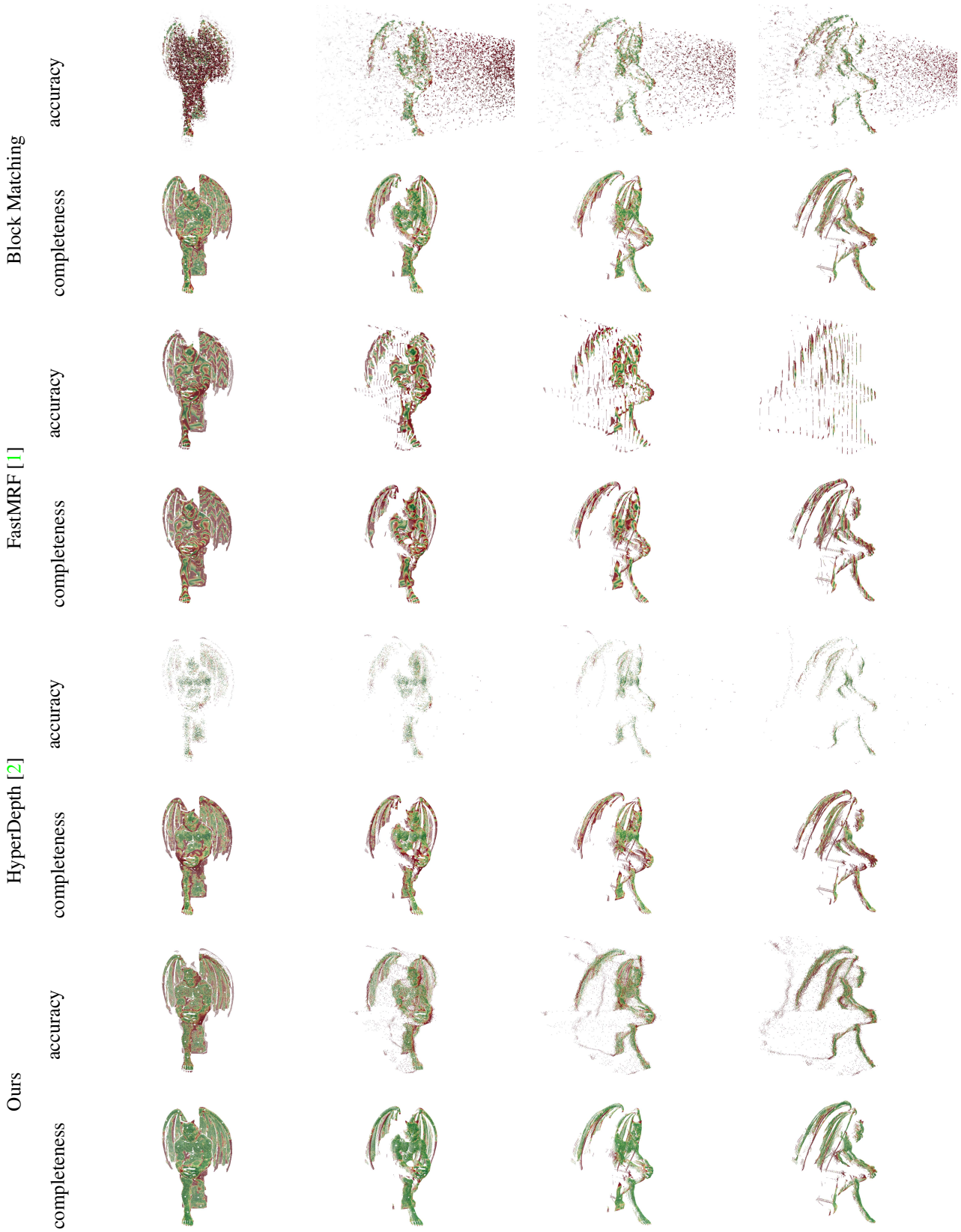
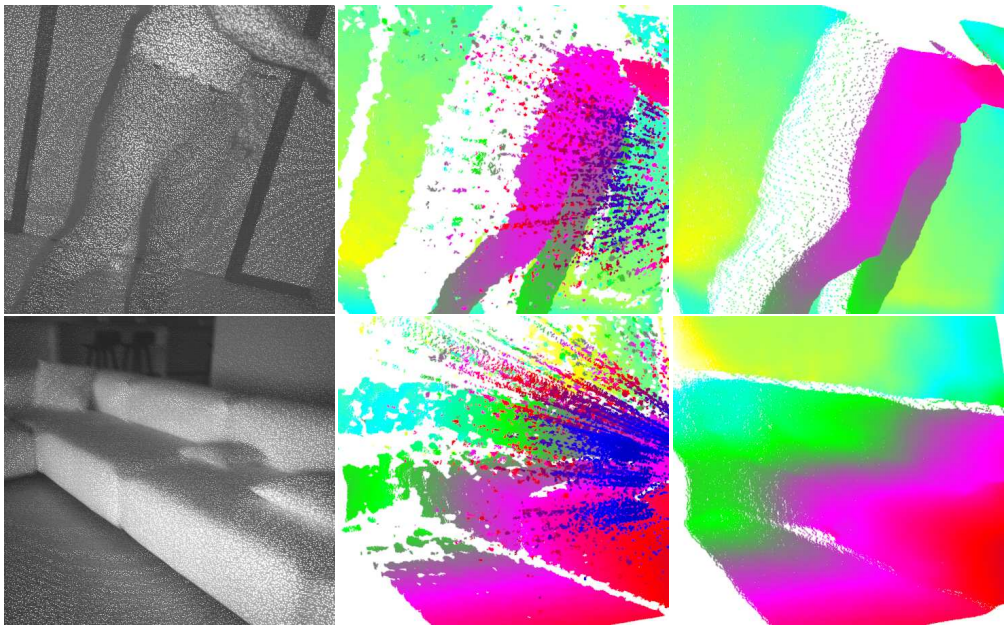


Figure 11: **Additional Qualitative 3D Results on Real Data.** *Gargoyle*. Event rows depict accuracy results: 3D point-cloud from estimated depthmap. The color indicates the distance to the closest 3D point of the ground-truth model, from dark green $= 0mm$, over yellow $= 5mm$, to red $\geq 1cm$. Odd rows show the completeness: 3D point-cloud of the ground-truth with the same color coding indicating the distance to the closest estimated 3D point.



(a) Input (b) Block Matching (c) Ours

Figure 12: **Qualitative 3D Results on Real Data in Complex Real-World Scenarios.**

References

[1] Q. Chen and V. Koltun. Fast MRF optimization with application to depth reconstruction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[2] S. R. Fanello, C. Rhemann, V. Tankovich, A. Kowdle, S. Orts-Escolano, D. Kim, and S. Izadi. Hyperdepth: Learning depth from structured light without matching. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of the International Conf. on Machine learning (ICML)*, 2015.

[4] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *arXiv preprint arXiv:1807.03247*, 2018.

[5] N. Mayer, E. Ilg, P. Haeusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.