A Benchmark and a Baseline for Robust Multi-view Depth Estimation – Appendix –

A1. Test sets

As specified in the main paper in Sec. 3.1, Test sets, we provide more details on the construction of the test sets for the Robust MVD Benchmark in the following.

KITTI [17] The KITTI test set is based on the commonly used Eigen test split [3], which contains 697 samples. More specifically, evaluation is done only on samples of the Eigen split where dense ground truth depth from Uhrig *et al*. [17] is available (652 samples), and where ground truth poses from the KITTI odometry benchmark are available (95 samples). For each sample, sequences with 10 source views before and 10 source views after the keyview are used. This additional restriction leads to the final KITTI test set with 93 samples.

ETH3D [14] On ETH3D, the test set is based on the original training split of the high-resolution multi-view benchmark with a total of 13 sequences and 454 views. For the test set, 8 views of each sequence are used as keyviews, resulting in a total of 104 samples. Each sample contains 10 source views, using the view selection provided by https://github.com/FangjinhuaWang/PatchmatchNet [19].

ScanNet [2] The ScanNet test set is based on the split from Teed and Deng [16], which in turn extends the split by Tang and Tan [15]. The split consists of 2000 samples taken from 90 of the 1513 totally available sequences. Each sample comprises a keyview and 7 source views, with 3 views before and 4 views after the keyview. For the benchmark test set, every 10th sample of the original split is used, resulting in a total of 200 samples. Images are resized to a resolution of 640x480px to match ground truth depth maps.

DTU [7, 1] On DTU, the test set is based on the evaluation split used in MVSNet [23]. The split comprises 22 scans, where each scan has 49 frames. For the benchmark test set, 5 views of each scan are used as keyviews, resulting in a total of 110 samples. Each sample contains 10 source views, using the view selection and depth maps provided by https://github.com/YoYo000/MVSNet [23].

Tanks and Temples (T&T) [8] The Tanks and Temples test is based on the scenes "Barn", "Courthouse", "Church", and "Ignatius" from the original training split. Other scenes from the training split were not used, as they were not available for download at the time of writing. To construct the test set, we start with the provided image sets of the respective scenes. We reconstruct camera poses and intrinsics by running COLMAP [13, 12] on the images. The reconstructed camera poses are aligned with the ground truth using the evaluation script from Tanks and Temples (https://github.com/isl-org/TanksAndTemples/tree/master/python_toolbox/evaluation). To obtain ground truth depth maps for each image, we use the provided ground truth pointclouds for each scene and project the points into images. After outlier filtering, we obtain ground truth depth maps, as shown in Fig. A1. To speed up evaluation, only few images per scene are used as keyviews for the test set (Barn: 18; Church: 25; Couthouse: 13; Ignatius: 13), resulting in a total of 69 samples. For each keyview, 10 source views are used, which are selected with the view selection script from https://github.com/FangjinhuaWang/PatchmatchNet/blob/main/colmap_input.py [19].

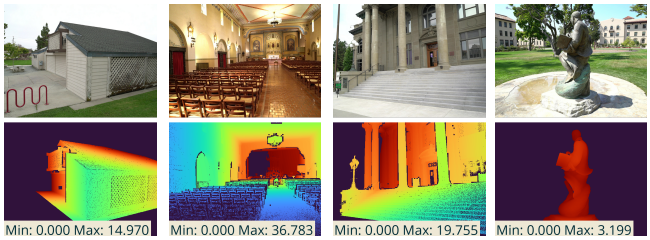

Figure A1. **Tanks and Temples ground truth depth**: the first row shows keyview images, the second ground truth depth maps that were reconstructed from the provided pointclouds. (a) Barn scene. (b) Church scene. (c) Courthouse scene. (d) Ignatius scene.

A2. Uncertainty estimation metrics

As specified in the main paper in Sec. 3.1, Uncertainty estimation metrics, we provide further details on the uncertainty estimation metrics and evaluation results in the following. In the main paper, we show Sparsification Error Curves for all evaluated models. Additionally, here, we explain and provide Sparsification Curves for all evaluated models, which should help in interpreting the Sparsification

Error Curves.

Sparsification Curves Sparsification Curves indicate the error reduction on a metric (here: the Absolute Relative Error) when excluding a certain fraction of pixels from the metric computation based on some ranking. This ranking can be based on estimated uncertainties or on actual errors relative to the ground truth. When using estimated uncertainties for the ranking, a monotonically decreasing Sparsification Curve indicates that uncertainties are aligned with actual errors. The ranking based on actual errors to the ground truth is referred to as Oracle Sparsification Curve and serves as a lower bound, as it represents a hypothetical optimal alignment of uncertainties and actual errors. Hence, smaller differences between both Sparsification Curves indicate better uncertainty estimates. The Sparsification Error Curves from the main paper are the curves that one obtains when taking the difference between both Sparsification Curves.

Both Sparsification Curves (Oracle and uncertainty-based) for different models are shown in Fig. A2 and A3. The difference between both curves are exactly the Sparsification Error Curves of the respective models that are shown in Fig. 3 of the main paper.

Finally, as a side-note, Sparsification can be measured on each sample individually, or on the whole test set. In this work, we compute Sparsification Curves on each sample of a test set individually and then compute the average across all samples in the test set. Following this, we compute the average across all test sets. In Fig. A2 and A3, we plot the average across all test sets in a bold color and indicate the Sparsification Curves on individual test sets in light colors. Additionally, we indicate the standard deviation across test sets as shaded areas in the plots.

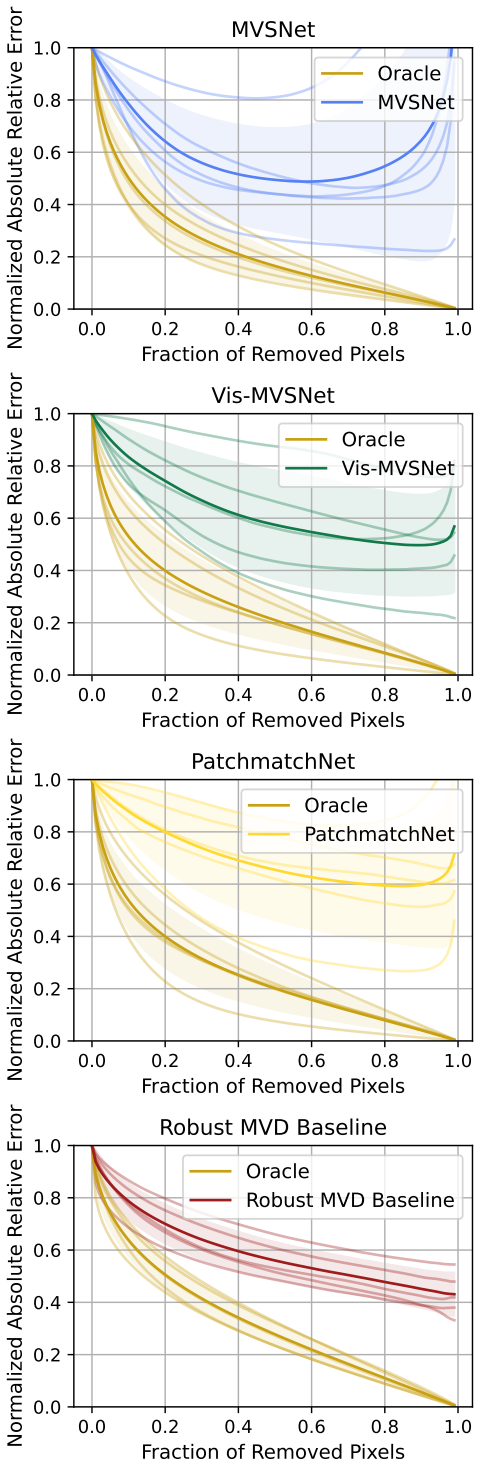

Figure A2. **Sparsification Curves for the Robust MVD Baseline model, MVSNet, Vis-MVSNet and PatchmatchNet.** The difference between both curves is exactly the Sparsification Error Curve from Fig. 3 in the main paper.

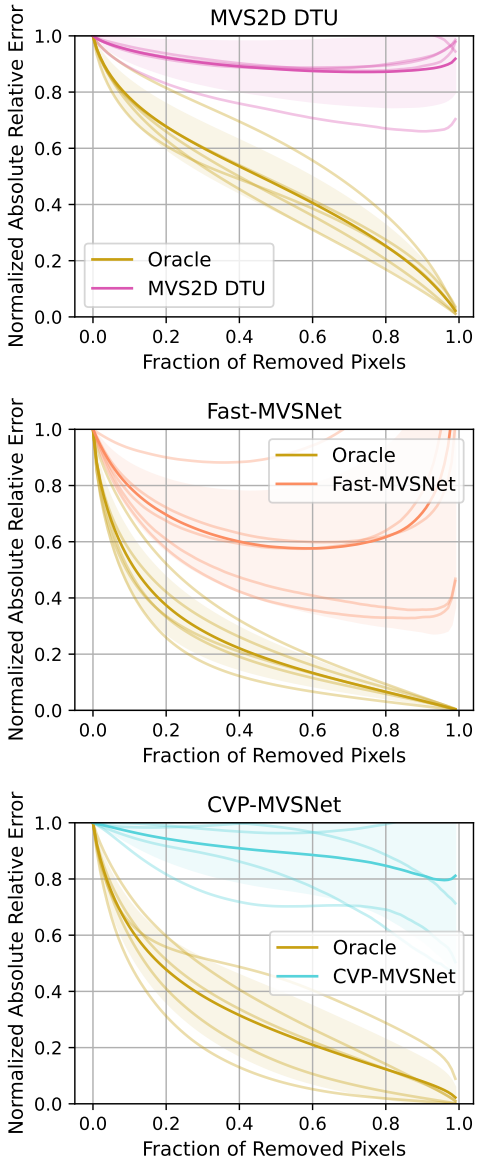

Figure A3. **Sparsification Curves for Fast-MVSNet, CVP-MVSNet and MVS2D.** The difference between both curves is exactly the Sparsification Error Curve from Fig. 3 in the main paper.

A3. Model runtimes

As specified in Sec. 3.2, Performances depending on source views of the main paper, we plot model runtimes for different numbers of source views in Fig. A4 and A5.

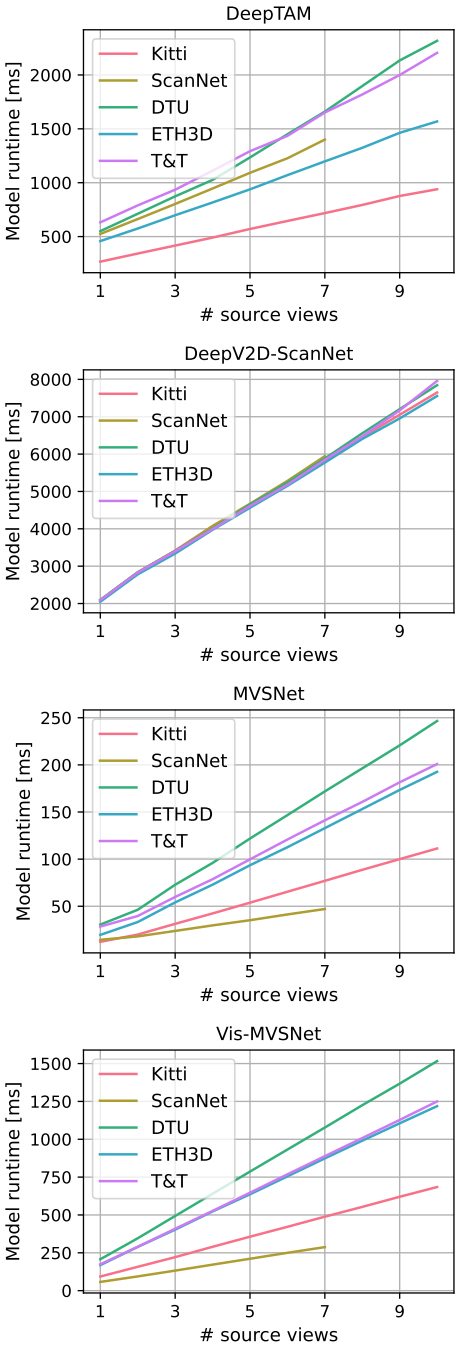

Figure A4. **Model runtimes for different numbers of source views** for DeepTAM, DeepV2D, MVSNet and Vis-MVSNet on all test sets. Different runtimes on the test sets are due to different input resolutions.

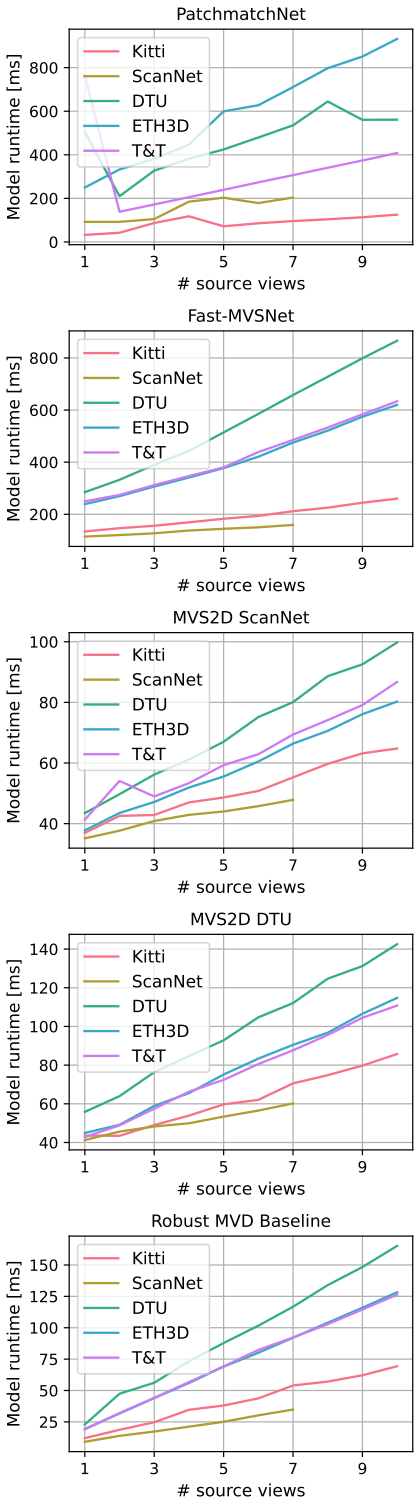

Figure A5. **Model runtimes for different numbers of source views** for PatchmatchNet, Fast-MVSNet, MVS2D and the Robust MVD Baseline model on all test sets. Different runtimes on the test sets are due to different input resolutions.

Operation	Kernel	Stride	Ch I/O	InpRes	OutRes	Input	Output
(1) Siamese encoder							
for each view $i = 0, .., k$:							
convolution	7×7	2	3/64	768×384	384×192	Image $\mathbf{I}_i$	conv1_i
convolution	5×5	2	64/128	384×192	192×96	conv1_i	conv2_i
convolution	3×3	2	128/256	192×96	96×48	conv2_i	conv3a_i
(2) Plane sweep correlation							
for each source view $i = 1, .., k$:							
correlate view 0 and i	-	-	$256/n = 256$	96×48	96×48	conv3a_0, conv3a_i	costvolume $\mathbf{C}_i$
(3) Context encoder							
convolution	1×1	1	256/32	96×48	96×48	conv3a_0	ctx
(4) Costvolume fusion							
4.1 for each source view $i = 1, .., k$:							
convolution	3×3	1	256/128	96×48	96×48	$\mathbf{C}_i$	convf1_i
convolution	3×3	1	128/1	96×48	96×48	convf1_i	weight $\mathbf{w}_i$
4.2 fuse to costvolume $\mathbf{C}$: $\mathbf{C} = (\sum \exp(\mathbf{w}_i) \cdot \mathbf{C}_i)/(\sum \exp(\mathbf{w}_i))$	-	-	256/256	96×48	96×48	$\mathbf{C}_i$, $\mathbf{w}_i$	$\mathbf{C}$
(5) Costvolume Decoder							
convolution	3×3	1	288/256	96×48	96×48	$\mathbf{C}$+ctx	conv3b
convolution	3×3	2	256/512	96×48	48×24	conv3b	conv4a
convolution	3×3	1	512/512	48×24	48×24	conv4a	conv4b
convolution	3×3	2	512/512	48×24	24×12	conv4b	conv5a
convolution	3×3	1	512/512	24×12	24×12	conv5a	conv5b
convolution	3×3	2	512/1024	24×12	12×6	conv5b	conv6a
convolution	3×3	1	1024/1024	12×6	12×6	conv6a	conv6b
convolution	3×3	1	1024/2	12×6	12×6	conv6b	pred6
transposed convolution	4×4	2	1024/512	12×6	24×12	conv6b	upconv5
convolution	3×3	1	1025/512	24×12	24×12	upconv5+pr6+conv5b	iconv5
convolution	3×3	1	512/2	24×12	24×12	iconv5	pred5
transposed convolution	4×4	2	512/256	24×12	48×24	iconv5	upconv4
convolution	3×3	1	769/256	48×24	48×24	upconv4+pr5+conv4b	iconv4
convolution	3×3	1	256/2	48×24	48×24	iconv4	pred4
transposed convolution	4×4	2	256/128	48×24	96×48	iconv4	upconv3
convolution	3×3	1	385/128	96×48	96×48	upconv3+pr4+conv3b	iconv3
convolution	3×3	1	128/2	96×48	96×48	iconv3	pred3
transposed convolution	4×4	2	128/64	96×48	192×96	iconv3	upconv2
convolution	3×3	1	193/64	192×96	192×96	$\text{upconv2+pr3+conv2}_0$	iconv2
convolution	3×3	1	64/2	192×96	192×96	iconv2	pred2
transposed convolution	4×4	2	64/32	192×96	384×192	iconv2	upconv1
convolution	3×3	1	97/32	384×192	384×192	$\text{upconv1+pr2+conv1}_0$	iconv1
convolution	3×3	1	32/2	384×192	384×192	iconv1	pred1

Table A1. **Architecture of the Robust MVD Baseline model**. The model is based on a DispNet architecture. The notation is the same as in the main paper, *i.e.* the model takes views $V = (V_0, \cdots, V_k)$ as input where V_0 is the keyview with an image $\mathbf{I}_0$ and $V_{1,..,k}$ are source views with images $\mathbf{I}_{i,..,k}$. n is the number of candidate inverse depth values in the plane sweep correlation layer (here: 256). The resolutions are indicated for training settings and differ for test inputs. The table is adapted from [10].

A4. Model architecture

As specified in Sec. 4.1, Model architecture of the main paper, we provide further details on the model architecture in the following.

The architecture is based on a standard DispNetC architecture [10] with the following adaptations: (1) the correlation layer is replaced by a plane sweep correlation layer, (2) costvolumes from multiple views are fused via weighted averaging with learned weights, and (3) all outputs have two channels instead of one, with the additional channel being the scale parameter of the predicted Laplace distribution. A summary of the architecture is provided in Tab. A1.

Instead of the original DispNet correlation layer, the model correlates in a plane sweep stereo fashion, which has been shown to work well in other multi-view depth architectures [16, 27, 23, 9, 5]. For each feature $\mathbf{f}_0$ at a location ${}^0\mathbf{x}$ in the keyview V_0, points ${}^i\mathbf{x}$ in a source view V_i are sampled by reprojecting ${}^0\mathbf{x}$ to the view V_i image plane for different candidate inverse depth values d as follows [4]:

$$ {}^i\tilde{\mathbf{x}}\,(d) = \mathbf{K}_i \cdot {}^i_0\mathbf{R} \cdot \mathbf{K}_0^{-1} \cdot {}^0\tilde{\mathbf{x}} + \mathbf{K}_i \cdot {}^i_0\mathbf{t} \cdot d, \qquad (1) $$

with ${}^0\tilde{\mathbf{x}}$ and ${}^i\tilde{\mathbf{x}}$ denoting ${}^0\mathbf{x}$ and ${}^i\mathbf{x}$ in homogeneous coordinates. We use $n = 256$ candidate inverse depth values d that are spaced equidistantly in the range $d_{min} = 0.001\,\mathrm{m}^{-1}$ to $d_{max} = 2.5\,\mathrm{m}^{-1}$. Given these sampling points, features

Approach	GT Poses	GT Range	Align	StaticThings3D rel ↓	τ ↑	AUSE ↓	time [s]	BlendedMVS rel ↓	τ ↑	AUSE ↓	time [s]
a) Classical:											
COLMAP [13, 12]	✓	✗	✗	5.8	79.5	-	≈ 2 min	1.4	97.2	-	≈ 2 min
COLMAP Dense [13, 12]	✓	✗	✗	16.6	65.7	-	≈ 2 min	4.2	92.3	-	≈ 2 min
b)											
DeMoN [18]	✗	✗	$\|\|\mathbf{t}\|\|$	36.4	6.3	-	0.07	34.4	14.5	-	0.08
DeepV2D ScanNet [16]	✗	✗	med	23.1	15.4	-	2.73	-	-	-	-
DeepV2D KITTI[16]	✗	✗	med	-	-	-	-	-	-	-	-
c)											
MVSNet [23]	✓	✓	✗	38.9	39.0	0.45	0.06	5.0	80.7	0.30	0.03
MVSNet Inv. Depth [23]	✓	✓	✗	26.5	43.3	0.59	0.12	46.1	14.8	0.38	0.07
Vis-MVSNet [26]	✓	✓	✗	10.4	59.3	0.42	0.47	2.5	92.6	0.20	0.47
CVP-MVSNet [21]	✓	✓	✗	-	-	-	-	-	-	-	-
PatchmatchNet [19]	✓	✓	✗	7.6	58.1	0.58	0.07	22.1	27.9	0.43	0.05
Fast-MVSNet [25]	✓	✓	✗	24.9	42.7	0.52	0.26	13.0	59.2	0.45	0.11
MVS2D DTU [22]	✓	✓	✗	88.3	2.6	0.52	0.04	38.9	7.3	0.54	0.04
MVS2D ScanNet [22]	✓	✓	✗	39.9	2.7	-	0.04	64.3	1.6	-	0.04
d) Absolute scale:											
DeMoN [18]	✓	✗	✗	36.3	6.2	-	0.08	44.2	7.6	-	0.08
DeepTAM [27]	✓	✗	✗	-	-	-	-	-	-	-	-
DeepV2D KITTI [16]	✓	✗	✗	-	-	-	-	-	-	-	-
DeepV2D ScanNet [16]	✓	✗	✗	86.2	1.8	-	1.92	-	-	-	-
MVSNet [23]	✓	✗	✗	26.1	42.2	0.48	0.12	-	-	-	-
MVSNet Inv. Depth [23]	✓	✗	✗	53.4	4.7	-	0.14	29.1	48.7	-	0.07
CVP-MVSNet [21]	✓	✗	✗	-	-	-	-	-	-	-	-
Vis-MVSNet [26]	✓	✗	✗	9.6	59.2	0.39	0.47	-	-	-	-
PatchmatchNet [19]	✓	✗	✗	42.5	2.6	0.36	0.04	44.7	8.4	0.38	0.06
Fast-MVSNet [25]	✓	✗	✗	26.1	35.9	-	0.21	174.5	15.8	-	0.19
MVS2D ScanNet [22]	✓	✗	✗	92.4	0.0	-	0.04	29.9	52.3	-	0.05
MVS2D DTU [22]	✓	✗	✗	97.2	0.0	0.02	0.04	45.6	21.1	0.19	0.04
Robust MVD Baseline	✓	✗	✗	6.3	54.7	0.19	0.04	3.4	81.4	0.20	0.04

Table A2. **Quantitative results on StaticThings3D and BlendedMVS test sets.** Results are reported for the Absolute Relative Error (rel), the Inlier Ratio (τ) with a threshold of 1.25, the Area Under Sparsification Error Curve (AUSE) and the model runtime in seconds. Results are for the quasi-optimal selection of source views. COLMAP results are not directly comparable due to a lower prediction density.

$\mathbf{f}_i = \mathbf{F}_i({}^i\mathbf{x})$ are sampled from the view V_i feature map $\mathbf{F}_i$ with bilinear interpolation. The sampled features are compared to the keyview feature with a dot product $\mathbf{f}_0 \cdot \mathbf{f}_i$, resulting in correlation scores for all candidate inverse depth values. This is done for all keyview features, resulting in a costvolume $\mathbf{C}_i$ with pixel-wise correlation scores.

The costvolumes from multiple source views are fused via weighted averaging and mapped to the predicted inverse depth map and uncertainty map by the costvolume decoder network.

A5. Training details

Training is done with four source views. On StaticThings3D, we randomly sample source views within a range of ± 3 frames around the keyview. On BlendedMVS, we randomly sample from the ten best source views as in [19]. Other than this, we use similar hyperparameters as in the original DispNet, *e.g.* train for 600k iterations (ca. 60 hours on a single 3090 RTX GPU), use the Adam optimizer, start with a learning rate of 1e-4 and decay it by a factor of 0.5 after 300k, 400k and 500k iterations. The network outputs predicted inverse depth maps at different resolutions and is trained with a negative log-likelihood loss for every resolution (see [6]).

A6. StaticThings3D and BlendedMVS Results

The proposed Robust MVD Baseline model is trained on StaticThings3D and BlendedMVS. In the main paper, we did not evaluate on these datasets, as these datasets are not commonly used for evaluation and as StaticThings3D has little overlap with real-world data. We provide results on test sets from these two datasets in the following.

StaticThings3D (ST3D) test set The StaticThings3D test split contains 600 sequences with 10 views each. We define one sample per sequence, using the fifth view as keyview plus 4 source views before and 5 after the keyview, resulting in 600 samples. To speed up evaluation, we use every 10th sample for the test set, resulting in a total of 60 samples.

BlendedMVS (BMVS) [24] test set The BlendedMVS test set is based on the original validation split, which contains 7 scenes with a total of 915 frames. For each frame, a selection of 10 source views is provided by the authors. For our test set, we use 10 frames per scene as keyviews, resulting in a total of 70 samples.

Results Results are provided in Tab. A2.

A7. Application to real-world data with known poses

A motivation for the Robust MVD Benchmark were real-world applications where camera poses are known and the objective is to reconstruct depth maps at real-world scale. To illustrate this, we conduct experiments on a small self-captured datasets that we term RealThings. RealThings is captured indoors with a handheld Intel RealSense D435 RGB-D camera and uses April Tags [20] for extrinsic calibration. This can be seen as a typical scenario in robotics, where it is not uncommon to use April Tags.

Qualitative outputs of the Robust MVD Baseline model are shown in Fig. A6. The model was applied with a fixed number of 6 input views and without changing any configurations. This shows that the model can be directly applied for depth estimation in real-world scenarios.

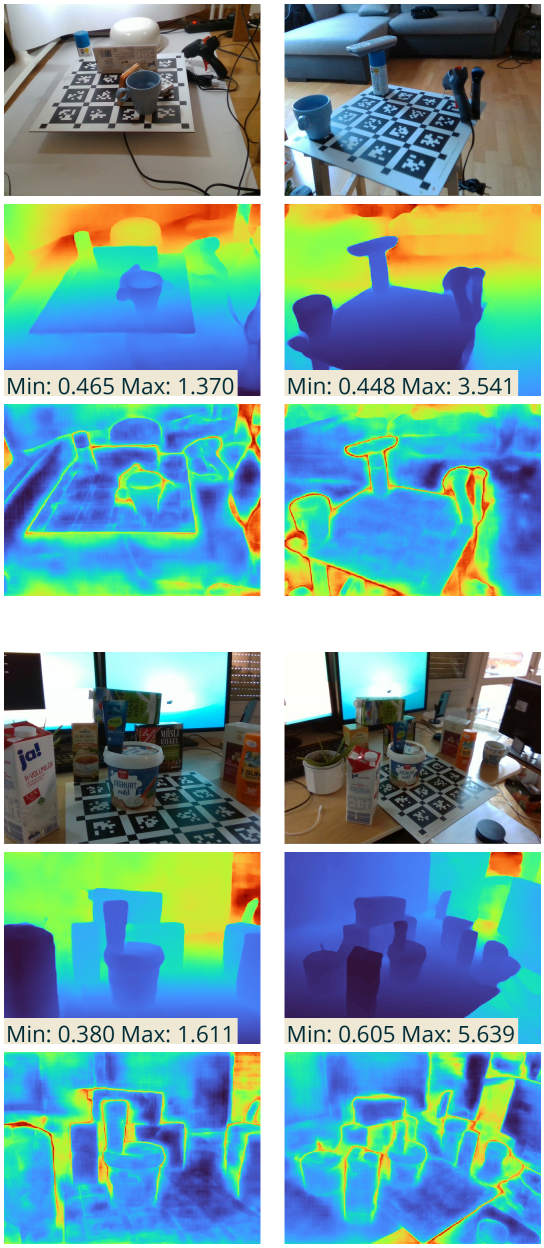

Figure A6. **Qualitative results of the Robust MVD Baseline on samples from the self-captured RealThings dataset.** For each sample, the figure shows three images: the first row shows the keyview image, the second row the predicted depth map, and the third row the predicted uncertainty. This example shows real-world application in a scenario where camera poses are known, *e.g.* from April Tags (tags are visible in the images). In such scenarios, the proposed Robust MVD Baseline model can be applied without any reconfiguration to derive depth maps at real-world scale. Blue colors indicate small values, red colors large values, and the ranges are given in meters.

A8. Limitations

The proposed Robust MVD Baseline model works well across domains and scales and gives reasonable predictions for most inputs. However, there is still room for improvement regarding overall accuracy of the predictions. Furthermore, the model fails for cases where it is difficult to match between the keyview image and source view images, *e.g.* specular reflections. Usually this results in incorrect depth estimates that are at least indicated by high uncertainties. However, in some cases (probably because incorrect matches due to reflections are consistent across source views), the erroneous depth estimates are not well represented by the estimated uncertainty (*e.g.* the TV screen in Fig. A7). Other obvious failure cases are common problems of multi-view depth estimation, namely small camera motion and dynamically moving objects. These cases could be resolved through stronger single-image priors [11], or by explicit detection and motion estimation for dynamic objects. The proposed model is designed to serve as a baseline on the proposed Robust MVD Benchmark and we look forward to future work that improves upon it. Examples for problematic cases are shown in Fig. A7.

A9. Qualitative results

In Fig. A8, Fig. A9, and Fig. A10, we show qualitative results of evaluated models. For models from related works, we show qualitatives for the three evaluation settings, namely for depth-from-video, multi-view stereo and absolute scale depth estimation.

In Fig. A11 and Fig. A12, we show additional qualitative results of the proposed Robust MVD Baseline model on samples from different domains and scene scales.

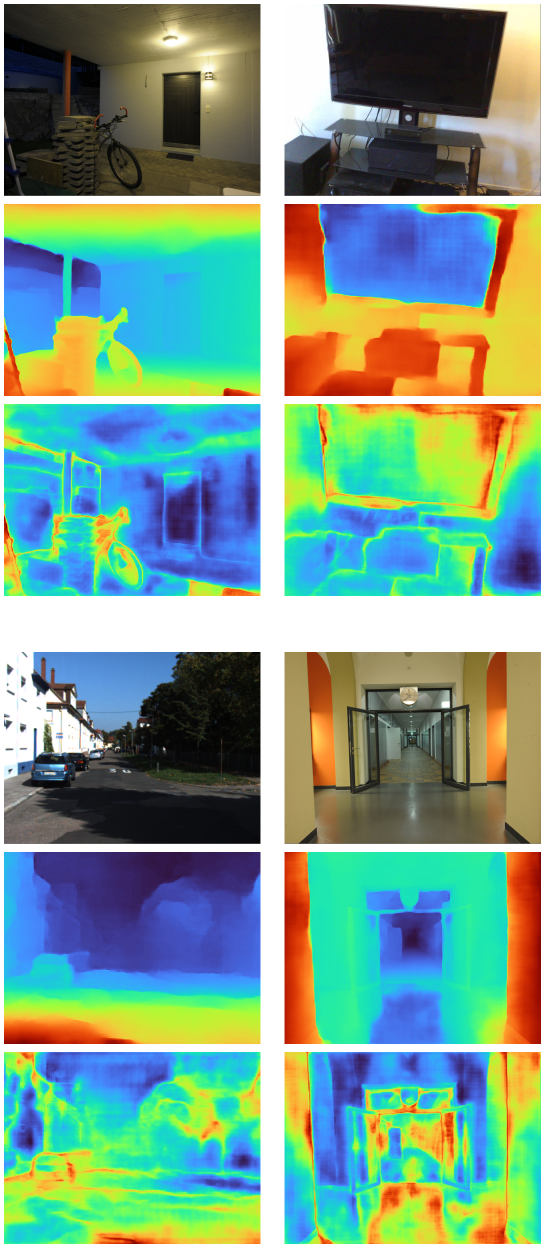

Figure A7. **Limitations of the Robust MVD Baseline.** For each sample, the figure shows three images: the first row shows the keyview image, the second the predicted inverse depth map, and the third the predicted uncertainty. The figure illustrates problematic cases where predictions are inaccurate, *e.g.* for fine structures (bike strokes in the first sample), reflective surfaces (screen in the second sample; floor in the last sample), or strong gradients in the color image (shadow on the road in the third sample).

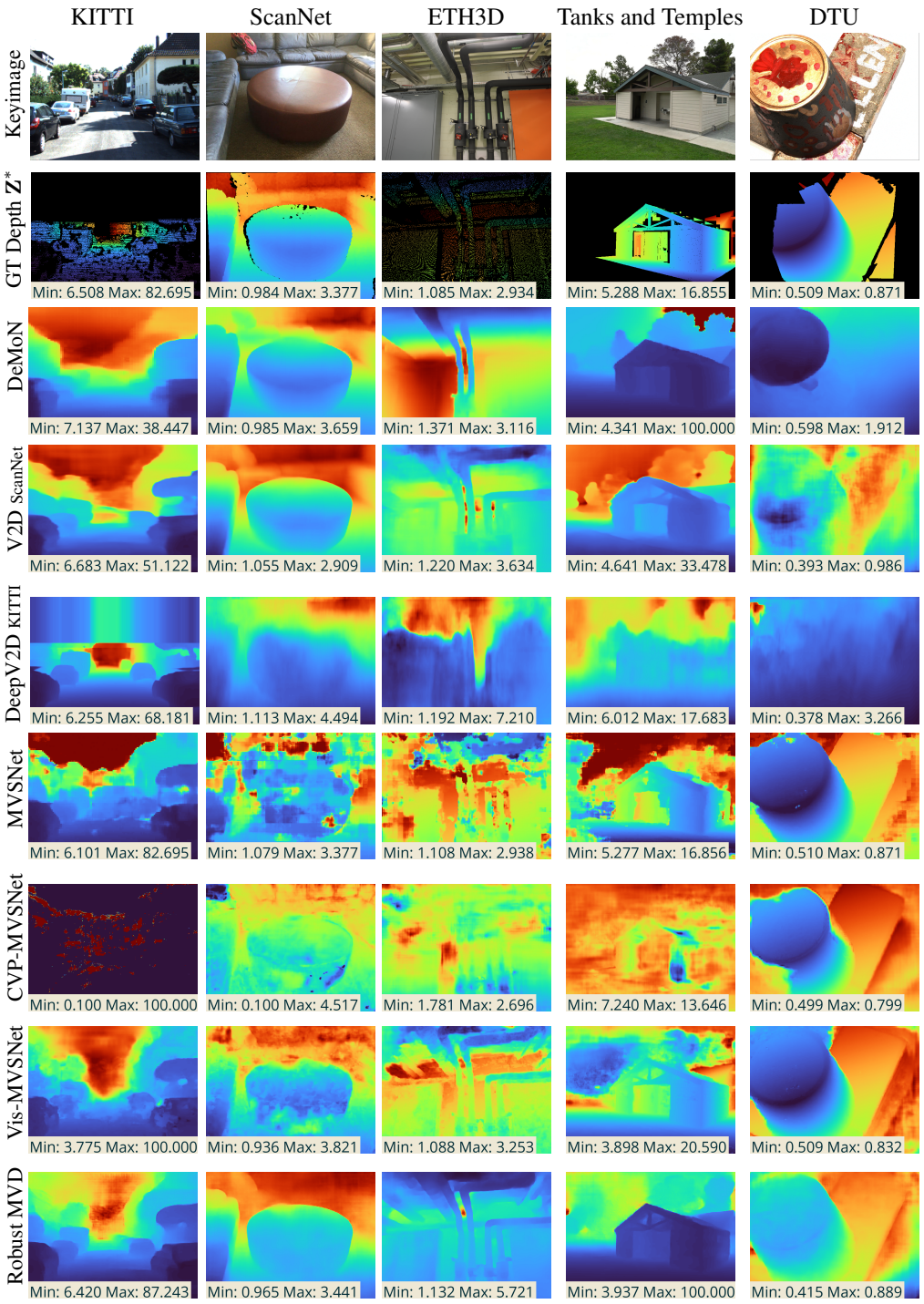

Figure A8. **Predicted depth maps of evaluated models on all test sets.** Models from related works are evaluated up to a relative scale, *i.e*. DeMoN and DeepV2D (V2D) are evaluated in the depth-from-video setting (without ground truth poses, predicted depths are aligned to the ground truth) and MVSNet, CVP-MVSNet and Vis-MVSNET are evaluated in the multi-view stereo setting (ground truth poses and ground truth depth range are provided to the model). The predictions hence correspond to the quantitative results in Tab. 2b and Tab. 2c of the main paper. The Robust MVD model is evaluated in the absolute depth setting (ground truth poses are provided to the model, no alignment between predicted and ground truth depths; Tab. 2d). All models from related works perform significantly better on data from their respective training domain. Blue colors indicate small values, red colors large values, and the ranges are given in meters. Corresponding predicted uncertainties are shown in Fig. A9.

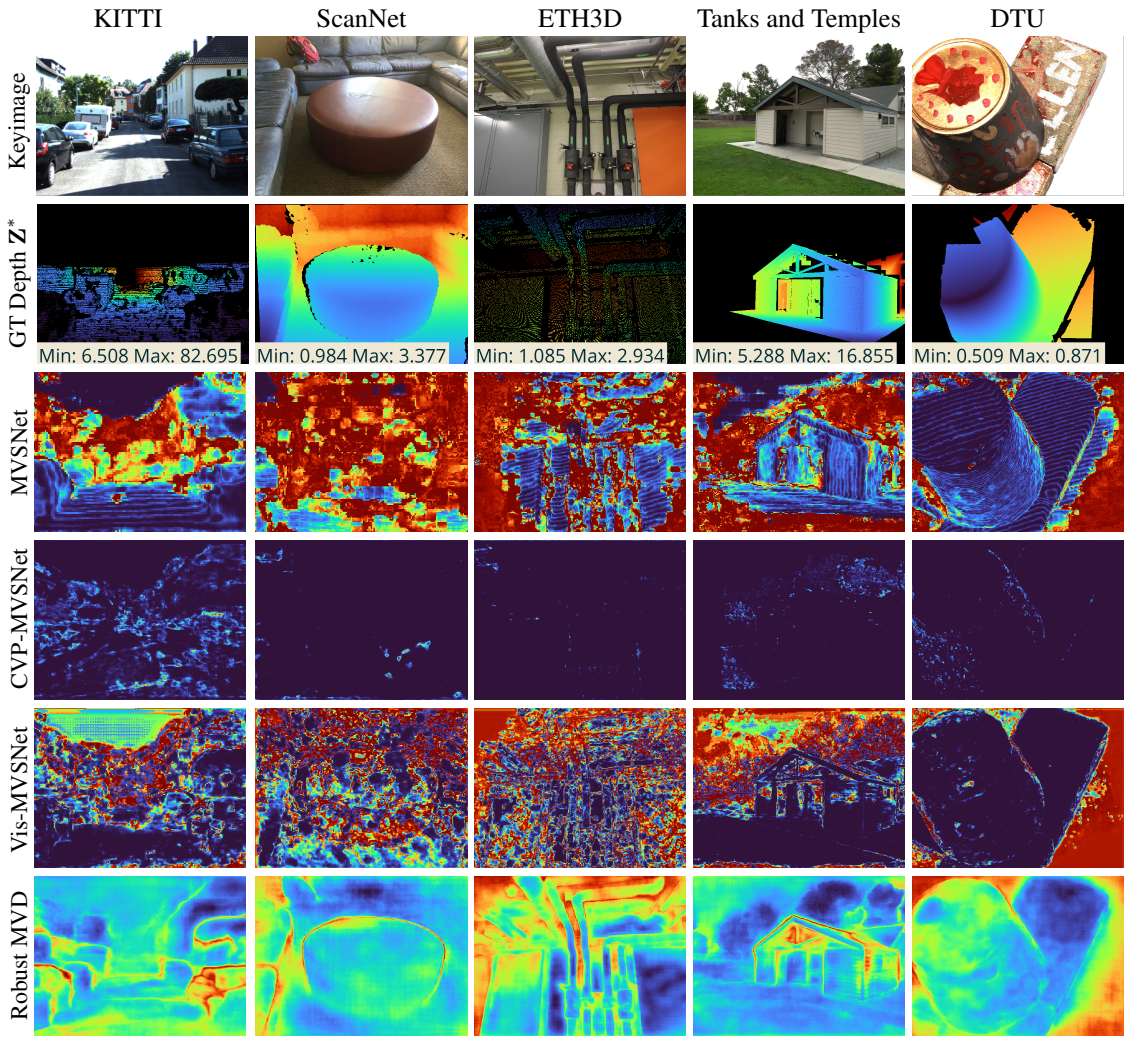

Figure A9. **Predicted uncertainties of evaluated models on all test sets.** Models from related works are evaluated up to a relative scale. The shown uncertainties hence correspond to the depth maps shown in Fig. A8 on the previous page and the quantitative results in Tab. 3 of the main paper. Blue colors indicate low uncertainties, red colors high uncertainties.

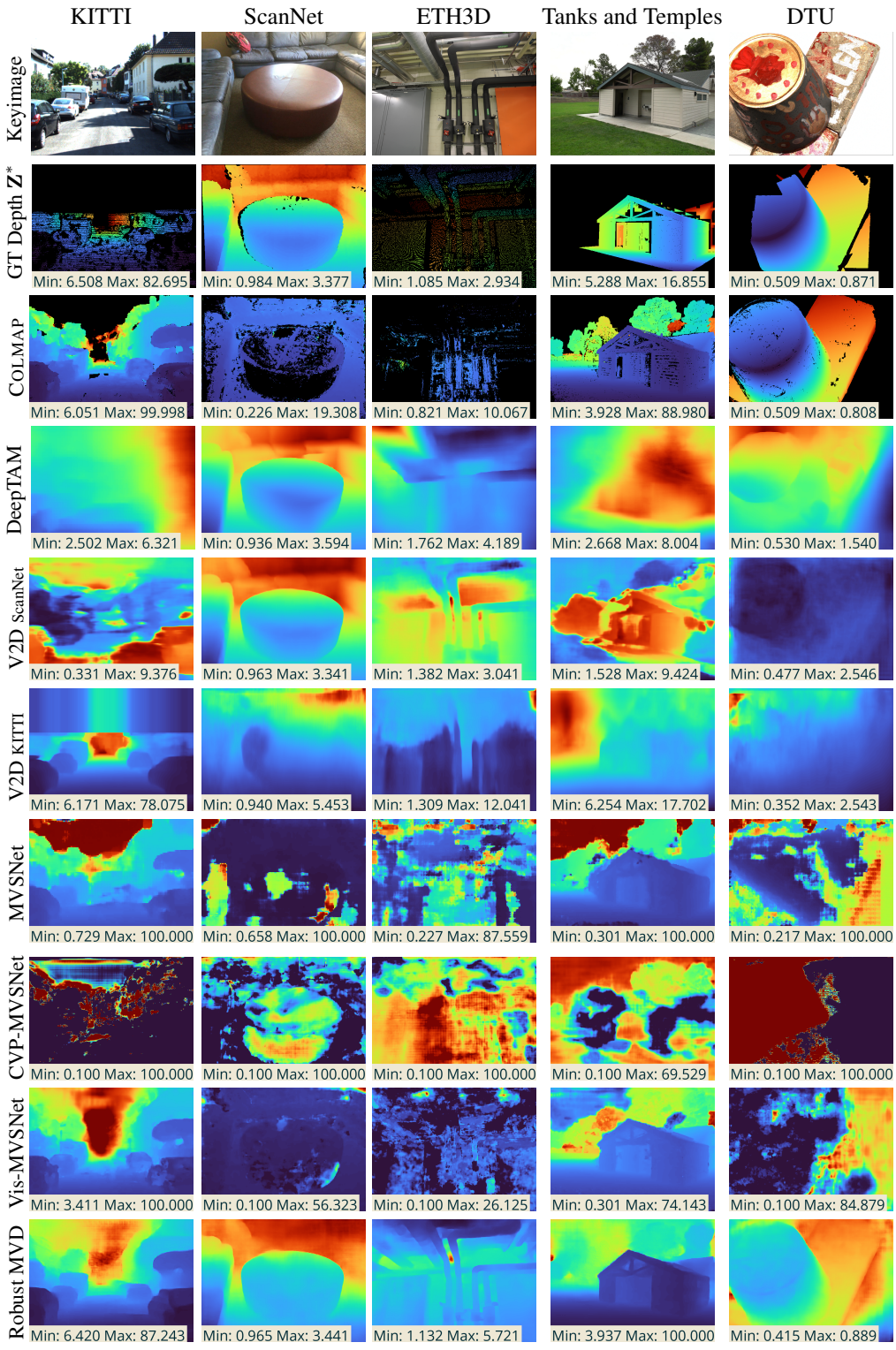

Figure A10. **Predicted depth maps of evaluated models on all test sets in the absolute depth estimation setting.** All models are applied with ground truth poses, without ground truth depth range, and without alignment between predicted and ground truth depths. The predictions shown here hence correspond to the quantitative results in Tab. 2d of the main paper. All models from related works overfit to the costvolume distribution seen during training and perform worse on inputs with different distributions. Blue colors indicate small values, red colors large values, and the ranges are given in meters.

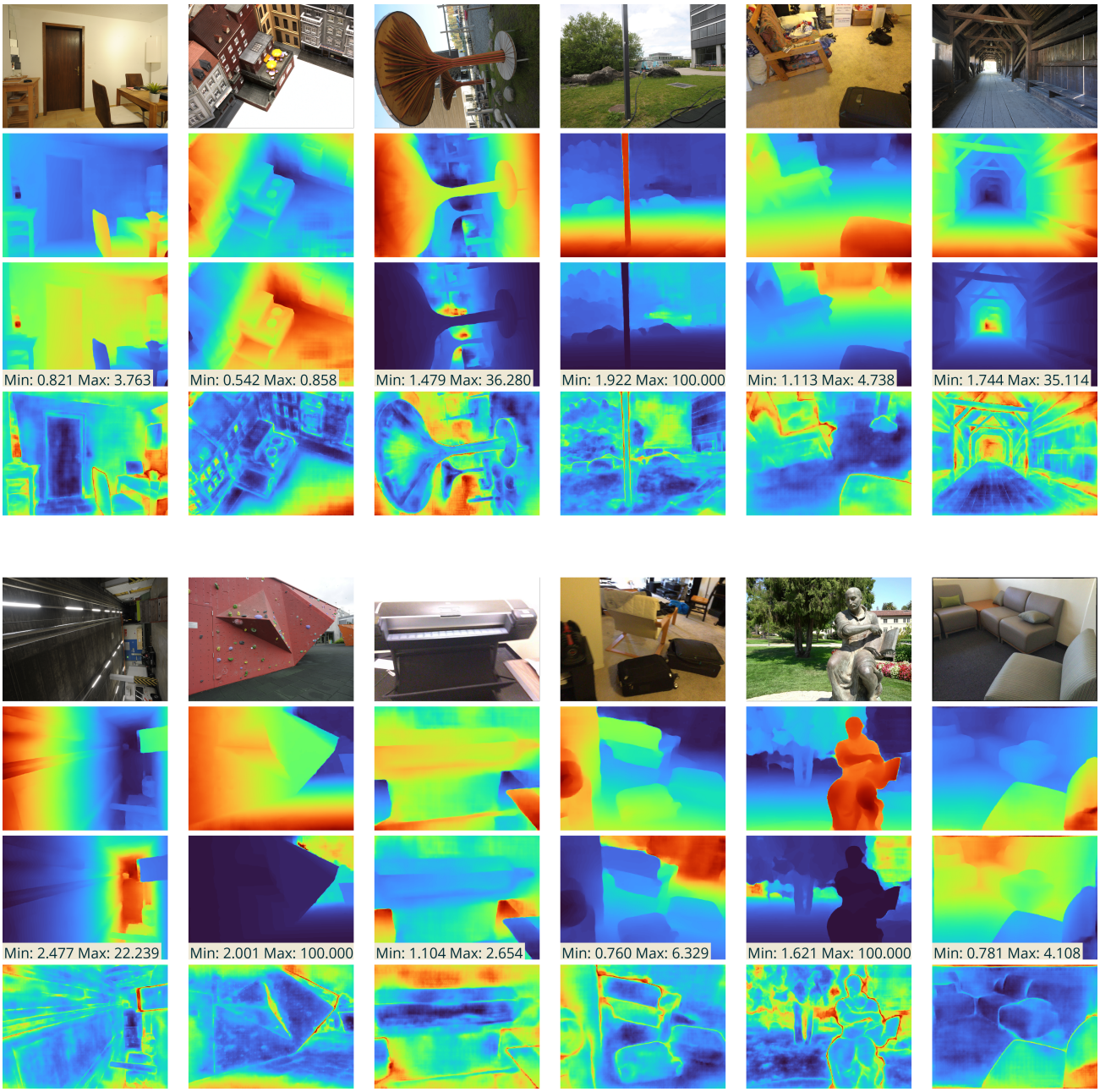

Figure A11. **Qualitative results of the Robust MVD Baseline** on samples from the KITTI, ScanNet, ETH3D, DTU, Tanks and Temples datasets. For each sample, the figure shows four images: the first row shows the keyview image, the second row the predicted inverse depth map, the third row the predicted depth map, and the fourth row the predicted uncertainty. Predicted inverse depth maps and predicted depth maps are equivalent. However, inverse depth maps are better suited for visualization, whereas depth maps allow for an easier interpretation of the scale of predicted values (here indicated in meters).

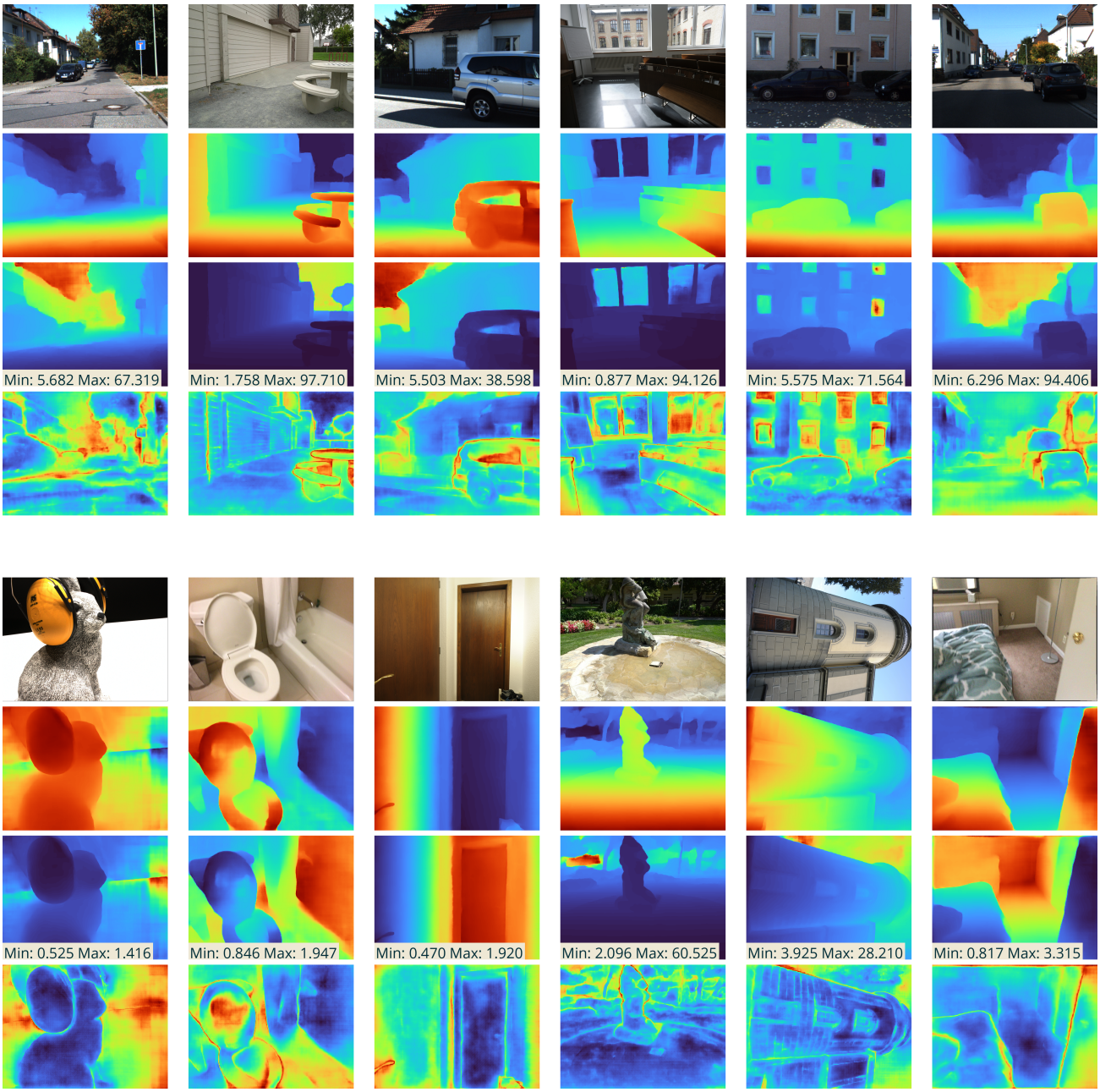

Figure A12. **Qualitative results of the Robust MVD Baseline** on samples from the KITTI, ScanNet, ETH3D, DTU, Tanks and Temples datasets. For each sample, the figure shows four images: the first row shows the keyview image, the second row the predicted inverse depth map, the third row the predicted depth map, and the fourth row the predicted uncertainty. Predicted inverse depth maps and predicted depth maps are equivalent. However, inverse depth maps are better suited for visualization, whereas depth maps allow for an easier interpretation of the scale of predicted values (here indicated in meters).

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 2016. 1

[2] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, July 2017. 1

[3] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, January 2014. 1

[4] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 4

[5] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *CVPR*, June 2018. 4

[6] Eddy Ilg, Özgün Çiçek, Silvio Galesso, Aaron Klein, Osama Makansi, F. Hutter, and T. Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *ECCV*, 2018. 5

[7] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, 2014. 1

[8] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and Temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 1

[9] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa Narasimhan, and Jan Kautz. Neural RGB-D sensing: Depth and uncertainty from a video camera. In *CVPR*, 2019. 4

[10] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, June 2016. 4

[11] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer. 2019. 7

[12] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *CVPR*, 2016. 1, 5

[13] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *ECCV*, 2016. 1, 5

[14] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *CVPR*, July 2017. 1

[15] Chengzhou Tang and Ping Tan. BA-Net: Dense bundle adjustment networks. In *ICLR*, May 2019. 1

[16] Zachary Teed and Jia Deng. DeepV2D: Video to depth with differentiable structure from motion. *ICLR*, 2020. 1, 4, 5

[17] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant CNNs. In *3DV*, October 2017. 1

[18] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 5

[19] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. PatchmatchNet: Learned multi-view patchmatch stereo, 2021. 1, 5

[20] John Wang and Edwin Olson. AprilTag 2: Efficient and robust fiducial detection. In *IROS*, October 2016. 6

[21] Jiayu Yang, Wei Mao, Jose M. Alvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *CVPR*, June 2020. 5

[22] Zhenpei Yang, Zhile Ren, Qi Shan, and Qixing Huang. MVS2D: Efficient multi-view stereo via attention-driven 2d convolutions. In *CVPR*, 2022. 5

[23] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. *ECCV*, September 2018. 1, 4, 5

[24] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. *CVPR*, 2020. 5

[25] Zehao Yu and Shenghua Gao. Fast-MVSNet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement. In *CVPR*, 2020. 5

[26] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. *BMVC*, 2020. 5

[27] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. DeepTAM: Deep tracking and mapping. In *ECCV*, September 2018. 4, 5