SimpleRecon: 3D Reconstruction Without 3D Convolutions Supplementary Material

Mohamed Sayed[2⋆] John Gibson[1] Jamie Watson[1,2]
Victor Prisacariu[1,3] Michael Firman[1] Clément Godard[4⋆]

[1]Niantic [2]UCL [3]University of Oxford [4]Google

S1 Overview

In this document we provide additional details about our method, and show additional qualitative results as well as ablations.

- **Section S2 (Prediction Causality and Runtime)**: In this section, we detail our keyframe selection process, and compare to other methods. In addition, we show how to break the online assumption of our keyframe selection process and note increased performance on reconstruction and depth metrics. We also discuss details of timing reconstruction methods.
- **Section S3 (Implementation Details)**: In this section, we detail our cost volume construction process, network structure, normal computation process, and depth fusion pipeline.
- **Section S4 (Ablation)**: In this section, we ablate different metadata inputs to our model.
- **Section S5 (Mesh Evaluation)**: In this section, we discuss our mesh evaluation procedure and compare different 3D evaluations from the literature.
- **Section S6 (Point Cloud Fusion)**: In this section, we compare our TSDF fusion + marching cubes method to direct point cloud fusion and show state-of-the-art performance using this method as well.
- **Section S7 (Qualitative Evaluation)**: In this section, we provide further qualitative results on 7-Scenes and ScanNet for depth metrics and 3D reconstruction metrics.
- **Section S8 (Metadata in Other MVS)**: In this section, we introduce metadata into a different MVS pipeline, CasMVSNet [7].

More 3D scene reconstruction results can be found in Figure S1, and more depth prediction results in Figures S2 and S3. In addition, we also provide a video comparing our method to DeepVideoMVS using online frames, and showing live reconstruction at 10Hz of a sequence captured using a mobile phone, also using online frames.

⋆ Work done while at Niantic, during Mohamed's internship.

	Online
ATLAS	No
TransformerFusion	No
NeuralRecon	Yes*
3DVNet	No
VoRTX	No
Ours	Yes

Table S1. Method's causality, i.e. whether or not a method needs to look into the future for a new prediction given how they are formulated. *NeuralRecon constructs geometry on a per chunk basis; each chunk is 9 keyframes long, so the method is only online as long as the latency penalty of waiting for those frames is paid.

S2 Prediction Causality and Runtime

Our depth prediction pipeline is causal, i.e. all our source frames come from past seen frames for online interactive reconstruction. This means our cost volume is fed source images that lag behind the current image for which we predict depth. All scores we report in depth estimation and mesh reconstruction are therefore predicated on our causal pipeline unless mentioned otherwise.

S2.1 Online Keyframe Selection

We use the same keyframe selection strategy outlined in DeepVideoMVS [4], including the same optimal rotation and translation distances quoted in the paper. We also limit the buffer size to the last 30 frames. We evaluate our depth maps and fuse them on these keyframes.

For when the buffer does not have enough frames, especially given our request for 7 measurement frames (for 8 image models), we accept any previous (non key) frame then pad the rest by repeating the available frames. We omit the evaluation for the first frame in a sequence where no previous frames exist.

S2.2 Non-Interactive Inference

If we assume that we have all available frames in a scene and wish to estimate the most accurate depths, then we can instead aim for the best selection of both past and future frames to feed our feature volume when predicting depth for a frame. With online keyframes, the cost volume will have empty regions for regions where past frames had not seen. With future frames, these empty regions are less severe, leading to a more accurate metric depth signal and a more accurate depth map. We denote when our model uses offline keyframes with "Offline" in Tables S2, S5, and S8. In this case, we modify DeepVideoMVS's keyframe buffer[4] to accept both future and past frames incrementally from the current frame.

S2.3 Inference

In the main paper at Sec. 4.3 and Table. 3, we outline the reconstruction time required to incorporate a new frame into a 3D reconstruction for current state-of-the-art methods.

All timings with the exception of TransformerFusion [2] were computed on an A100 GPU. Since TransformerFusion's code is not available at the time of submission (March 14$^{\text{th}}$, 2022), we estimate the method's latency using the authors' descriptive breakdown of the method's runtime on an RTX 3090.

Note that the timings we state are the least time required for each method to update its representation. This does not however guarantee that the methods would include the measurement accurately, since in most cases this is not representative of how the methods are meant to operate given how they are formulated. Most volumetric methods [14, 2, 20] are designed to output a prediction using all frames. NeuralRecon requires 9 keyframe inputs per chunk to accurately reconstruct a portion of the scene. This means that practically the overall latency may be much higher.

	ScanNet							
	Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓	logRMSE↓	$\delta < 1.05 \uparrow$	$\delta < 1.10 \uparrow$	$\delta < 1.25 \uparrow$
DPSNet (FT) [10]	0.1552	0.0795	0.0299	0.2307	0.1102	49.36	73.51	93.27
MVDepthNet (FT) [24]	0.1648	0.0848	0.0343	0.2446	0.1162	46.71	71.92	92.77
DELTAS [19]	0.1497	0.0786	0.0276	0.2210	0.1079	48.64	73.64	93.78
GPMVS (FT) [9]	0.1494	0.0757	0.0292	0.2287	0.1086	51.04	75.65	93.96
DeepVideoMVS, fusion† [4]	0.1186	0.0583	0.0190	0.1879	0.0868	60.20	83.66	96.76
Ours (no metadata)	0.0941	0.0467	0.0139	0.1544	0.0717	70.48	89.28	97.84
Ours	0.0885	0.0434	0.0125	0.1468	0.0673	73.16	90.57	98.09
DeepVideoMVS, pairnet† [4]	0.1431	0.0712	0.0253	0.2152	0.0999	51.92	77.24	94.99
DeepVideoMVS, fusion† [4]	0.1186	0.0583	0.0190	0.1879	0.0868	60.20	83.66	96.76
Ours* two frames	0.1225	0.0625	0.0196	0.1796	0.0856	57.58	81.42	96.18
Ours*	0.0867	0.0425	0.0123	0.1450	0.0665	73.86	90.98	98.18
ESTDepth [13]	0.1665	0.0917	0.0352	0.2392	-	44.75	69.17	91.51
IDNSolver [25]	0.1281	0.0666	0.0241	0.1998	0.0987	59.11	79.89	94.71
Ours every 20th	0.0910	0.0461	0.0133	0.1467	0.0698	71.18	89.30	97.80
Ours offline source frames	0.0829	0.0405	0.0114	0.1401	0.0640	75.65	91.96	98.32

Table S2. Depth evaluation on ScanNet For each metric, the best-performing method is marked in red, the second-best in orange, and the third-best in yellow. Results for previous methods were taken from [4], or evaluated for each method using their keyframes. **Ours*** indicates our model trained on the same 90/10 training split as of DVMVS instead of the official ScanNetv2 split. † two measurement frames. Trained on 90/10 split.

S3 Implementation Details

S3.1 Cost Volume

We use 64 depth planes in our feature volume, spaced uniformly in log space following FAL [6], between a minimum depth of 0.25m and a maximum depth of 5m. We perform the computation at quarter resolution, corresponding to the

7-Scenes								
	Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓	logRMSE↓	$\delta < 1.05 \uparrow$	$\delta < 1.10 \uparrow$	$\delta < 1.25 \uparrow$
DPSNet (FT) [10]	0.1966	0.1147	0.0550	0.2728	0.1511	38.81	61.91	87.07
MVDepthNet (FT) [24]	0.2009	0.1161	0.0623	0.2828	0.1513	38.81	62.92	87.70
DELTAS [19]	0.1915	0.1140	0.0490	0.2653	0.1470	36.36	60.22	88.13
GPMVS (FT) [9]	0.1739	0.1003	0.0462	0.2557	0.1403	42.71	67.63	90.32
DeepVideoMVS, pairnet† [4]	0.1861	0.1071	0.0498	0.2573	0.1396	39.66	65.03	89.33
DeepVideoMVS, fusion† [4]	0.1448	0.0828	0.0335	0.2254	0.1231	47.96	74.67	93.79
Ours (no metadata)	0.1105	0.0617	0.0175	0.1684	0.0891	57.30	82.73	97.02
Ours	0.1045	0.0575	0.0153	0.1596	0.0838	59.78	84.71	97.38
DeepVideoMVS, pairnet† [4]	0.1861	0.1071	0.0498	0.2573	0.1396	39.66	65.03	89.33
DeepVideoMVS, fusion† [4]	0.1448	0.0828	0.0335	0.2254	0.1231	47.96	74.67	93.79
Ours* two frames	0.1342	0.0766	0.0220	0.1855	0.1002	44.97	73.48	95.98
Ours*	0.1043	0.0574	0.0156	0.1599	0.0840	60.23	84.34	97.46
Ours, every 20th	0.1100	0.0619	0.0173	0.1657	0.0890	57.95	82.43	96.51

Table S3. Depth evaluation on 7-Scenes For each metric, the best-performing method is marked in red, the second-best in orange, and the third-best in yellow. Results for previous methods were taken from [4], or evaluated for each method using their keyframes. **Ours*** indicates our model trained on the same 90/10 training split as of DVMVS instead of the official ScanNetv2 split. † two measurement frames. Trained on 90/10 split.

resolution of the second block of the main encoder. We experimented with both lower and higher resolution but found it to be a good trade off. We also apply instance normalization [22] on the last layer of the matching network, as we found it to produce more stable results when training the baseline model using mixed precision. We initially also experimented with per-pixel normalization but found it to perform worse. We predict log-depth as it ensures the depth to be non-negative and prevents loss of precision when computing the main regression loss.

S3.2 Network Details

On acceptance we will release code for training and evaluation, models and precomputed results. Our decoder is similar to UNet++ [27]. Each block is a residual `BasicBlock` following [8] of 256, 128, 64, and 64 channels (from bottom to top). We make use of Leaky ReLU layers with a negative slope of 0.2 as our activation function.

The cost volume and image feature encoder follows the design of [4] with the same `BasicBlock` with the following number of channels per layer: 64, 128, 256, 384. We experimented with batch normalization but found it to create bubble-like artifacts, similar to the ones described in [11], in the predicted depths which are particularly visible in the normal-map. We therefore do not use batch normalization outside of the pretrained encoders. We actually suspect the artifacts visible in the normal maps from DVMVS [4] and ESTDepth [13] are caused by batch-normalization. For the matching feature encoder we use the first two blocks of ResNet18 [8].

Our MLP has channel sizes $[202, 128, 128, 1]$ for 8 views. The 202 input channels are composed of 16×8 visual channels, 1×7 visual feature dot products, 1×7 mask values, 3×8 rays, 1×7 ray angles, 1×8 depth values, 1×7 overall

pose distances, 1×7 pose rotation distance, and 1×7 pose translation distance. Views are ordered according to pose distance.

S3.3 Normal computation

Because of the discretization and patchiness of the ground truth depth, we first blur it using a 5x5 Gaussian blur then compute normal vectors using local image gradients on the backprojected depth following [16]. We apply the same processing to our predicted depths to obtain normals. This process is differentiable, allowing us to supervise our training with an additional loss on the normals, as discussed in the main paper.

S3.4 Depth Fusion Pipeline

We fuse depths using standard TSDF fusion with confidence weighting from InfiniTAM [15]. Similarly to [21], we have noticed that double walled meshes unfairly lead to higher recall scores due to extra available geometry, so we modify the voxel check during marching cubes in `scikit-image` [23] to only consider a voxel if all neighborhood voxels are larger than -1. This ensures that our mesh generated from the fused TSDF is single-walled.

We also report scores with our model fused using the Open3D library [26], which notably produces single-walled meshes from their marching cube implementation. When we fuse a predicted depth map from our model, we limit the maximum fused depth to 3m, similar to [21, 4].

Although we can fuse at a higher resolution with little penalty given voxel hashing, we use a voxel resolution of 4cm to match the other methods we evaluate against. We also report our model's score at 2.5cm and 3cm.

We fuse only the keyframes defined by the keyframe buffer from DeepVideoMVS [4].

S4 Ablation

In Table S4 we provide additional ablations of our method and comparisons to baselines.

Baseline — We ablate the information we make available to our cost reduction MLP in multiple steps. We first show that using *no* MLP and 16 feature channels, using the dot product of the image features performs better than using the mean absolute difference. Interestingly, using 64 feature channels instead of 16 slightly degrades accuracy while being significantly slower.

Losses — We also show that a combination of multi-scale geometric losses is important for performance. Removing either the normals loss or multi-view loss, as defined in the main paper, results in decreased accuracy and slower convergence times. We also show that our multiscale log-L1 loss leads to better

	Depth evaluation					Mesh eval	
	Abs Diff↓	Sq Rel↓	RMSE↓	$\delta < 1.05 \uparrow$	$\delta < 1.25 \uparrow$	Chamfer↓	F-score↑
Ours	**0.0885**	**0.0125**	**0.1468**	**73.16**	**98.09**	**5.81**	**67.1**
Ours baseline w/ dot product CV 16c	0.0941	0.0139	0.1544	70.48	97.84	6.29	64.2
Ours baseline w/ Absdiff CV 16c	0.1024	0.0155	0.1632	66.65	97.43	6.50	62.7
Ours baseline w/ dot product CV 64c	0.0944	0.0140	0.1548	70.49	97.84	6.08	65.4
Ours w/o Normals Loss	0.0896	0.0129	0.1495	72.76	98.04	5.87	66.7
Ours w/o Multiview Loss	0.0897	0.0127	0.1483	72.68	98.09	6.04	65.7
Ours, w/o Multiscale Gradient	0.0892	0.0127	0.1481	73.00	98.09	5.87	66.8
Ours w/ only Inv-L1 Loss	0.0933	0.0137	0.1551	71.47	97.88	6.09	65.5
Ours w/ ScaleInv instead of log-L1	0.0916	0.0126	0.1481	71.39	98.04	6.08	65.5
Ours w/ MnasNet Encoder	0.0929	0.0141	0.1560	72.11	97.79	6.05	66.0
Ours w/ FPN Matching Network	0.0870	0.0124	0.1455	73.89	98.14	5.74	67.4
DVMVS, fusion† [4]	0.1186	0.0190	0.1879	60.20	96.76	11.05	48.9
Ours 320×256	0.0916	0.0135	0.1525	71.99	97.94	5.88	66.5
Ours 320×256 + MnasNet	0.0947	0.0146	0.1587	71.24	97.68	5.92	66.3

Table S4. Ablation Evaluation Ablation evaluation on depth and reconstruction metrics using DVMVS keyframes for ScanNet. Scores for our full method (e.g. metadata-enriched MLP with frame ordering and 8 views) are shown in bold. † two measurement frames. Trained on 90/10 split.

performance over both the scale-invariant loss of Eigen et al. [5] and the inverse L1 loss used by DeepVideoMVS [4].

Encoders and Matching Networks — We evaluate our choice of image encoders, both for semantic feature extraction – where we use EfficientNetV2 S (ENv2S), as well as for matching, where we use the first 2 blocks of ResNet18 (R18). We first swap ENv2S for MnasNet as used in DeepVideoMVS [4]. Although it runs slightly faster than our default, and overall performance suffers, it outperforms DeepVideoMVS. On the other hand, using a multi-scale Feature Pyramid Network (FPN) [12] for matching, as in [4], improves our model performance but at the cost of a $\sim$50% increase in runtime. Our selected configuration, namely ENv2S + R18 provides a good trade-off between performance and speed, resulting in state-of-the-art results while maintaining fast inference time and lower memory use.

Comparison with DVMVS [4] We compare our main model to DVMVS. In both trials, we make use of the same input resolution of 320×256, and additionally for one trial we use the same image encoder as DVMVS (MnasNet). Our method outperforms [4] in all cases.

S5 Mesh Evaluation

For completeness, we include the mesh evaluation from ATLAS [14] in Tables S6 and S7. As noted by TransformerFusion [2] and 3DVNet [17], ATLAS evaluations on both meshes and depth are are flawed and produce inconsistent results. To support this we evaluate the ground truth meshes against themselves and note that errors are significantly far from zero and that precision, recall, and F-score

are likewise significantly far from one. We instead use the evaluation of [2] which produces sensible and consistent results on the GT meshes.

In addition, we include VolumeFusion's [3] reported results on the ScanNet test set in this section, compared to our model trained on the same data, as the authors were unable to share their code or predicted meshes in time for the submission deadline.

	Volumetric	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall ↑	F-Score ↑
RevisitingSI	No	14.29	16.19	15.24	0.346	0.293	0.314
MVDepthNet (FT)	No	12.94	8.34	10.64	0.443	0.487	0.460
GPMVS (FT)	No	12.90	8.02	10.46	0.453	0.510	0.477
ESTDepth	No	12.71	7.54	10.12	0.456	0.542	0.491
DPSNet (FT)	No	11.94	7.58	9.77	0.474	0.519	0.492
DELTAS	No	11.95	7.46	9.71	0.478	0.533	0.501
COLMAP	No	10.22	11.88	11.05	0.509	0.474	0.489
DeepVideoMVS	No	10.68	6.90	8.79	0.541	0.592	0.563
Neural Recon	Yes	5.09	9.13	7.11	0.630	0.612	0.619
ATLAS	Yes	7.16	7.61	7.38	0.675	0.605	0.636
3DVNet	Yes	7.72	6.73	7.22	0.655	0.596	0.621
TransformerFusion	Yes	5.52	8.27	6.89	0.728	0.600	0.655
VoRTX	Yes	4.31	7.23	5.77	0.767	0.651	0.703
Ours	No	5.53	6.09	5.81	0.686	0.658	0.671
Ours 2.5cm	No	5.58	5.66	5.62	0.675	0.687	0.679
Ours 3cm	No	5.54	5.80	5.67	0.683	0.678	0.679
Ours Open3d 4cm	No	5.30	6.16	5.73	0.701	0.641	0.668
Ours Offline frames	No	5.26	5.95	5.61	0.701	0.668	0.683
Ours (FPN Matching Net)	No	5.45	6.02	5.74	0.690	0.662	0.674
Ours FPN + offline frames	No	5.21	5.89	5.55	0.705	0.674	0.688
Groundtruth	-	0.15	0.80	0.48	1.000	1.000	1.000

Table S5. Mesh Evaluation. We use [2]'s evaluation. The Volumetric column designates whether a method is a volumetric 3D reconstruction method; other MVS methods that produce only depth maps were reconstructed using standard TSDF fusion. The final row shows a variant of our model which uses an FPN [12] matching encoder as in [4].

S6 Point Cloud Fusion

3DVNet [17] fuse their depths directly into a point cloud without using a TSDF volume representation. For fairness, we also use their pipeline for point cloud fusion from depth maps on our depths. We also downsample our point clouds to a 2cm resolution. We then use the ground-truth predictions masks provided in [2] when we evaluate the point clouds against the ground-truth in table S8.

Our model using online DeepVideoMVS keyframes outperforms 3DVNet in F-score. We see a further improvement in metrics if we ignore the online requirement - appropriate given point cloud fusion - and use offline source frames instead.

	Training split	Comp↓	Acc↓	Prec↑	Recall ↑	F-Score ↑
COLMAP [18]	Train	0.069	0.135	0.634	0.505	0.558
MVDepthNet [24]	Train	0.040	0.240	0.831	0.208	0.329
GPMVS [9]	Train	0.031	0.879	0.871	0.188	0.304
DPSNet [10]	Train	0.045	0.284	0.793	0.223	0.344
ATLAS [14]	Train	0.084	0.102	0.598	0.565	0.578
NeuralRecon [21]	Train	0.128	0.054	0.479	0.684	0.562
TransformerFusion [2]	Train	0.099	0.078	0.648	0.547	0.591
3DVNet [17]	Train	0.077	0.221	0.506	0.545	0.520
VoRTX [20]	Train	0.082	0.062	0.688	0.607	0.644
Ours	Train	0.078	0.065	0.641	0.581	0.608
Groundtruth	-	0.020	0.016	0.967	0.965	0.966
VolumeFusion [3]	Train+Val	0.125	0.038	-	-	0.508
Ours	Train+Val	0.076	0.063	0.653	0.590	0.618

Table S6. 3D Reconstruction Mesh Evaluation following ATLAS [14]

S7 Qualitative Evaluation

All visualizations made of our method are from the **Ours** model from the main text with online DeepVideoMVS keyframes.

S7.1 Mesh

We show additional comparisons between 3D reconstruction methods and different TSDF fusion methods in Table S5. Example reconstructions for our method and baselines is shown in Figure S1. We note that our model reconstructs more fine detail, as compared to the baselines.

S7.2 Depth

We also show comparisons between depth estimation methods and our method for the 7-Scenes dataset in Table S3 and for the ScanNet dataset in Table S2. We also note sharper results with additional detail in our predicted depth maps, as compared to baseline methods, in Figures S2 and S3. In these figures, we show normal maps computed from depths as described in Section S3. We also include a qualitative comparison of our estimated normals to those predicted directly by IDNSolver in Fig. S4

	Training split	Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓	$\delta < 1.25 \uparrow$	Comp ↑
COLMAP [18]	Train	.264	.137	.138	.502	83.4	.871
MVDepthNet [24]	Train	.191	.098	.061	.293	89.6	.928
GPMVS [9]	Train	.239	.130	.339	.472	90.6	.928
DPSNet [10]	Train	.158	.087	.035	.232	92.5	.928
ATLAS [14]	Train	.123	.065	.045	.251	93.6	.999
NeuralRecon [21]	Train	.106	.065	.031	.195	94.8	.909
TransformerFusion [2]	Train	.099	.065	.042	.205	93.4	.905
3DVNet [17]	Train	.107	.062	.042	.214	94.1	.984
VoRTX [20]	Train	.092	.058	.036	.199	93.8	.950
Ours	Train	.083	.046	.022	.173	95.4	.944
VolumeFusion [3]	Train+Val	.084	.049	.021	.164	-	-
Ours	Train+Val	.081	.045	.022	.172	95.5	.945
Groundtruth	-	.038	.023	.014	.115	96.9	93.7

Table S7. 3D Reconstruction Depth Evaluation following ATLAS [14] These are depth scores comparing mesh-rendered depths from a fused mesh to the groundtruth. All models were trained on the ScanNetv2 training set, except VolumeFusion [3] which was also trained on the validation split (approximately 25% more data).

	Volumetric	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall ↑	F-Score ↑
3DVNet	Yes	5.84	7.53	6.68	0.704	0.620	0.655
Ours	No	4.99	9.14	7.06	0.646	0.681	0.659
Ours, Offline source frames	No	4.76	8.94	6.85	0.664	0.670	0.672
Ours FPN, Offline source frames	No	4.71	8.84	6.78	0.669	0.692	0.676

Table S8. Point Cloud Evaluation. 3DVNet [17] evaluate their fused point clouds directly without exporting a mesh. To compare, we fuse our depths maps using their point cloud fusion pipeline and evaluate both methods using the ground-truth mask provided in [2].

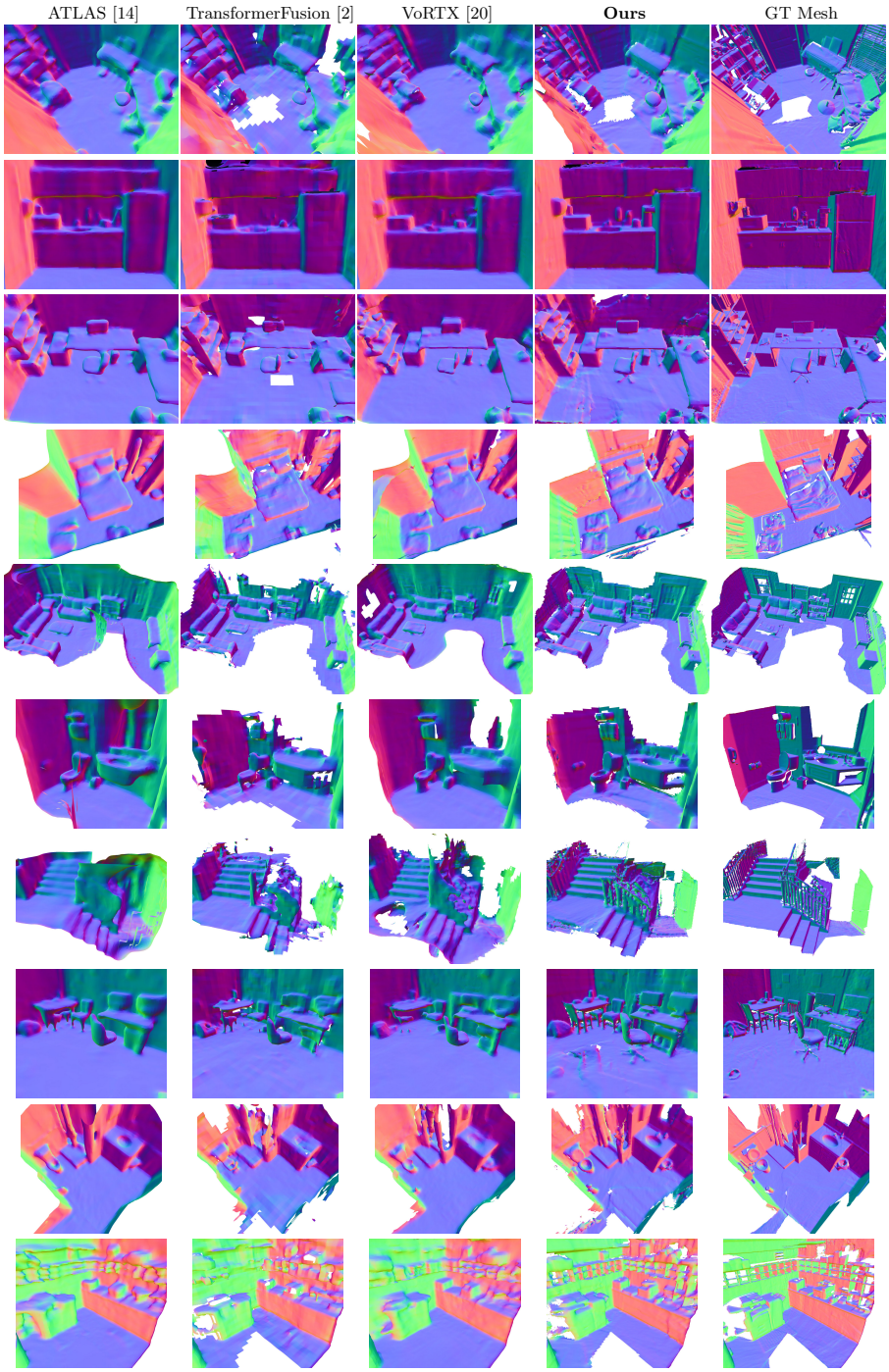

Fig. S1. Reconstructions on ScanNet Our model produces more detailed results compared to the baselines.

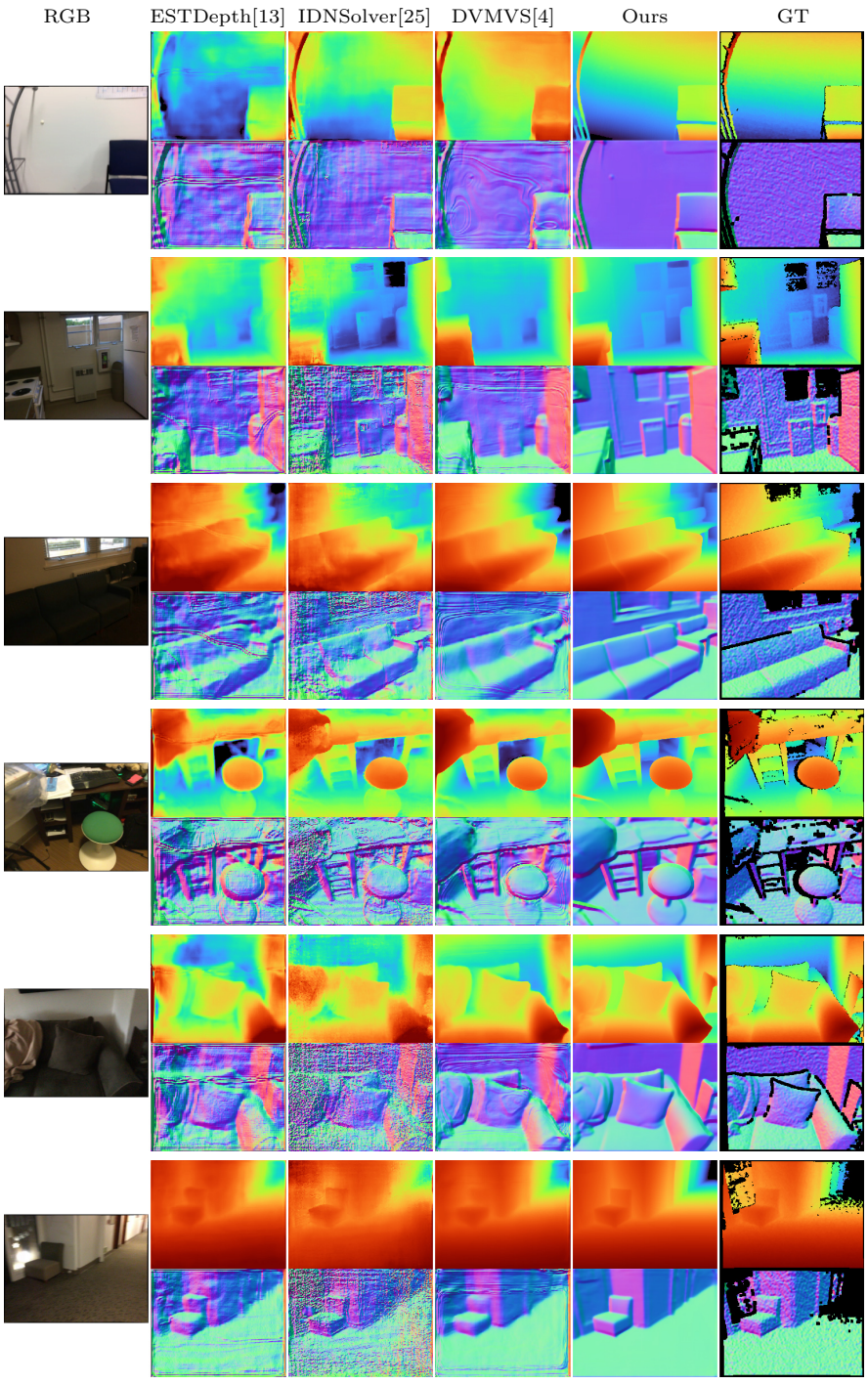

Fig. S2. Depth Results on ScanNet Our model produces sharper results compared to baselines.

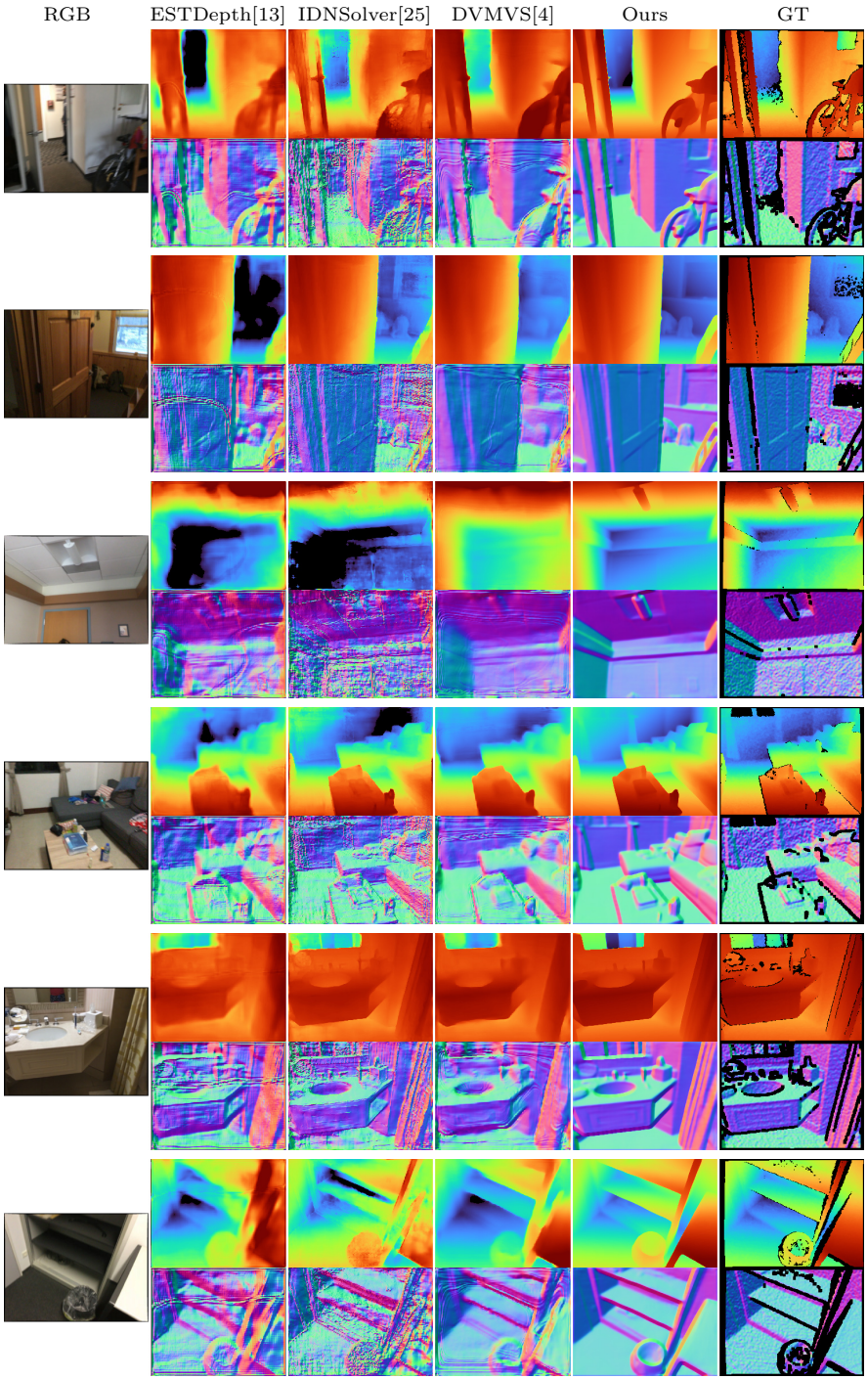

Fig. S3. Depth Results on ScanNet (continued)

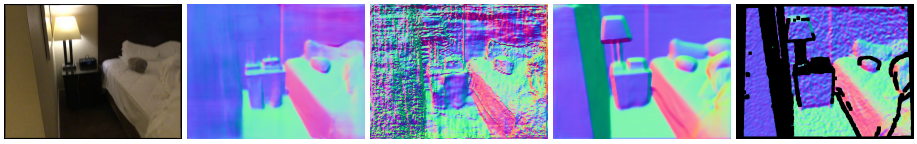

Fig. S4. Estimated and Predicted Normals on ScanNet Our model produces significantly sharper normals. We compute the estimated normals from depth for both IDNSolver [25]'s and our depth prediction. We also visualize IDNSolver's predicted normals alongside.

S8 Metadata in Other MVS

We introduce metadata into CasMVSNet [7]. We expand the dimensions of the cost volume to also include the metadata features described in the paper. We pass this data in an ordered fashion through an MLP at every scale. We evaluate both the base CasMVSNet network and a version with metadata in Table S9. We show that metadata improves both reconstruction and especially depth estimation accuracy on the DTU dataset [1]. Both models are trained with four reference views alongside the source.

	Abs Diff↓	Abs Rel↓	Sq Rel↓	$\delta < 1.05 \uparrow$	$\delta < 1.10 \uparrow$	Acc↓	Comp↓	Overall↓
CasMVSNet	11.05	0.0143	2.153	92.80	95.95	**0.347**	0.341	0.344
CasMVSNet + **Metadata**	**10.58**	**0.0133**	**2.067**	**93.64**	**96.15**	0.368	**0.314**	**0.341**

Table S9. Metadata in other MVS Pipelines We compare a variant of CasMVS-Net [7] modified to use metadata in its cost volume. We show an improvement in reconstruction and depth on the DTU dataset [1].

References

1. Aanæs, H., Jensen, R.R., Vogiatzis, G., Tola, E., Dahl, A.B.: Large-scale data for multiple-view stereopsis. International Journal of Computer Vision pp. 1–16 (2016)
2. Bozic, A., Palafox, P., Thies, J., Dai, A., Nießner, M.: TransformerFusion: Monocular RGB scene reconstruction using transformers. NeurIPS (2021)
3. Choe, J., Im, S., Rameau, F., Kang, M., Kweon, I.S.: VolumeFusion: Deep depth fusion for 3D scene reconstruction. In: ICCV (2021)
4. Duzceker, A., Galliani, S., Vogel, C., Speciale, P., Dusmanu, M., Pollefeys, M.: Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In: CVPR (2021)
5. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NeurIPS (2014)
6. GonzalezBello, J.L., Kim, M.: Forget about the lidar: Self-supervised depth estimators with med probability volumes. Advances in Neural Information Processing Systems **33**, 12626–12637 (2020)
7. Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., Tan, P.: Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: CVPR (2020)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
9. Hou, Y., Kannala, J., Solin, A.: Multi-view stereo by temporal nonparametric fusion. In: ICCV (2019)
10. Im, S., Jeon, H.G., Lin, S., Kweon, I.S.: DPSNet: End-to-end deep plane sweep stereo. ICLR (2019)
11. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8110–8119 (2020)
12. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
13. Long, X., Liu, L., Li, W., Theobalt, C., Wang, W.: Multi-view depth estimation using epipolar spatio-temporal networks. In: CVPR (2021)
14. Murez, Z., van As, T., Bartolozzi, J., Sinha, A., Badrinarayanan, V., Rabinovich, A.: Atlas: End-to-end 3D scene reconstruction from posed images. In: ECCV (2020)
15. Prisacariu, V.A., Kähler, O., Golodetz, S., Sapienza, M., Cavallari, T., Torr, P.H., Murray, D.W.: Infinitam v3: A framework for large-scale 3D reconstruction with loop closure. arXiv preprint arXiv:1708.00783 (2017)
16. Riba, E., Mishkin, D., Ponsa, D., Rublee, E., Bradski, G.: Kornia: an open source differentiable computer vision library for pytorch. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3674–3683 (2020)
17. Rich, A., Stier, N., Sen, P., Höllerer, T.: 3dvnet: Multi-view depth prediction and volumetric refinement. In: International Conference on 3D Vision (3DV) (2021)
18. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: CVPR (2016)
19. Sinha, A., Murez, Z., Bartolozzi, J., Badrinarayanan, V., Rabinovich, A.: Deltas: Depth estimation by learning triangulation and densification of sparse points. In: ECCV (2020)
20. Stier, N., Rich, A., Sen, P., Höllerer, T.: Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In: International Conference on 3D Vision (3DV) (2021)

21. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: NeuralRecon: Real-time coherent 3D reconstruction from monocular video. In: CVPR (2021)
22. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
23. Van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T.: scikit-image: image processing in python. PeerJ **2**, e453 (2014)
24. Wang, K., Shen, S.: MVDepthNet: Real-time multiview depth estimation neural network. In: 3DV (2018)
25. Zhao, W., Liu, S., Wei, Y., Guo, H., Liu, Y.J.: A confidence-based iterative solver of depths and surface normals for deep multi-view stereo. In: ICCV. pp. 6168–6177 (October 2021)
26. Zhou, Q.Y., Park, J., Koltun, V.: Open3d: A modern library for 3d data processing. arXiv preprint arXiv:1801.09847 (2018)
27. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: UNet++: A nested U-Net architecture for medical image segmentation. In: Deep learning in medical image analysis and multimodal learning for clinical decision support (2018)