DeepSFM: Structure From Motion Via Deep Bundle Adjustment - Supplementary Material -

Xingkui Wei[1], Yinda Zhang[2], Zhuwen Li[3], Yanwei Fu[1], and Xiangyang Xue[1]

[1]Fudan University [2]Google Research [3]Nuro, Inc

1 Implementation Details

We implement our system using PyTorch. The training procedure takes 6 days on 3 NVIDIA TITAN GPUs to converge on all 160k training sequences. The training batch size is set to 4, and the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) is used with learning rate 2×10^{-4}, which decreases to 4×10^{-5} after 2 epochs. Within the first two epochs, the parameters in 2D CNN feature extraction module are initialized with pre-trained weights of [3] and frozen, and the ground truth depth maps for source images are used to construct D-CV and P-CV, which are replaced by predicted depth from network in latter epochs. During training, the length of input sequences is 2 (one target image and one source image). The sample number L for D-CV is set to 64 and the sample number P for P-CV is 1000. The range of both cost volumes is adapted during training and testing. For D-CV, its range is determined by the minimum depth values of the ground truth, which is the same as [3]. For P-CV, the bin size of rotation sampling is 0.035 and the bin size of translation sampling is $0.10 \times norm(t^*)$ for each initialization translation vector t^*.

Loss weights We follow two rules to set λ_r, λ_t and λ_d for $\mathcal{L}_{final}$: 1) the loss term provides gradient on the same order of numerical magnitude, such that no single loss term dominates the training process. This is because accuracy in depth and camera pose are both important to reach a good consensus. 2) we found in practice the camera rotation has higher impact on the accuracy of the depth but not the opposite. To encourage better performance of pose, we set a relatively large λ_r. In practice, the weight parameter λ for $\mathcal{L}_{depth}$ to balance loss objective is set to 0.7, while $\lambda_r = 0.8$, $\lambda_t = 0.1$ and $\lambda_d = 0.1$.

Feature extraction module As shown in Fig.1, we build our feature extraction module referring to DPSNet [3]. The module takes $4W \times 4H \times 3$ images as input and output feature maps of size $W \times H \times 32$, which are used to build D-CV and P-CV.

Cost volumes Figure 2 shows the detailed components for the P-CV and D-CV. Each channel of cost volume is composed of four components: reference view feature maps, warped source view feature maps, the warped source view initial depth map and the projected reference view depth plane or initial depth map. For P-CV construction, we take each sampled hypothetical camera pose, and carry out the warping process on source view depth maps and initial depth map based on the camera pose. And the initial reference view depth map is projected to align numeric values with the warped source view depth map. Finally those four components are concatenated as one channel of 4D P-CV. We do this on all P sampled camera poses, and get the P channel P-CV. The building approach for D-CV is similar, we take each sampled hypothetical depth plane, and carry out warping process on source view feature maps and the initial depth map. And the depth plane is projected to align with the source view depth map. After concatenation, one channel in D-CV is got. Same computation is done based on all L virtual depth planes, and the L channel D-CV is built up.

3D convolutional layers The detail architecture of 3D convolutional layers after D-CV is almost the same as DPSNet [3], except for the fist convolution layer. In order to compatible with the newly introduced depth consistent components in D-CV, We adjust the input channel number to 66 instead of 64. As shown in Fig.3, for 3D convolutional layers after P-CV, the architecture is similar to D-CV 3D convolution layers with three extra 3D average pooling layers and finally there is one global average pooling in the dimensions of image width and height, after which we get a $P \times 1 \times 1$ tensor.

2 Evaluation on ScanNet

ScanNet[1] provides a large set of indoor sequences with camera poses and depth maps captured from a commodity RGBD sensor. Following BA-Net[7], we leverage this dataset to evaluate the generalization capability by training models on DeMoN and testing here. The testing set is the same as BA-Net, which takes 2000 pairs filtered from 100 sequences.

We evaluate the generalization capability of DeepSFM on ScanNet. Table 1 shows the quantitative evaluation results for models trained on DeMoN. The results of BA-Net, DeMoN[9], LSD-SLAM[2] and Geometric BA[5] are obtained from [7]. As can be seen, our method significantly outperforms all previous work, which indicates that our model generalizes well to general indoor environments.

3 Evaluation on Tanks and Temples

As illustrated in Section4.2, We compare DeepSFM with COLMAP and R-MVSNet[10] on the Tanks and Temples[4] dataset. Figure 4 are more experimental results on Tanks and Temples dataset. All 7 training sequences provided by the dataset are used for the evaluation and the F-score are calculated as average. We add noise to COLMAP poses by down-scaling the images, subsampling temporal frames or directly add random Gaussian noise. Compared with COLMAP and R-MVSNet, our method is robuster to initialization quality.

Table 1. Results on ScanNet. (sc_inv: scale invariant log rms; **Bold**: best.)

Method	Depth					Motion	
	abs_rel	sq_rel	rms	log_rms	sc_inv	Rot	Trans
Ours	**0.227**	**0.170**	**0.479**	**0.271**	**0.268**	1.588	**30.613**
BA-Net	0.238	0.176	0.488	0.279	0.276	**1.587**	31.005
DeMoN	0.231	0.520	0.761	0.289	0.284	3.791	31.626
LSD-SLAM	0.268	0.427	0.788	0.330	0.323	4.409	34.360
Geometric BA	0.382	1.163	0.876	0.366	0.357	8.560	39.392

4 Computational costs

The computational costs on DeMoN dataset are shown in Table. 2. The memory cost of DeMoN and ours is the peak memory usage during testing on a TiTAN X GPU.

Table 2. The computational costs on DeMoN dataset.

Network	Ours	BANet	DeMoN
Memory/image	1.17G	2.30G	0.60G
Runtime/image	410ms	95ms	110ms
Resolution	640*480	320*240	256*192

Table 3. The performance of the optimization iterations for testing.

	Initialization	Iteration 2	Iteration 4	Iteration 6	Iteration 10	Iteration 20
abs relative	0.254	0.153	0.126	0.121	0.120	0.120
log rms	0.248	0.195	0.191	0.190	0.190	0.191
translation	15.20	9.75	9.73	9.73	9.73	9.73
rotation	2.38	1.43	1.40	1.39	1.39	1.39

Table 4. The performance with different warping methods.

MVS Dataset	L1-inv	sc-inv	L1-rel	Rot	Trans
Billinear interpolation	0.023	0.134	0.079	2.867	9.910
Nearest neighbor	0.021	0.129	0.076	2.824	9.881

5 More Ablation Study

5.1 More Iterations for Testing

We take up to four iterations when we train DeepSFM. During inference, the predicted depth maps and camera poses of previous iteration are taken as initialization of next iteration. To show how DeepSFM performs with more iterations than it is trained with, we show results in Table 3. We tested with up to 20 iterations, and it converges at the 6-th iteration.

5.2 Bilinear Interpolation vs Nearest Neighbor Sampling

For the construction of D-CV and P-CV, depth maps are warped via the nearest neighbor sampling instead of bilinear interpolation. Due to the discontinuity of the depth values in depth maps, the bilinear interpolation may bring some side effects. It may do damage to the geometry consistency and smooth the depth boundaries. As a comparison, we replace the nearest neighbor sampling with the bilinear interpolation. As shown in Table 4, the performance of our model gains a slight drop with the bilinear interpolation, which indicates that the nearest neighbor sampling method is indeed more geometrically meaningful for depth. In contrast, the differentiable bilinear interpolation is required for the warping of image features, whose gradients are back propagated to feature extractor layers. Further exploration will be an interesting future work.

5.3 Geometric consistency

We include both the image features and the initial depth values into the cost volumes to enforce photo-consistency and geometric consistency. To validate the geometric consistency, we conduct an ablation study on MVS dataset and show the depth accuracy w/ and w/o geometric consistency with same GT poses in Table 5. Meanwhile, as shown in Fig. 8, the geometric consistency is especially helpful for regions with weak photometric consistency, e.g. textureless, specular reflection.

5.4 Initialization data augmentation

Adding random noises to the initialization is a commonly used way to increase the robustness of the pipeline. As a comparison, we initialize our pipeline by

Table 5. Results on MVS with and w/o geometric consistency($\alpha = 1.25$). The metrics are the same as those on Eth3D datasets.

Method	abs_rel	abs_diff	sq_rel	rms	log_rms	$\delta < \alpha$	$\delta < \alpha^2$	$\delta < \alpha^3$
w/	0.0698	0.1629	0.0523	0.3620	0.1392	90.25	96.06	98.18
w/o	0.0813	0.2006	0.0971	0.4419	0.1595	88.53	94.54	97.35

adding small Gaussian random noises to DeMoN pose and depth map results and then fine-tune the network on the training set of DeMoN datasets. After the fine-tuning, we test our method on MVS dataset on which the network is not trained on, and the performance of our network decreases slightly after the data augmentation. This demonstrates that adding small Gaussian random noises dose not increase the generalization ability of our method on unseen data, since it's easy for the network to over fit the noise distribution.

Table 6. Results on MVS with and w/o scale invariant gradient loss($\alpha = 1.25$). The metrics are the same as those on Eth3D datasets.

Method	abs_rel	abs_diff	sq_rel	rms	log_rms	$\delta < \alpha$	$\delta < \alpha^2$	$\delta < \alpha^3$
w/	0.0712	0.1630	0.0531	0.3637	0.1379	90.25	96.02	98.24
w/o	0.0698	0.1629	0.0523	0.3620	0.1392	90.25	96.06	98.18

5.5 Smoothness on depth map

When compared with DeMoN[9], the output depth maps of ours are sometimes less spatially smooth. Besides L1 loss on the inverse depth map values, DeMoN applied scale invariant gradient loss for depth supervision, which enhances the smoothness of estimated depth maps. To address the smoothness issue, we add scale invariant gradient loss and set its weight as 1.5 times of L1 loss follow DeMoN to retrain the network. As shown in Table 6, no significant improvement is observed on depth evaluation metrics. Nevertheless, there are qualitative improvements of depth map in some samples as shown in Fig. 5.

5.6 Pose sampling

As described in section 3.3 of the paper, We use the same strategy for pose space sampling on different datasets. To show the generalization and the robustness of our method with different pose sampling strategies, we show the performance of our method with different bin size of rotation/translation without retraining in Table 7. Our model is a fully physical-driven architecture and shows well generalization ability.

Table 7. Results on MVS with different bin size (first column) of rotation/translation. The metrics are the same as those on DeMoN datasets.

Rotation(radian)	Rot error	Trans error	Translation(×norm)	Rot error	Trans error
0.07	3.024	9.974	0.20	3.252	10.117
0.05	2.916	9.890	0.15	3.080	9.758
0.03	2.825	9.836	0.10	2.825	9.836
0.02	2.893	9.941	0.05	3.308	11.013

6 Discussion with DeepV2D

DeepV2D[8] is a concurrent learning-based SfM method, which has shown excellent performance across a variety of datasets and tasks. We couldn't make a fair comparison with DeepV2D due to different settings of two methods. Here is a brief discussion and comparison. DeepV2D composes geometrical algorithms into the differentiable motion module and the depth module, and updates depth and camera motion alternatively. The depth module of DeepV2D builds a cost volume which is similar to our work except for the geometric consistency introduced by our method. The motion module of DeepV2D minimizes the photometric re-projection error between image features of each pair via Gauss-Newton iterations, while our method learns correspondence of photometric and geometric features between each pair by P-CV and 3D conv.

7 Visualization

We show some qualitative comparison with the previous methods. Since there are no source code available for BA-Net [7], we compare the visualization results of our method with DeMoN [9] and COLMAP [6]. Figure 6 shows the predicted dense depth map by our method and DeMoN on the DeMoN datasets. As we can see, demon often miss some details in the scene, such as plants, keyboard and table legs. In contrast, our method reconstructs more shape details. Figure 7 shows some estimated results from COLMAP and our method on the ETH3D dataset. As shown in the figure, the outputs from COLMAP are often incomplete, especially in textureless area. On the other hand, our method performs better and always produce an integral depth map. In Fig.8, more qualitative comparisons with COLMAP on challenging materials are provided.

References

1. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
2. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European conference on computer vision. pp. 834–849. Springer (2014)
3. Im, S., Jeon, H.G., Lin, S., Kweon, I.S.: Dpsnet: End-to-end deep plane sweep stereo. In: International Conference on Learning Representations (2019), `https://openreview.net/forum?id=ryeYHi0ctQ`
4. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4) (2017)
5. Nistér, D.: An efficient solution to the five-point relative pose problem. IEEE transactions on pattern analysis and machine intelligence **26**(6), 756–770 (2004)
6. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4104–4113 (2016)
7. Tang, C., Tan, P.: Ba-net: Dense bundle adjustment network. arXiv preprint arXiv:1806.04807 (2018)
8. Teed, Z., Deng, J.: Deepv2d: Video to depth with differentiable structure from motion. arXiv preprint arXiv:1812.04605 (2018)
9. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: Demon: Depth and motion network for learning monocular stereo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5038–5047 (2017)
10. Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., Quan, L.: Recurrent mvsnet for high-resolution multi-view stereo depth inference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5525–5534 (2019)

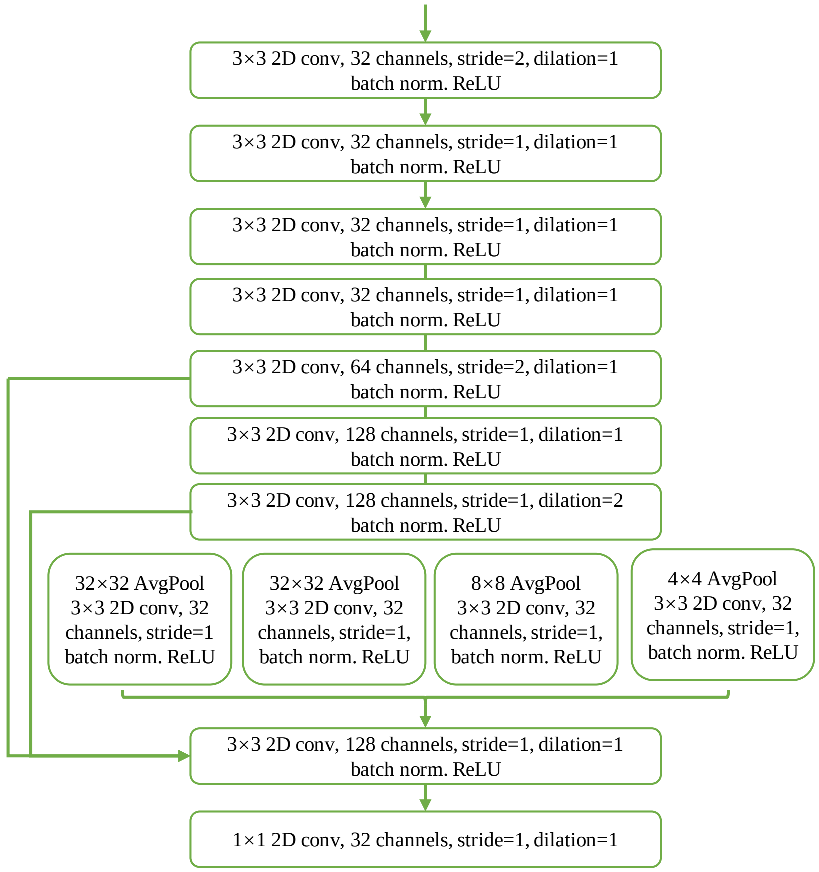

3×3 2D conv, 32 channels, stride=2, dilation=1
batch norm. ReLU
3×3 2D conv, 32 channels, stride=1, dilation=1
batch norm. ReLU
3×3 2D conv, 32 channels, stride=1, dilation=1
batch norm. ReLU
3×3 2D conv, 32 channels, stride=1, dilation=1
batch norm. ReLU
3×3 2D conv, 64 channels, stride=2, dilation=1
batch norm. ReLU
3×3 2D conv, 128 channels, stride=1, dilation=1
batch norm. ReLU
3×3 2D conv, 128 channels, stride=1, dilation=2
batch norm. ReLU
32×32 AvgPool
3×3 2D conv, 32
channels, stride=1
batch norm. ReLU
32×32 AvgPool
3×3 2D conv, 32
channels, stride=1,
batch norm. ReLU
8×8 AvgPool
3×3 2D conv, 32
channels, stride=1,
batch norm. ReLU
4×4 AvgPool
3×3 2D conv, 32
channels, stride=1,
batch norm. ReLU
3×3 2D conv, 128 channels, stride=1, dilation=1
batch norm. ReLU
1×1 2D conv, 32 channels, stride=1, dilation=1

Fig. 1. Detail architecture of feature extractor.

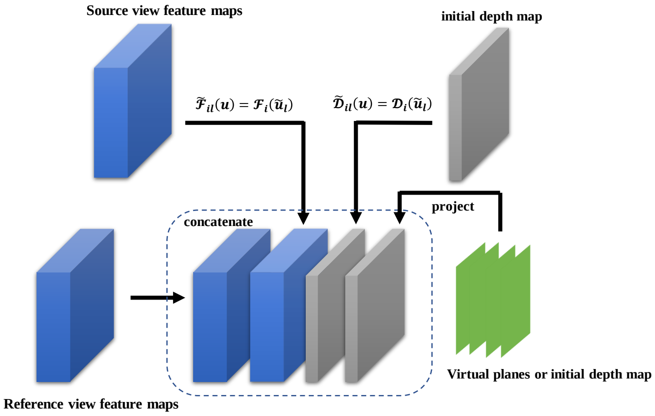

Source view feature maps
initial depth map
$\widetilde{\mathcal{F}}_{il}(\boldsymbol{u}) = \mathcal{F}_i(\widetilde{\boldsymbol{u}}_l)$
$\widetilde{\mathcal{D}}_{il}(\boldsymbol{u}) = \mathcal{D}_i(\widetilde{\boldsymbol{u}}_l)$
project
concatenate
Virtual planes or initial depth map
Reference view feature maps

Fig. 2. Four components in D-CV or P-CV.

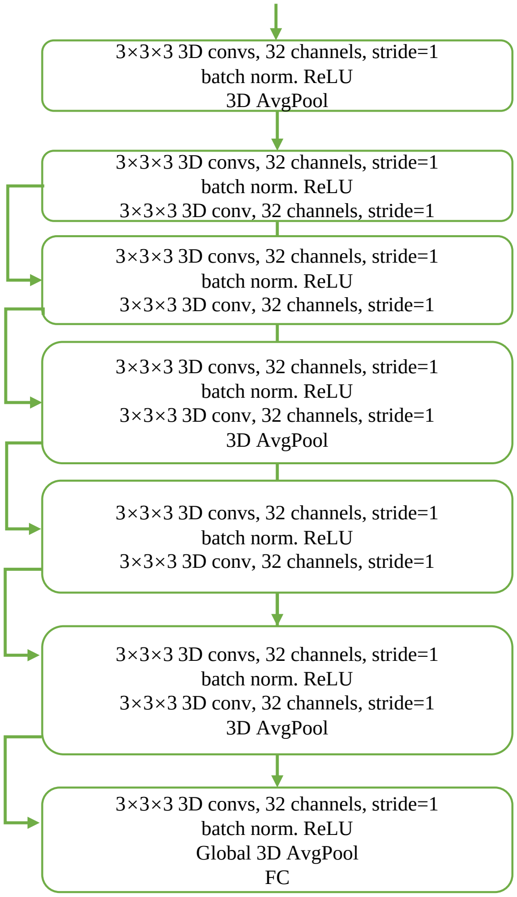
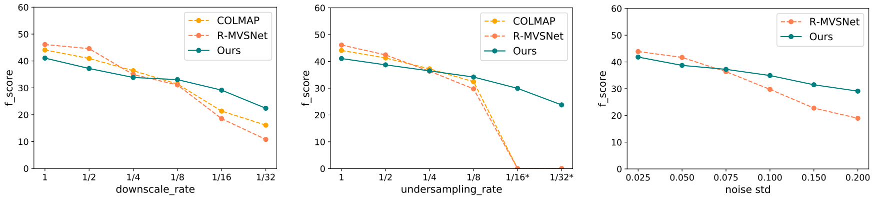

3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3D AvgPool
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3×3×3 3D conv, 32 channels, stride=1
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3×3×3 3D conv, 32 channels, stride=1
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3×3×3 3D conv, 32 channels, stride=1
3D AvgPool
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3×3×3 3D conv, 32 channels, stride=1
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
3×3×3 3D conv, 32 channels, stride=1
3D AvgPool
3×3×3 3D convs, 32 channels, stride=1
batch norm. ReLU
Global 3D AvgPool
FC

Fig. 3. 3D convolutional layers After P-CV.



(a) resolution downscale (b) temporary sub-sample (c) Gaussian noise

Fig. 4. Comparison with COLMAP[6] and R-MVSNet[10] with noisy input. Our work is less sensitive to initialization.

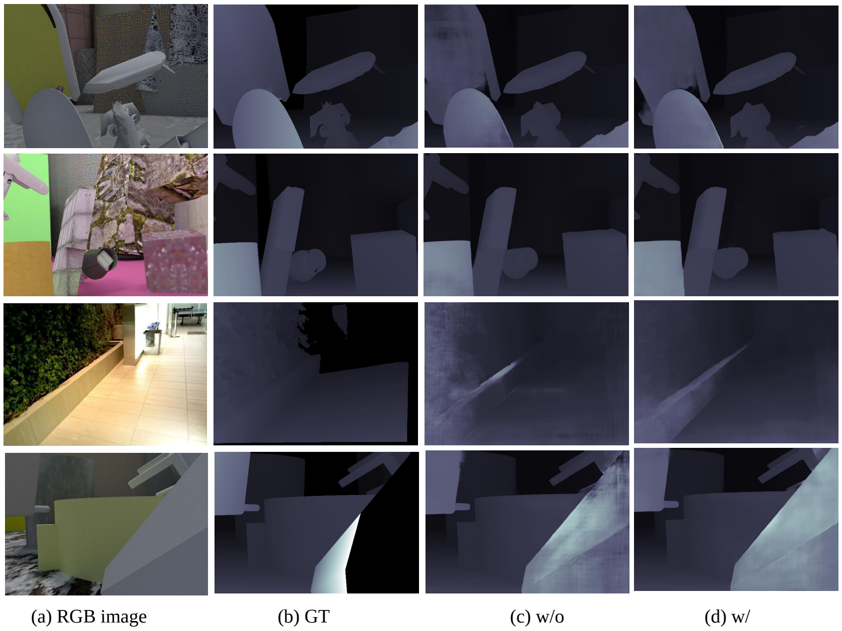


Fig. 5. Qualitative comparisons with and w/o scale invariant gradient loss.

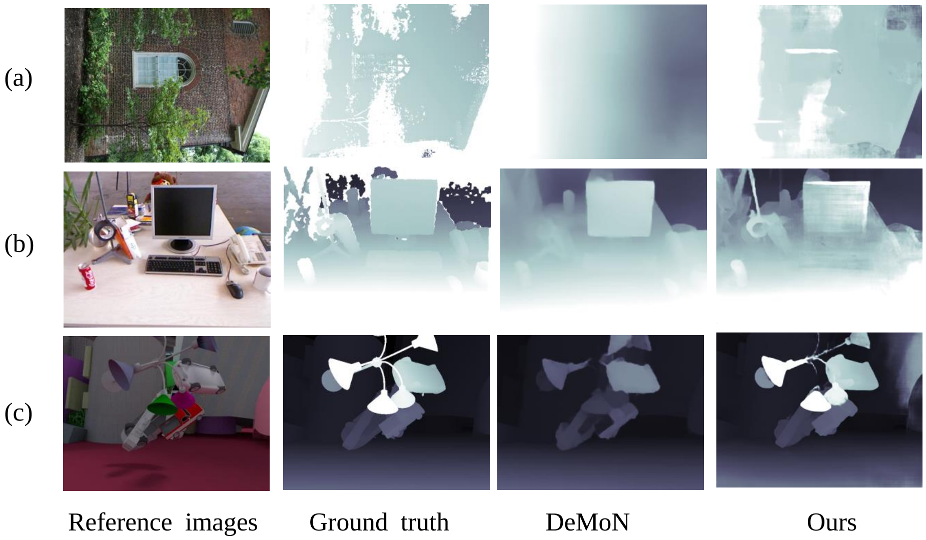


Fig. 6. More Qualitative comparisons with DeMoN [9] on DeMoN datasets.

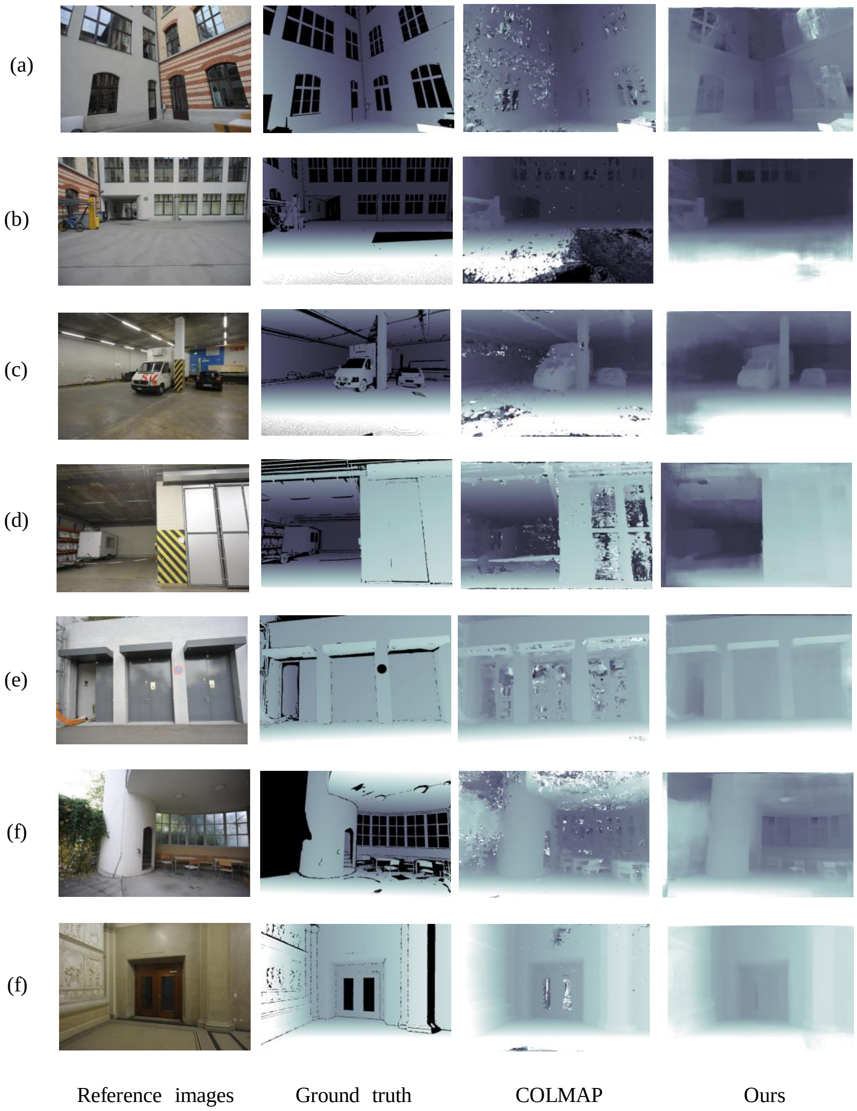


Fig. 7. Qualitative comparisons with COLMAP [6] on ETH3D datasets.

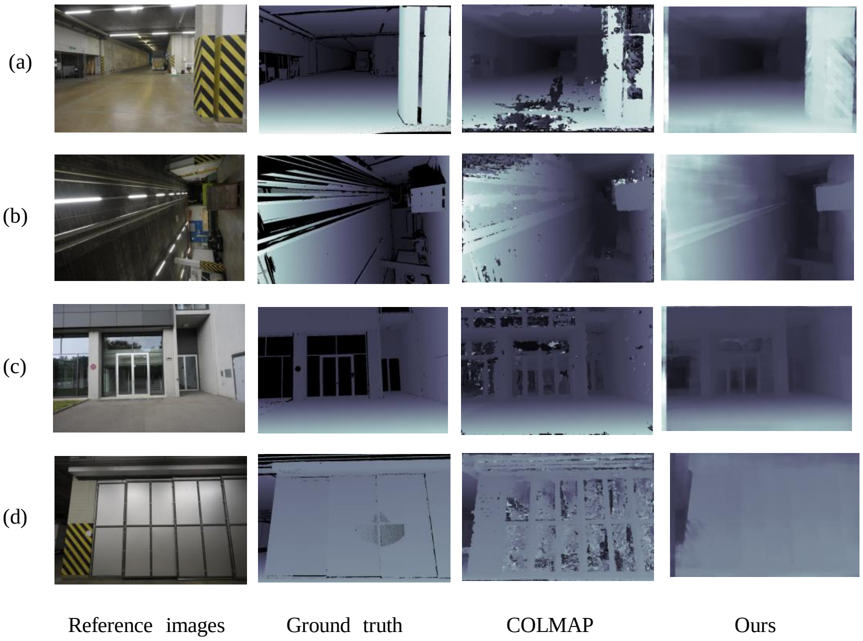


Fig. 8. Qualitative comparisons with COLMAP [6] on challenging materials. a) Textureless ground and wall. b) Poor illumination scene. c) Reflective and transparent glass wall. d) Reflective and textureless wall.