MVS2D: Efficient Multi-view Stereo via Attention-Driven 2D Convolutions Supplementary Materials

Zhenpei Yang[1,*] Zhile Ren[2] Qi Shan[2] Qixing Huang[1]
[1]The University of Texas at Austin [2]Apple

1. Details of Experiments

1.1. Model Details

We use ResNet-18(with the fully connected layer and pooling layer removed) as the building block of our networks $\mathcal{G}$ and $\mathcal{F}$. $\mathcal{F}$ additionally includes two up-sampling and convolution layer that output a feature map $F_0 \in \mathcal{R}^{\frac{h}{4} \times \frac{w}{4} \times c}$, where h and w are the input image's height and width respectively. F_0 is further feed into a small network with three convolutions and three de-convolutions to recover a depth probability volume $F_1 \in \mathcal{R}^{\frac{h}{4} \times \frac{h}{4} \times k'}$. F_1 is finally decoded into a depth map using the soft-argmin operation as in MVSNet.

1.2. Training Details

We use one DGX-V100 GPU with 32GB memory to train our models. We train for 15 epochs with a batch size 16, and reduce the learning rate by ten at epoch 10. The training settings are shared for all our variants. The input image size is 640×480. Our model predicts an image at a quarter of the original resolution and is then up-sampled (by nearest-neighbor interpolation) to the original resolution. The loss is applied to the quarter-resolution prediction.

To illustrate model convergence behavior, we save the checkpoint after each epoch and evaluate it on the ScanNet test set. We plot the training time vs. depth error(AbsRel) curve in Figure 1. We can see although **Ours-mono** finish the training procedure earlier, our methods that utilize multi-view cues, such as **Ours** and **Ours-robust**, performs significantly better. After a single epoch, **Ours** can achieve 0.091 on AbsRel, which is close to the final performance of many other multi-view methods. At the same time, **Ours** trains faster than **Ours-robust**, making it the most efficient approach in both training and inference. Please refer to Sec 2.1 for description of **Ours-nomask**.

1.3. FPS Comparison

We benchmark the FPS of each methods on the same machine with a Intel(R) Xeon(R) E5-2637 v4 @ 3.50GHz CPU and a GeForce GTX 1080 Ti GPU. We feed each methods a input image of size 640x480. Since the prediction of MVSNet/FastMVSNet and Ours are not at full resolution, we further up-sampled them to the input resolution using nearest-neighbor interpolation. The FPS results are averaged over 500 random inputs for each methods.

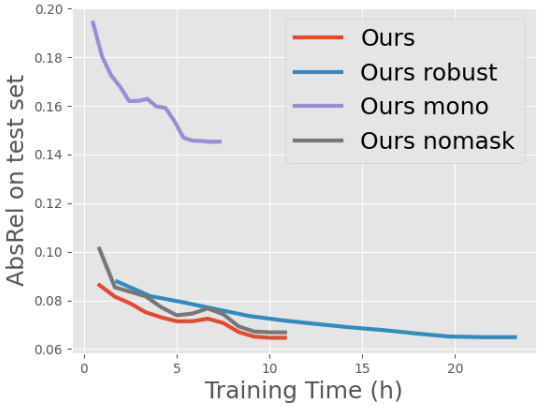

Figure 1. Training curves of our models. We save checkpoints after each epoch and evaluate the performance on the test set of ScanNet. **Ours** converge in less than 15 hours.

1.4. Pose Corruption

We use the following procedure to generate perturbations for input poses. Assume the ground truth relative pose is $T = [R \,|\, t]$ between the source image and a certain reference image. Firstly, we sample N points $\{p_k\}_{k=1}^N$ on corresponding ground-truth 3D point cloud of the source image. We use $N = 10$ in our experiments. Then, we project those N points into reference image using ground truth relative pose T and camera intrinsics $\mathcal{K}$ to get $\{\overline{p}_k\}_{k=1}^N$. We then perturb $\{\overline{p}_k\}_{k=1}^N$ by adding noise from a uniform distribution whose maximum value is 10 pixels. We solve a PnP problem [1] using $\{p_k\}_{k=1}^N$ and the perturbed pixels $\{\overline{p}_k\}_{k=1}^N$ to get the corrupted $\overline{T} = [\overline{R} \,|\, \overline{t}]$. We accept the perturbed $\overline{T}$ if the average pixel offset over the source image is less than 10 pixels. Otherwise, we set $\overline{T} = T$. We pre-compute all perturbations for all image pairs. Figure 2 shows the statistics of pose perturbations. Specifically, we plot the histogram of $\Delta R = \arccos(\frac{\mathrm{Tr}(\overline{R}R^{-1})-1}{2})$, and $\Delta t = \left\|\overline{t} - t\right\|_2$.

Method	AbsRel ↓	AbsDiff ↓	SqRel ↓	RMSE ↓	RMSELog ↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
COLMAP	0.384	0.843	1.26	1.480	0.500	0.482	0.663	0.840
DeMoN	0.311	1.330	19.970	2.607	0.247	0.641	0.902	0.967
DeepMVS	0.231	0.663	0.615	1.149	0.302	0.674	0.887	0.941
DPSNet	0.081	0.201	0.097	0.442	0.160	0.885	0.945	0.973
NAS	**0.068**	**0.168**	**0.056**	**0.375**	**0.142**	**0.905**	**0.964**	**0.988**
Ours-robust	0.119	0.307	0.128	0.517	0.186	0.844	0.934	0.970
Ours	0.117	0.297	0.116	0.515	0.183	0.846	0.944	0.977

Table 1. Depth evaluation results on the MVS dataset (trained on RGBD, SUN3D, and Scenes11). Please see Sec. 2.2 for discussion.

Method	AbsRel ↓	SqRel ↓	log10 ↓	RMSE ↓	RMSELog ↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
MVSNet	0.154	0.125	0.067	0.478	0.212	0.779	0.927	0.973
NAS	0.134	0.094	0.064	0.434	0.190	0.789	0.932	0.979
Ours	**0.109**	**0.060**	**0.049**	**0.331**	**0.149**	**0.878**	0.963	0.988
Ours-robust	0.112	**0.060**	0.051	0.346	0.152	0.862	**0.965**	**0.994**

Table 2. Depth evaluation results on the SUN3D dataset (trained on ScanNet). Please see Sec. 2.2 for discussion.

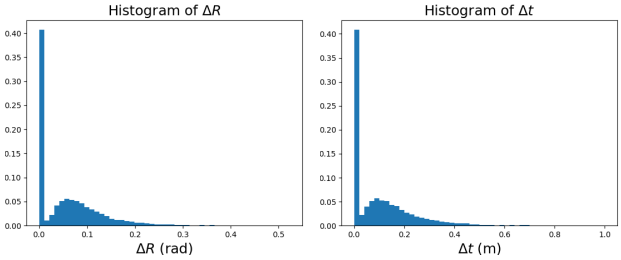

Figure 2. Pose corruption statistics. Left: histogram of rotation perturbation. Right: histogram of translation perturbation.

Method	AbsRel ↓	RMSE ↓	$\delta < 1.25 \uparrow$
Ours-nomask	0.067	0.164	0.956
Ours	**0.065**	**0.165**	**0.958**

Table 3. Ablation study on mask encoding. Please refers to Sec 2.1 for discussion.

Acc	$\frac{1}{N}\sum_i(\min_{p_j^* \in S^*} \|\|p_i - p_j^*\|\|_2)$
Comp	$\frac{1}{N}\sum_j(\min_{p_i \in S} \|\|p_i - p_j^*\|\|_2)$
Precision	$\frac{1}{N}\sum_i(\min_{p_j^* \in S^*} \|\|p_i - p_j^*\|\|_2 < \tau)$
Recall	$\frac{1}{N}\sum_j(\min_{p_i \in S} \|\|p_i - p_j^*\|\|_2 < \tau)$
F-score	$2\frac{\text{Precision}\times\text{Recall}}{(\text{Precision}+\text{Recall})}$

Table 4. The quantitative metrics for 3D reconstruction evaluation. p_i is one predicted 3D point, and p_j^* is one ground truth 3D point. N corresponds to all 3D points. We use $\tau = 0.05$m in our experiments

2. Additional Experimental Results

2.1. Ablation Study on Mask encoding

To study the benefits of mask encoding, we further experiment with **Ours-nomask** which removes the mask encoding in Eq 5. The results on ScanNet can be found in Table 3 and Figure 1. Remove mask encoding (**Ours-nomask**) leads to a slower convergence and worse results than **Ours**. Such behavior is reasonable since mask encoding provides an easy way for the network to distinguish the valid and invalid interpolation, thus facilitate the training.

Metric	Comp(m)↓	Acc(m)↓	Recall↑	Precision↑	F-score↑
NAS	0.154	0.092	0.395	0.546	0.435
MVSNet	0.250	**0.037**	0.396	**0.778**	0.470
DPSNet	**0.146**	0.081	0.411	0.557	0.447
FastMVSNet	0.216	0.041	0.342	0.747	0.443
Ours	0.152	0.062	**0.426**	0.631	**0.482**

Table 5. Quantitative results for dense 3D reconstruction. Our approach can achieve comparable performance to 3D convolution based methods.

2.2. Generalization Ability

We did two experiments to evaluate the generalization ability of all methods to unseen datasets. Following the experimental setups of DeMoN and NAS, we use the model trained on SUN3D/RGBD/Scenes11 and test on the MVS dataset, which is an outdoor dataset and the data distributions are very different from the training set. The results can be found in Table 1. Our methods perform better than COLMAP, DeMoN and DeepMVS, although they still fall behind DPSNet and NAS. Such a result is reasonable since our approach is better adapted to the training distribution, which will lead to performance drop on heavily out-of-distribution test data.

To further examine each method's performance on unseen datasets whose input data statistics are similar to those in the training sets, we further test models trained using ScanNet on SUN3D test sets (see Table 2). The input data of SUN3D ScanNet are all indoor scenes. We can see that our methods still perform favorably among other methods.

2.3. Quantitative Results on 3D Reconstruction

Following MVSNet, we measure the bi-directional distance between the predicted point-cloud to the ground-truth point cloud. The definitions of metrics can be found in Table 4 and the quantitative results can be found in Table

5. Although all other methods use expensive 3D convolutions, and some of them are designed for 3D reconstruction(MVSNet/FastMVSNet), our method can still achieve comparable performance.

2.4. More Qualitative Results

We show additional visualizations of depth predictions in Figure 3 & 4. Our method produces higher quality depth estimations compared to other MVS methods and performs better than one of the state-of-the-art single-view depth estimation method Bts. Additionally, we show more qualitative comparisons on 3D reconstruction in Figure 5. Our method generates comparable or even better visual results with other methods that require expensive 3D convolutions.

References

[1] Alex M Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001. 1

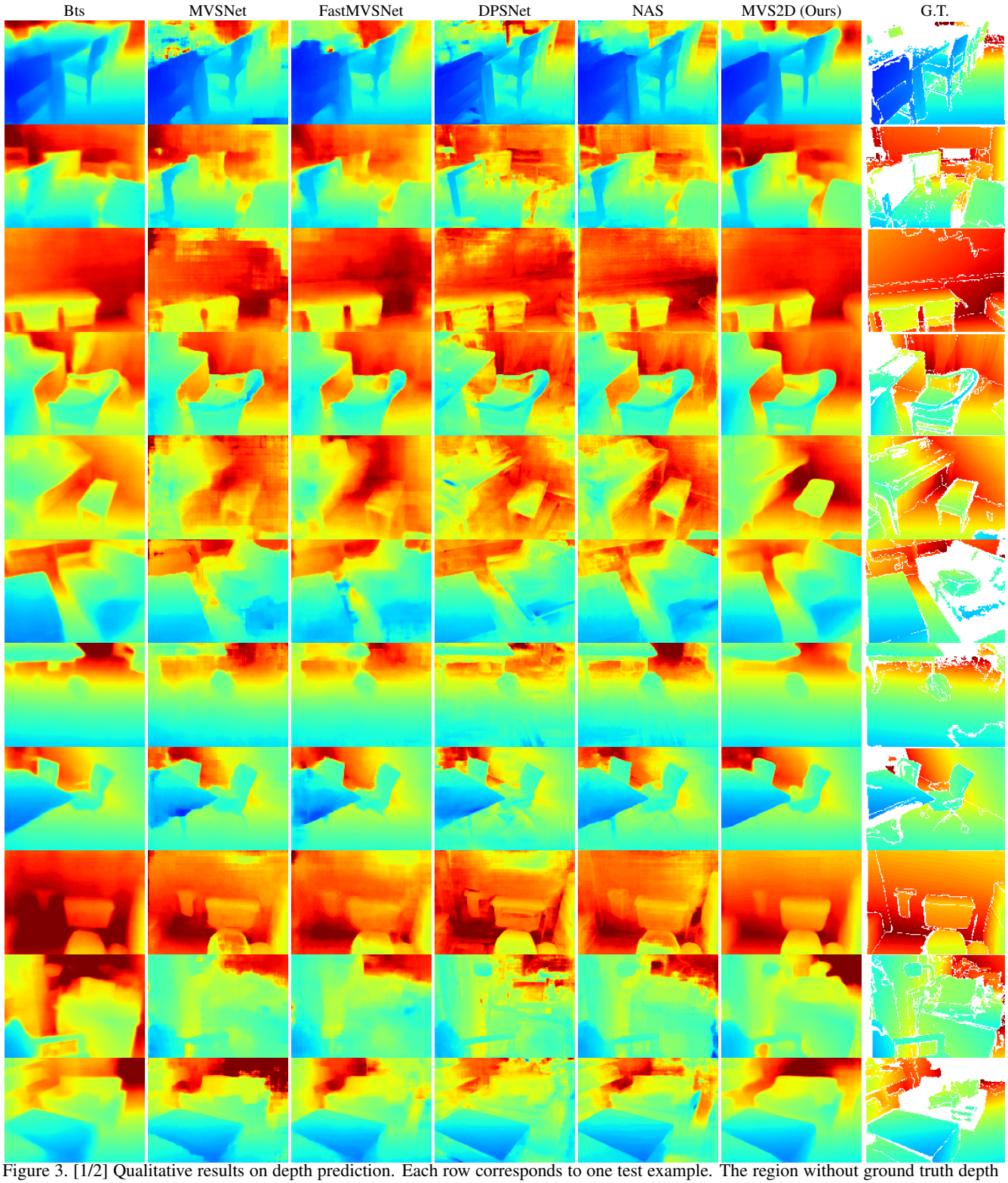

Figure 3. [1/2] Qualitative results on depth prediction. Each row corresponds to one test example. The region without ground truth depth labels is colored white in the last column. Our prediction outperforms both the single-view depth estimation method and other multi-view methods.

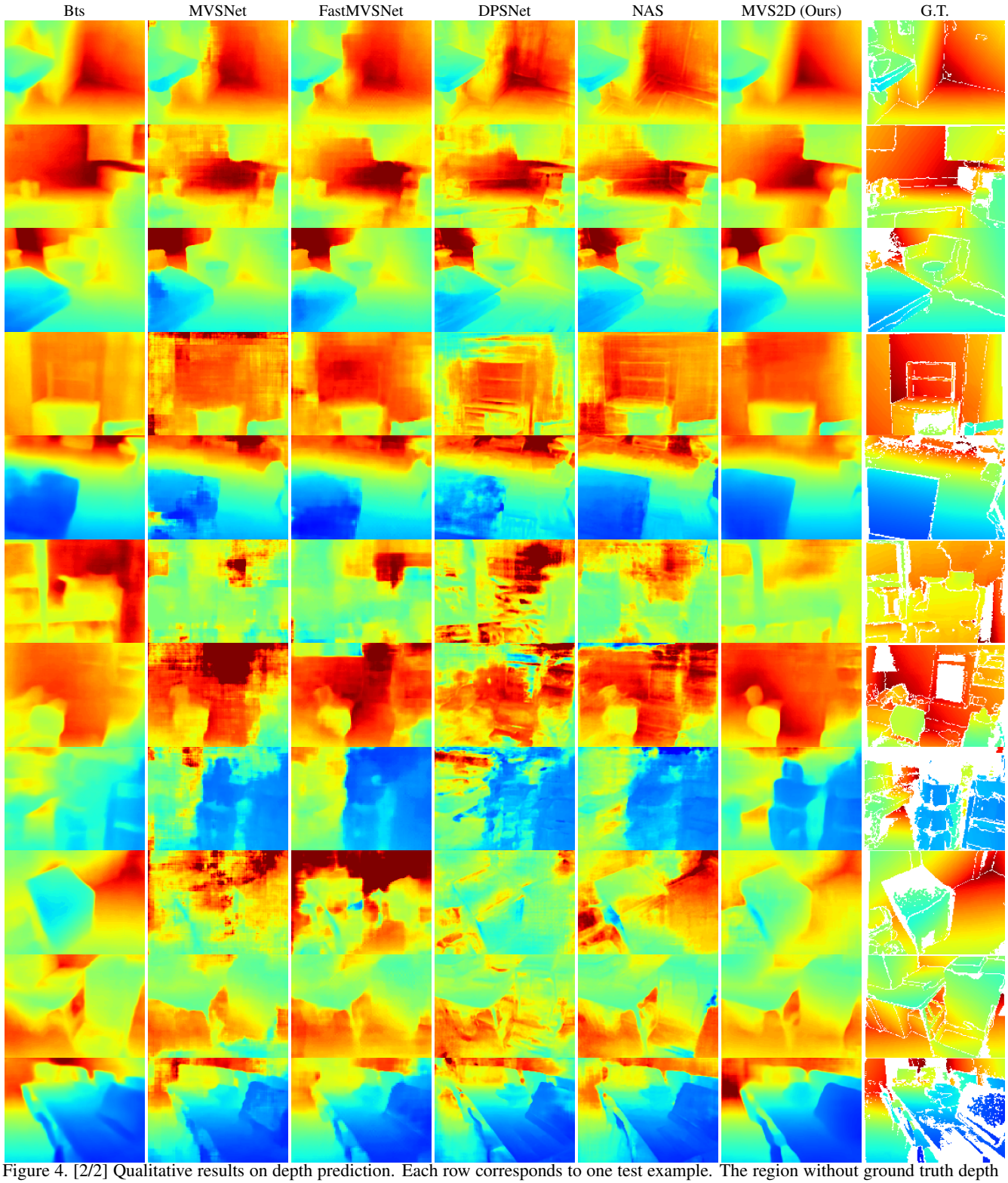

Figure 4. [2/2] Qualitative results on depth prediction. Each row corresponds to one test example. The region without ground truth depth labels is colored white in the last column. Our prediction outperforms both the single-view depth estimation method and other multi-view methods.

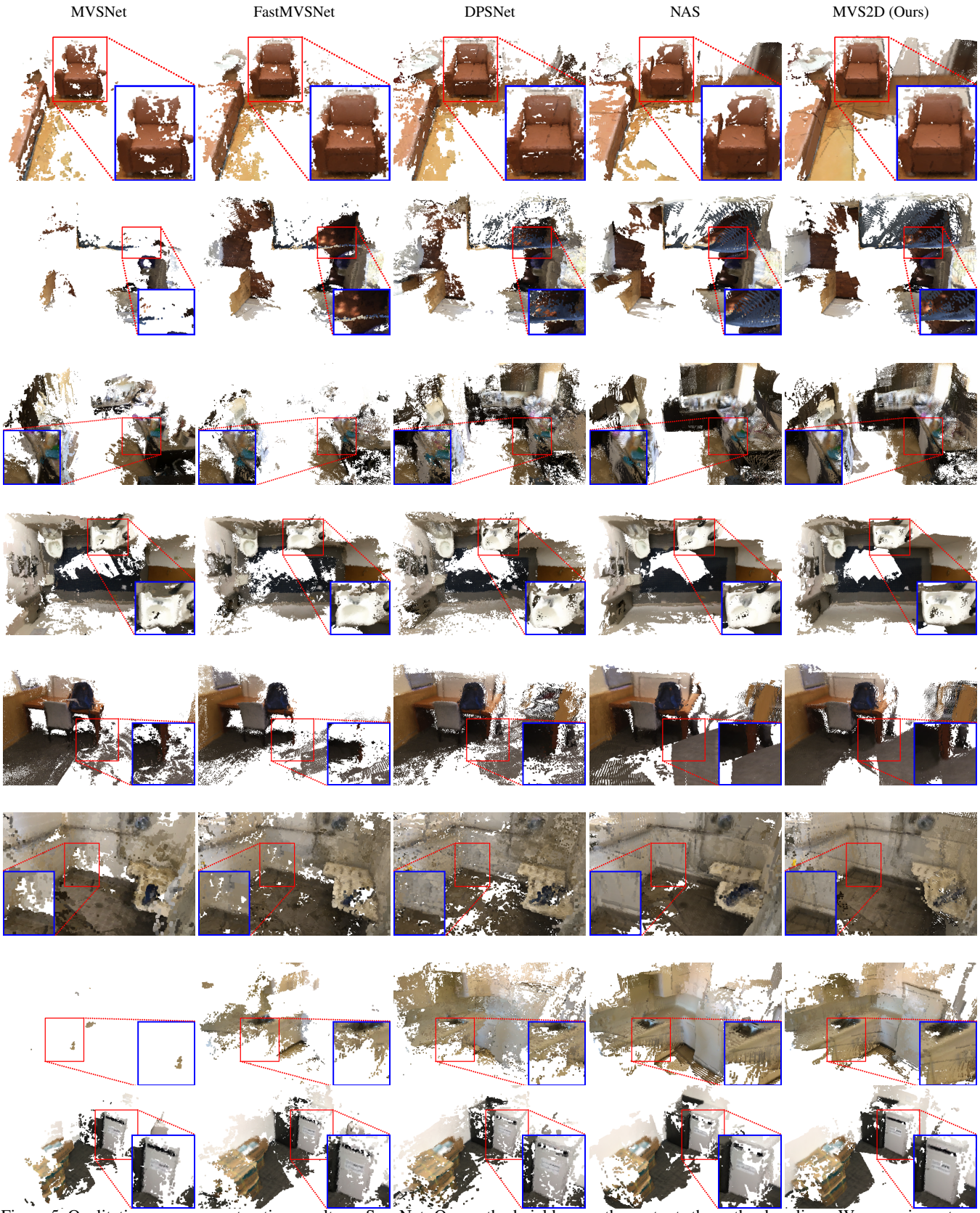

Figure 5. Qualitative scene reconstruction results on ScanNet. Our method yields smoother outputs than other baselines. We zoom in parts of a scene (red box) and show at the corner (blue box) to highlight the differences. Best viewed in PDF.