DoubleTake:
Geometry Guided Depth Estimation

Mohamed Sayed[1] Filippo Aleotti[1] Jamie Watson[1,2]
Zawar Qureshi[1] Guillermo Garcia-Hernando[1] Gabriel Brostow[1,2]
Sara Vicente[1] Michael Firman[1]

[1]Niantic [2]UCL
https://nianticlabs.github.io/doubletake/

Abstract. Estimating depth from a sequence of posed RGB images is a fundamental computer vision task, with applications in augmented reality, path planning etc. Prior work typically makes use of previous frames in a multi view stereo framework, relying on matching textures in a local neighborhood. In contrast, our model leverages historical predictions by giving the latest 3D geometry data as an extra input to our network. This self-generated geometric hint can encode information from areas of the scene not covered by the keyframes and it is more regularized when compared to individual predicted depth maps for previous frames. We introduce a *Hint MLP* which combines cost volume features with a hint of the prior geometry, rendered as a depth map from the current camera location, together with a measure of the confidence in the prior geometry. We demonstrate that our method, which can run at interactive speeds, achieves state-of-the-art estimates of depth and 3D scene reconstruction in both offline and incremental evaluation scenarios.

1 Introduction

High quality depth estimations have been shown to be effective for virtual occlusions [65], path planning [14], object avoidance and a variety of augmented reality (AR) effects [14]. For all these applications, we need per-frame depths to be generated at interactive speeds. While the best quality depth maps can be achieved via offline methods *e.g.* [27, 73], these are not suitable for interactive applications. For interactive use, depths are typically estimated via a multi-view stereo (MVS) approach, where a network takes as input a target frame together with nearby posed and matchable source frames to build a cost volume [30] and estimate a depth map as the output of a neural network [5, 7, 52, 75, 76].

Matchability needs the texture on the visible surfaces to be similar and visible in both source and target frames, though this is often not possible *e.g.* due to occlusions or distance from the surface. We make the case that each time we estimate depth for a location, it is unlikely that this is the first time we have seen this place. We might revisit a location in the short term, for example turning to look at a kitchen appliance we were last looking at a few seconds ago. Alternatively we may revisit a location after a period of time has passed, for

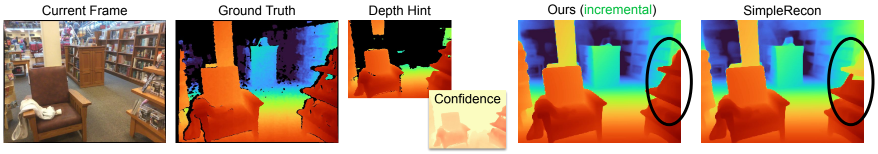


Fig. 1: A depth hint and confidence are rendered from a prior estimate of geometry and given as input to our method. This enables it to correctly predict the depth for ambiguous parts of the scene.

example entering a room we last visited a week prior. In this work, we argue that these short and long-term geometric 'snapshots' of the scene can be a crucial source of information which can be used to gain higher quality instantaneous depths. We introduce a system that maintains a low-cost global representation of 3D geometry as a truncated signed distance function (TSDF). When predicting depth for a new frame, we render a depth map from the TSDF at the current camera pose and give that as input to our depth estimator (see Figure 1).

Our experiments show that, surprisingly, just naively passing in a depth render of a global mesh into a depth estimation network fails to bring performance gains. Instead, we introduce a carefully-designed architecture to make use of such prior geometry 'hints' as input, and show that the same network can make use of these hints both from the short and long term. We validate our approach via extensive experiments and ablations, and show that our method achieves a new state-of-the-art on depth estimation and reconstruction, as validated on the challenging ScanNetV2, 7Scenes, and 3RScan datasets. Our contributions are:

1. A system which can use prior estimations of geometry to improve instantaneous depth estimates. If a hint isn't available for a frame, our system gracefully falls back to the performance of a strong baseline model. We also extend this to include *historical*, long-term hints into our framework, enabling the use of hints when revisiting a location after a period of time.
2. As geometry is constructed in real-time, certain parts of it may still be unreliable. We therefore propose an architecture which incorporates this geometry alongside a measure of its confidence. This combined information is integrated with multi-view stereo cost volume data using a 'Hint MLP'.
3. We introduce a new evaluation protocol that pays more careful consideration to the limitations of the ground-truth mesh, and evaluate ours and several baselines with this new evaluation.

2 Related Work

Multi-view Stereo. Depth estimation from posed monocular videos is a long-standing problem in computer vision. Traditional patch-based solutions [17, 53] have been outpaced by learning strategies [25,72], where a cost volume is built by warping features from multiple source frames at different depth hypotheses [10].

The resulting 4D volume can be regularized by 3D convolutions [72], which are expensive in terms of memory. To tackle this, pyramidal approaches [20, 71] compute multiple cost volumes at different resolutions with a narrow set of hypotheses, and depths computed at lower scales can provide geometric priors to following computation [3, 77]. SimpleRecon [52] shows that accurate depths can be achieved only via 2D convolutions, using cost volumes enriched by additional *metadata*, while FineRecon [59] further improves the results with a resolution-agnostic 3D training loss and a depth guidance strategy. These multi-view stereo (MVS) approaches have several failure cases: (i) unmatched surfaces, *e.g.* empty areas in the cost volume where no source frames have a matching view; (ii) depth planes are sparser at greater depths; (iii) greater depths require wider baselines and many more keyframes. Our proposed approach helps overcome these problems by injecting prior information into the network based on previously computed geometry data.

Use of Input Sequences. For example, [8,36,58] warp previous predictions or features and use these as input to the current prediction, in some cases together with a confidence estimate [58]. We also use an estimate of confidence, but our geometry priors are from a global reconstruction, enabling priors from long before in time. Instead of regularizing the cost volume, CER-MVS [41] iteratively updates a disparity field using recurrent units. Recurrent layers [15] and Gaussian Processes [24] have also been used to ingest sequence data. Our work uses sequences but we do not rely on generic layers to encode the prior information. Instead, we directly extract prior knowledge from a 3D reconstruction of the scene to guide the model. An alternative use of sequences is to update predicted depth maps or network weights at *test time* by optimizing image reconstruction losses *e.g.* [4, 6, 27, 34, 38, 42, 56] . These methods require the entire sequence to be seen ahead of time, and aren't suitable for interactive applications.

Priors for Depth Estimation. When available, additional knowledge about the scene might be injected to boost depth estimation methods. In autonomous driving [19,43], for instance, LiDAR scans are often completed into a dense depth map [9, 39, 40, 68]. Similarly, sparse point clouds from Simultaneous Localization And Mapping (SLAM) algorithms can be used as input to improve depth estimators [69, 70]. However, these priors might be unevenly scattered or unavailable. To tackle these issues, sparsity-agnostic methods [11, 63] and networks capable of both predicting and completing depth maps [22] have been proposed. mvg-MVS [47] uses depth data from a depth sensor to directly modulate values in the cost volume. In contrast, our approach uses prior geometry generated from our own estimates as an additional input to an MLP alongside the cost volume. Khan *et al.* [32] convert per-frame depth estimates to temporally consistent depth via a globally fused point cloud which is input into their final depth network. Their focus is primarily temporal consistency; we compare with [32] and find our approach achieves higher quality depths. Like us, 3DVNet [50] maintain a global reconstruction to improve depth maps. However, their reconstruction and depths are iteratively refined, precluding online use. We also employ priors to improve current depth predictions. Unlike these prior works, we argue that previous predictions of the model itself, if properly managed, are strong hints to improve current estimates for online depth estimation. Furthermore, we also show that our model is robust even in the absence of such information.

3D Scene Reconstruction. Traditional methods for 3D reconstruction estimate depths (*e.g.* from MVS), fuse them into a TSDF and extract a mesh via Marching Cubes [37]. In contrast, feedforward volumetric methods [18,45,61,79] directly estimate the volume occupancy (usually encoded as a TSDF), often leveraging expensive 3D CNNs. Implicit representations [35, 46, 74] can generate high quality reconstructions via test-time optimization. These solutions are computationally expensive as they typically require a *per-scene* optimization. Neural Radiance Fields (NeRF) [44] and Gaussian Splatting [31] enable novel view synthesis, but their meshes tend to be noisy and additional post-processing steps [21, 48] or a different scene representation [33] might be necessary to generate consistent 3D representations. Extensions to NeRFs use structure-from-motion point clouds and depth estimation to improve view synthesis [13, 51, 64] or reconstructions [16,74]. These offline approaches require the whole video to be seen ahead of time and are typically slow, making them unsuitable for interactive applications. Conversely, our method efficiently estimates depth maps using a 2D MVS model boosted by its own predictions, unlocking accurate reconstructions and depth estimates with a low overhead in computation.

3 Method

Our method takes as input a live online sequence of RGB images along with their poses and intrinsics. The goal of our method is to predict a depth map for the current frame given all frames that have come before it. To train our method we assume we have access to a ground truth depth map for each training image.

Similar to previous MVS methods *e.g.* [52, 72], we rely on features matched between the current frame and a few recent source frames to build a feature-based cost volume. Our key contribution is to show how a 3D representation of the scene, constructed from previous depth estimates, can be used as additional input to our network to improve depth predictions for each new frame. In particular, this geometric 'hint' complements the likelihood depth estimated via multi-view-stereo on the current and source frames. We incorporate this extra source of information by combining it with the information provided by the cost volume using a "Hint MLP." An overview of our method can be seen in Fig. 2.

3.1 MVS background

Our method builds on SimpleRecon [52], a state-of-the-art depth estimation method from posed images. Similar to other MVS methods *e.g.* [10,30,72], SimpleRecon constructs a cost volume using multiple viewpoints of the same scene. To compute the cost volume, SimpleRecon starts by extracting feature maps for the current image as well as associated source frames. Extracted source-frame features are then warped to the current camera at each hypothesis depth plane

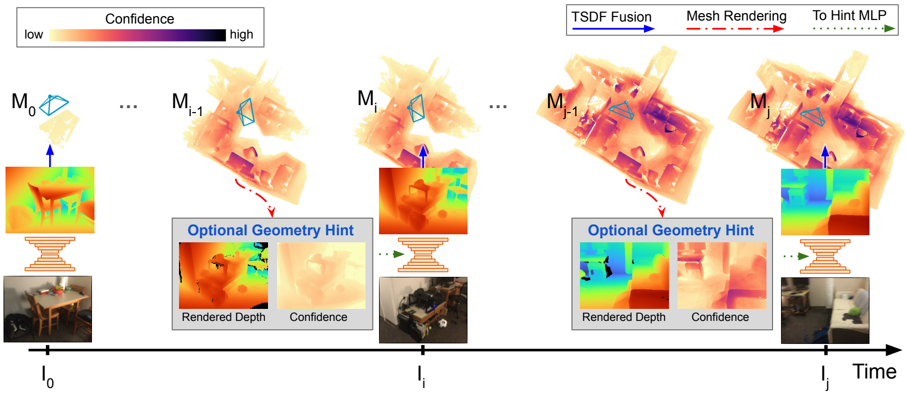


Fig. 2: Method overview. Like other MVS methods, we take as input a sequence of RGB frames. Optionally, our network can additionally ingest a depth map rendered from the 3D representation of the scene built up so far, encoded in a TSDF volume. Alongside the rendered depth map, we include an estimate of how confident the global geometry is at each point, visualized as vertex colors (purple is higher confidence) on the incremental mesh here. The depth predicted by our model is fused back into the TSDF to update the 3D geometry incrementally. When no such rendered depth map is available, our network gracefully falls back to our baseline model's performance.

and compared against features for the current frame. This comparison, together with additional *metadata*, produces a $C \times D \times H \times W$, feature volume with depth likelihood estimates at each location, where C is the channel dimension, D the number of depth planes and $H \times W$ the dimensions of the feature maps. A *matching MLP* is applied to each aggregated feature vector in parallel to obtain the final cost volume with dimensions $D \times H \times W$. The advantage of this reduction step is that the cost volume can then be processed using 2D convolutions, in contrast to methods that bypass this step and require expensive 3D convolutions. A follow-up monocular prior head regularizes the volume using 2D CNNs and a depth decoder finally produces the output depth map.

3.2 Hint MLP

In addition to using source and target frames, our method allows the input of a *rendered depth* and a *confidence map* from a prior geometric representation of the scene. Note that this geometry could be from the distant past, but it could also be from the recent past *e.g.* as constructed incrementally from previous frames we have recently seen. The rendered depth map is a 2D image of depth values, where each pixel represents the rendered depth from the current camera position to our prior estimate of geometry along that camera ray. The confidence map is a 2D image where each pixel gives an estimate of how certain we are that the prior depth estimate is correct at that pixel, where values of 1.0 indicate that we have extremely high confidence in the prior depth at this pixel and values of 0.0 mean a very low confidence in the rendered geometry at this pixel.

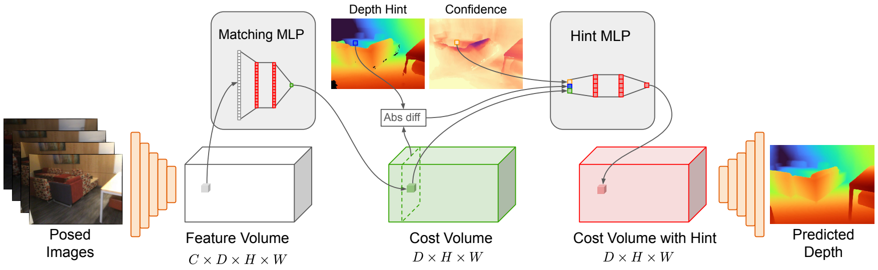


Fig. 3: Method detail. Our feature volume is reduced to a cost volume via a *matching MLP*. Our *Hint MLP* then combines the multi-view-stereo cost volume with an estimate of previously predicted geometry. For every location in the cost volume, the *Hint MLP* takes as input (i) the visual matching score, (ii) the geometry hint, formed as the absolute difference between the rendered depth hint and the depth plane at that cost volume position, and (iii) an estimate of the confidence of the hint at that pixel.

Following [52] we create a feature volume and apply a *matching MLP* to each combined feature vector individually to give a matching score, creating a cost volume with dimensions $D \times H \times W$. Inspired by this, we use an additional *Hint MLP* to combine the information from the cost volume with the rendered depth and confidence images. Like the matching MLP, the Hint MLP is applied in parallel at each spatial location and depth plane in the feature volume. Our Hint MLP takes as input a vector with three elements: (i) the matching score from the cost volume, (ii) the geometry hint, defined as the absolute difference between the TSDF rendered depth and the current depth plane and (iii) the confidence value at that pixel. For pixels where there isn't a rendered depth value, for example because they haven't been seen before, we set the confidence to 0 and the geometry hint to -1. This is done both at training and test time.

See Fig. 3 for an illustration of how the *Hint MLP* is incorporated into the network architecture. We validate our choice of how to combine the different sources of information in our ablations in Section 4.3.

3.3 Maintaining persistent 3D geometry

In our work, we encode our persistent geometry as a truncated signed distance function (TSDF). A TSDF is a volumetric representation of the scene that stores at each voxel the distance (truncated and signed) from the voxel to the nearest estimated 3D surface, together with a scalar confidence value. Our TSDF is built up using fused depth maps produced using our method at previous keyframes.

Rendered depth and rendered confidence. For each new frame, given the corresponding intrinsics and extrinsics, we render a depth map and a confidence map from the TSDF from the point of view of the camera. This is achieved using marching cubes [37] to obtain a mesh, followed by a mesh rendering step. We render the mesh using PyTorch3D [49], which runs on the GPU. The marching cubes step takes 9.4ms and the mesh rendering step takes 9.2ms on average (see

Section 4.2 for more timings). We obtain the confidence map by sampling the voxel confidence channel in the TSDF with coordinates from the backprojected rendered depth; this takes less than 1ms.

Updating the persistent 3D geometry. Once we have estimated a depth map for a new frame we aggregate it into the current geometry estimate. Our fusion follows InfiniTAM [28, 29]. When updating the TSDF values we also update the confidence measures. The confidence for a pixel with depth d is $c = 0.025 \times \max((1-\hat{d})^2, 0.25)$, where $\hat{d} = (d - \text{depth}_{\text{min}})/(\text{depth}_{\text{max}} - \text{depth}_{\text{min}})$ is the depth value normalized to $[0, 1]$. This gives pixels with higher depths a lower confidence values, while the clamp to a minimum value of 0.25 ensures that even distant predictions have some confidence assigned.

Motivating our representations. We use TSDFs and meshes for our persistent geometry as they are ubiquitous, lightweight in terms of memory and runtime, and easy and quick to update and extract geometry from. This is unlike other geometry representations *e.g.* NeRFs or other implicit methods, where the geometry is more 'baked-in' and harder to update and extract.

3.4 Sources of prior geometry

Our prior geometry can come from different sources.

- When we see an environment for the very first time, *i.e.* we haven't previously scanned this environment, the TSDF is built up incrementally as the camera moves in the new environment. When the camera views a completely unseen part of the environment, the network doesn't have access to a geometry hint and must rely only on the cost volume. Over time, more and more of the environment will be present in the geometry hints.
- Alternatively, when we **revisit** a location we have previously seen, we can load previously generated geometry to use for geometry hints. In this scenario, we assume the current camera position is in the same coordinate system as the loaded geometry. However, we may find that some items have moved since the original geometry was created. Our experiments, with the 3RScan dataset [66], evaluate precisely this scenario.
- Finally, if we want the best possible reconstruction from our feedforward model we can run **offline**. Here, we use a *two-pass* approach: We first run the full RGB sequence through our system *without* using hints to build an initial TSDF. We then run the full sequence a second time, where the TSDF generated on the first pass is used as the hint for every frame.

3.5 Training data generation

At test time, our *Hint MLP* is likely to encounter a range of scenarios: at the start of sequences, for example, there may not be a geometry hint available. At the other extreme, after a whole sequence has been observed, subsequent predictions may have access to a hint for almost every input pixel. To ensure our network is robust to different test-time scenarios, we train with different randomly selected types of hints.

For 50% of the training items we don't provide a geometry hint, instead feeding -1 as the rendered depth and 0 as the confidence; in this scenario, the network must learn to rely only on the matching cost. For the other 50% of the training items the network has access both to the matching cost and to the rendered depth and rendered confidence. We generate a dataset of geometry hints by fusing depths from [52] into a training-time TSDF. 50% of the time that we give a training-time hint it is rendered from the full and complete TSDF, assuming the entire scene has been previously fused, and 50% a partial TSDF fusing only frames up to the current training frame.

3.6 Implementation details

We train our model (and all our ablations) with a batch size of 16 on each of two Nvidia A100 GPUs. We use a learning rate of $1e{-}4$ that we drop by a factor of 10 at 70000 and 80000 steps. We train with an input dimension of 512×384, and an output dimension of 256×192. During training, we color augment RGB images. Our network follows that of [52], with the addition of our 'Hint MLP' as described above. This network uses an EfficientNetV2 S [62] encoder for the monocular image prior and the first two blocks of a ResNet18 [23] encoder for generating matching features for the cost volume. The decoder follows UNet++ [78]. Our *Hint MLP* is small with two hidden layers, each comprising 12 neurons; this adds 2ms to our runtime. Full architecture details are given in the supplementary material. Our training losses directly follow [52] and their all applied at the final network output. Our *Hint MLP* does not require any intermediate supervision. We select source frames for the cost volume with the strategy and hyperparameters from [15]. For evaluation of online methods, all source frames come from the past in a sequence. For evaluation of offline methods, source frames may come from anywhere in a sequence. For both 'ours', SimpleRecon [52] and DeepVideoMVS [15] we use the same TSDF fusion method. We fuse depth maps to a maximum depth of 3.5m into a TSDF volume of 2cm resolution. For the FineRecon [59] baseline we use a 1cm voxel resolution.

4 Evaluation

We evaluate with three challenging 3D datasets, all acquired with a handheld RGBD sensor. We train and evaluate on **ScanNetV2 [12]**, which comprises 1,201 training scans, 312 validation scans, and 100 testing scans of indoor scenes. We additionally evaluate our ScanNetV2 models on **7-Scenes [55]** without fine-tuning, following *e.g.* [52] and with the test split from [15]. We also evaluate on **3RScan [66]**. This dataset captures the same environment in multiple separate scans, between which objects' positions have changed. This tests our ability to use scans captured in the past as 'hints' for instantaneous depth estimates.

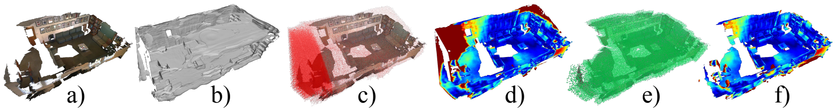


Fig. 4: We introduce a more accurate mesh evaluation. (a) shows the ground truth mesh, which contains many holes. (b) shows an example predicted mesh, here from [59]. This is punished for being too complete, as [2]'s visibility mask (c) extends beyond the ground truth, giving high *Acc* error on their prediction (d). Our new masking (e) is tighter to the ground truth mesh, giving a more meaningful error (f).

	ScanNetV2					**7Scenes**				
	Abs Diff↓	Abs Rel↓	Sq Rel↓	$\delta < 1.05$↑	$\delta < 1.25$↑	Abs Diff↓	Abs Rel↓	Sq Rel↓	$\delta < 1.05$↑	$\delta < 1.25$↑
DPSNet [26]	.1552	.0795	.0299	49.36	93.27	.1966	.1147	.0550	38.81	87.07
MVDepthNet [67]	.1648	.0848	.0343	46.71	92.77	.2009	.1161	.0623	38.81	87.70
DELTAS [57]	.1497	.0786	.0276	48.64	93.78	.1915	.1140	.0490	36.36	88.13
GPMVS [24]	.1494	.0757	.0292	51.04	93.96	.1739	.1003	.0462	42.71	90.32
DeepVideoMVS, fusion [15]*	.1186	.0583	.0190	60.20	96.76	.1448	.0828	.0335	47.96	93.79
SimpleRecon [52]	.0873	.0430	.0128	74.12	98.05	.1045	.0575	.0156	60.12	97.33
Ours (incremental)	.0767	.0369	.0112	79.94	98.35	.0985	.0534	.0156	64.76	97.01
Ours (no hint)	.0870	.0428	.0128	74.35	98.02	.1082	.0593	.0171	59.44	96.97
Ours (offline)	.0627	.0306	.0092	86.46	98.62	.0858	.0466	.0133	71.74	97.61

Table 1: Depth evaluation on ScanNetV2 and 7Scenes. Unless stated otherwise, predictions are computed incrementally, without access to future frames or frames from previous scans of the scene. We highlight the best, second-best and third-best methods per metric. Previous scores are from [15,52]. * [15] was boosted by computing depths using three inference frames instead of two; they also use a custom 90/10 split.

4.1 Mesh evaluation metrics

We follow existing works [2,45] and report reconstruction metrics based on point to point distances on sampled point clouds. ScanNetV2 ground truth meshes are not complete, as the scan sequences don't have full coverage. This means methods which overpredict geometry not in the ground truth get unfairly punished. TransformerFusion [2] use a visibility mask to trim predictions when computing prediction to ground-truth distances. However, these masks are over-sized, and include large areas of geometry which aren't present in the ground truth mesh. We propose new visibility volumes using rendered depth maps of the ground-truth meshes, which much tighter fit the ground truth meshes. Figure 4 shows the difference between masks from [2] and our new proposed masks. More are shown in the supplementary. We show results on both sets of masks in Tables 4 and 2, and we will release our updated masks for reproducibility.

4.2 Depth and reconstruction performance

Depth estimation. Table 1 shows our depth estimation performance on ScanNetV2 and 7Scenes in both incremental and offline modes (see Section 3.4 for details of these modes). Our incremental method outperforms all baseline methods across most scores. Our offline approach outperforms all competitors including our incremental method.

		Volumetric	Acc↓	Comp↓	Chamfer↓	Prec↑	Recall ↑	F-Score ↑
Online	DeepVideoMVS [15]	No	6.49	6.97	6.73	.568	.595	.579
	NeuralRecon [61]	Yes	7.31	10.81	9.06	.453	.592	.511
	SimpleRecon [52] (online)	No	5.56	5.02	5.29	.631	.712	.668
	Ours (incremental)	No	4.92	5.49	5.20	.685	.701	.692
Offline	ATLAS [45]	Yes	5.59	7.52	6.55	.671	.610	.637
	TransformerFusion [2]	Yes	4.68	8.27	6.48	.698	.600	.644
	VoRTX [60]	Yes	4.38	7.23	5.80	.726	.651	.685
	SimpleRecon [52] (offline) †	No	5.25	4.86	5.05	.654	.725	.687
	FineRecon [59]	Yes	4.92	5.06	4.99	.687	.737	.710
	Ours (offline)	No	4.15	4.85	4.50	.743	.734	.738

Table 2: Mesh Evaluation on ScanNetV2 with new visibility masks. We evaluate methods using the evaluation from [2] using our visibility masks that more accurately represent the ground-truth mesh (Sec. 4.1). The 'Volumetric' category indicates whether a technique involves volumetric 3D reconstruction. Following [52], for other Multi-View Stereo (MVS) methods that generate solely depth maps, we applied conventional TSDF fusion for reconstruction. **Chamfer** distance is the mean of accuracy and completion and **F-Score** is the harmonic mean of precision and recall. † Like Ours (offline), SimpleRecon [52] (offline) uses source frames from both past and future.

Reconstruction. For meshing results we separate all methods into 'online' and 'offline': 'Online' methods can produce meshes instantaneously, while 'offline' methods are designed with the assumption that all frames are available at once. Tables 4 and 2 show reconstruction performance both 'online' and 'offline'. Our incremental model sets a new state-of-the-art for online reconstruction performance, validating our approach to depth and reconstruction. See Figures 5 and 7 for qualitative comparisons with previous state-of-the-art. Ours (offline) obtains even better results; we present these in the second section of the table, where we compare against other offline reconstruction approaches *e.g.* [2,59,60]. Our approach is first or second best in most metrics. Figure 8 shows the benefit running ours offline can bring.

Long-term hints on 3RScan. Our system can use long-term hints *e.g.* where we **revisit** an environment we have first observed at some time in the past. We use the 3RScan dataset [66] for this scenario, as this includes multiple scans of the same location at different times. We use the TSDF generated from a previous visit as the hint for the current, online depth estimates. See Table 5 for our results, where we see the benefit of using long-term hints vs baselines which don't have access to hints, or an incremental version of our method. The table also includes figures showing how our method can gracefully cope with the situation where the scene has changed since the hint TSDF was generated.

Timings. Our online, incremental system takes just 76.6ms to compute depth for a single frame, as measured on an Nvidia A100. The majority of this time (52.8ms) is running the forward pass of the depth network, while the remainder is generating the hint and updating the TSDF. A version of our model with smaller networks than [52] runs at 50.4ms per frame, see J in Table 3. This

		Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓	$\delta < 1.05$↑	$\delta < 1.25$↑
A	**Ours** without Hint MLP (as in SimpleRecon [52])	.0873	.0430	.0128	.1483	74.12	98.05
B	**Ours** w/ online hint, without confidence	.0863	.0392	.0129	.1529	77.81	98.02
C	**Ours** w/ hint & confidence to cost volume encoder	.0890	.0438	.0130	.1506	73.39	97.95
D	**Ours** w/ warped depth as hint	.0889	.0436	.0130	.1492	73.26	98.06
E	**Ours** w/ hint-based variable depth planes [70]	.0821	.0405	.0121	.1432	76.67	98.19
F	SimpleRecon [52] w/ TOCD [32]	.0880	.0437	.0134	.1505	74.19	97.83
G	**Ours** w/ hint cost volume modulation [47]	.0773	.0372	.0112	.1381	79.52	98.34
H	**Ours** w/ single MLP for matching and hints	.0773	.0371	.0112	.1381	79.56	98.35
I	**Ours** (no hint)	.0870	.0428	.0128	.1477	74.35	98.02
J	**Ours** (incremental, fast)	.0826	.0400	.0125	.1473	77.14	98.05
K	**Ours** (incremental)	.0767	.0369	.0112	.1377	79.94	98.35
L	**Ours** offline w/ hint cost volume modulation [47]	.0651	.0320	.0094	.1242	84.92	98.61
M	**Ours** (offline)	.0627	0306	.0092	.1225	86.46	98.62

Table 3: Incremental ablation evaluation. Scores are depth metrics on ScanNetV2. See the text for descriptions of these variants, and the supplementary for full metrics.

		Volumetric	Acc↓	Comp↓	**Chamfer**↓	Prec↑	Recall↑	**F-Score**↑
Online	DPSNet [26]	No	11.94	7.58	9.77	.474	.519	.492
	DELTAS [57]	No	11.95	7.46	9.71	.478	.533	.501
	DeepVideoMVS [15]	No	5.84	6.97	6.41	.639	.595	.615
	NeuralRecon [61]	Yes	5.09	9.13	7.11	.630	.612	.619
	SimpleRecon [52] (online)	No	5.72	5.02	5.37	.682	.712	.696
	Ours (incremental)	No	4.70	5.49	5.09	.730	.701	.714
Offline	COLMAP [53, 54]	No	10.22	11.88	11.05	.509	.474	.489
	ATLAS [45]	Yes	7.11	7.52	7.31	.679	.610	.640
	3DVNet [50]	Yes	6.73	7.72	7.22	.655	.596	.621
	TransformerFusion [2]	Yes	5.52	8.27	6.89	.729	.600	.655
	VoRTX [60]	Yes	4.31	7.23	5.77	.767	.651	.703
	SimpleRecon [52] (offline)†	No	5.37	4.86	5.12	.702	.725	.712
	FineRecon [59]	Yes	5.25	5.06	5.16	.779	.737	.756
	Ours (offline)	No	4.96	4.85	4.90	.752	.734	.742

Table 4: Mesh Evaluation on ScanNetV2. Here we use the evaluation and visibility masks from [2]. Note that VoRTX wins on accuracy, but its very sparse predictions give a poor completion score.

compares to *e.g.* 58ms per frame update of [52] or 90ms of [61]. Offline, we are faster than close competitor FineRecon [59], taking on average 13.8s per scene vs. [59]'s 48.1s. See Supplementary Material for a full breakdown of timings.

4.3 Ablations and variants

Table 3 shows ablations and variants of our approach in incremental mode. Row A doesn't use a geometry hint or a Hint MLP at all in training or evaluation, so it is functionally equivalent to [52]. Ablations C and H replace our Hint MLP with alternatives methods; both of these score worse than **ours** (K). Our use of a separate Hint MLP has an additional advantage that we can cache the cost volume output for the second pass of offline mode. In B the network has no access to confidences, so it may incorrectly rely on under-construction

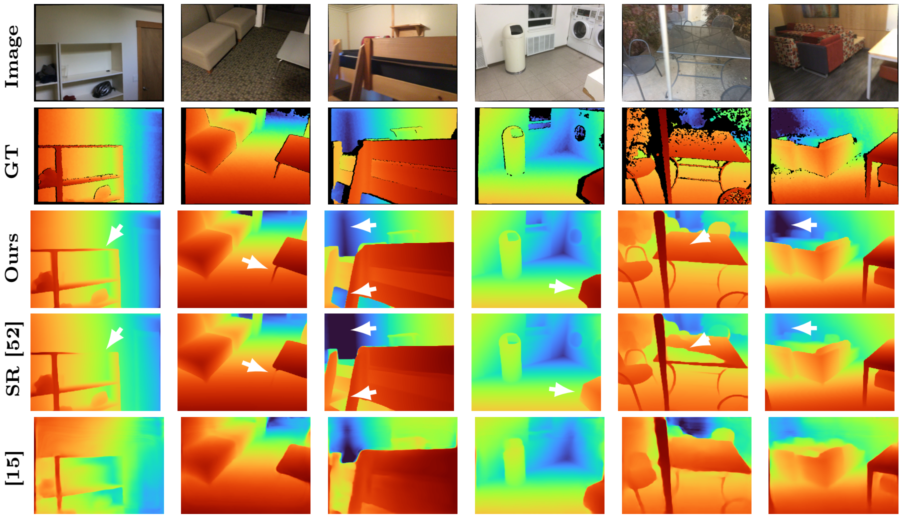


Fig. 5: Qualitative depth results on ScanNetV2. All methods are run incrementally, where we run at interactive speeds only with access to previous frames. We compare with [52] and [15], the two closest-performing baselines. Our depth maps are more accurate with better small details (*e.g.* top) and overall geometry (*e.g.* bottom).

subpar geometry. We then vary the format the geometry hint takes: ablation D forward-warps previous depth predictions to the current viewpoint. E and F are implementations in the spirit of previous publications [32, 70] which allow depth as input to an MVS system; see supplementary for full details. I is our full model but where we avoid giving a hint for any test-time frames; this performs similarly to A, showing we are not disadvantaged in situations where no hint is available. L and G uses our TSDF depth render to modulate cost volume values as in [47] instead of our Hint MLP. Finally, **Ours** (K) outperforms all ablations.

Sensitivity to pose errors. In the revisit scenario, to simulate errors in the relocalization algorithm, we add noise to the one-off alignment between the previous capture and the current capture. The result is shown in Table 5, and shows our tolerance to noisy alignments. See the supplementary for details.

Non-static scenes. Figure 6 shows our system is robust to scene geometry changes after the initial visit. Please see supplementary material for results and details showing our robustness to moving objects

4.4 Limitations

Like other methods which reconstruct 3D scenes by fusing depth maps *e.g.* [52], we only reconstruct geometry we have directly observed, and do not complete any hidden or out-of-frustum surfaces. Since we limit to 3.5m the depths fused into the TSDF, our method has the most benefits when revisiting an area that has been previously observed close by. For example, if the camera is moving forward through a long corridor, our proposed geometric hints would only cover

	Abs Diff↓	Sq Rel↓	RMSE↓	$\delta < 1.05$↑	$\delta < 1.25$↑
Rendered depth from TSDF	.2506	.1293	.3338	23.53	67.97
Densified rendered depth	.1763	.0629	.2264	30.61	81.69
SimpleRecon [52]	.1350	.0437	.1879	46.88	89.32
Ours (no hint)	.1346	.0449	.1879	47.28	89.39
Ours (incremental)	.1255	.0395	.1787	48.76	90.47
Ours (revisit)	.1182	.0368	.1710	50.24	91.78
Ours (revisit, pose noise)	.1199	.0372	.1725	49.46	91.56

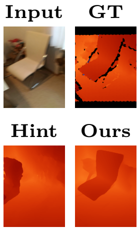


Table 5: Long-term hints on 3RScan. Ours (revisit) shows depth scores when we use the geometry from a *previous visit* as 'hints' for our current depth estimates. This mode beats previous baselines for this dataset, validating our proposal to retain long-term hints. *Rendered depth* uses the prior TSDF's render as the current depth estimate. *Densified rendered depth* improves upon the above baseline with a network to fill in missing geometry. Full metrics and descriptions are in the supplementary. On the right we show we are robust to stale geometry: the chair has moved position since the hint TSDF was generated, but our system gracefully recovers.

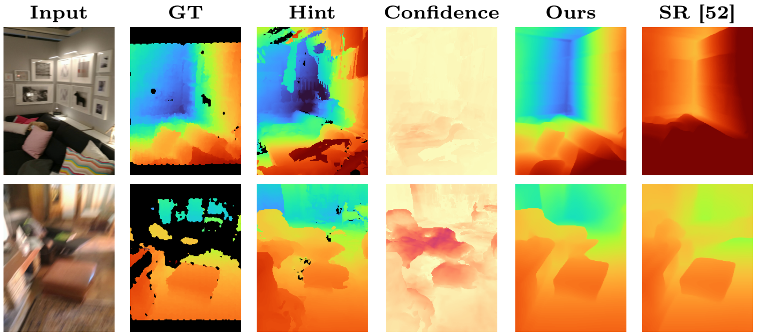


Fig. 6: Qualitative results on 3RScan. Note the major improvement of our model over [52] on this challenging dataset, and how we can recover from misleading hints.

a small region of the image. Similarly to other multi-view stereo based methods, our method struggles with transparent and reflective surfaces.

5 Conclusion

We introduced a system for depth estimation and 3D reconstruction which can take as input, where available, previously-made estimates of the scene's geometry. Our carefully-designed architecture takes as input geometrical hints, and makes high quality depth estimates even when these hints are not available. We showed how our method can make use of hints from the near-term for instantaneous depths in new environments, and also hints from the past when estimating depths in previously visited locations. We evaluated on a range of challenging datasets where we have obtained state-of-the-art depths and reconstructions.

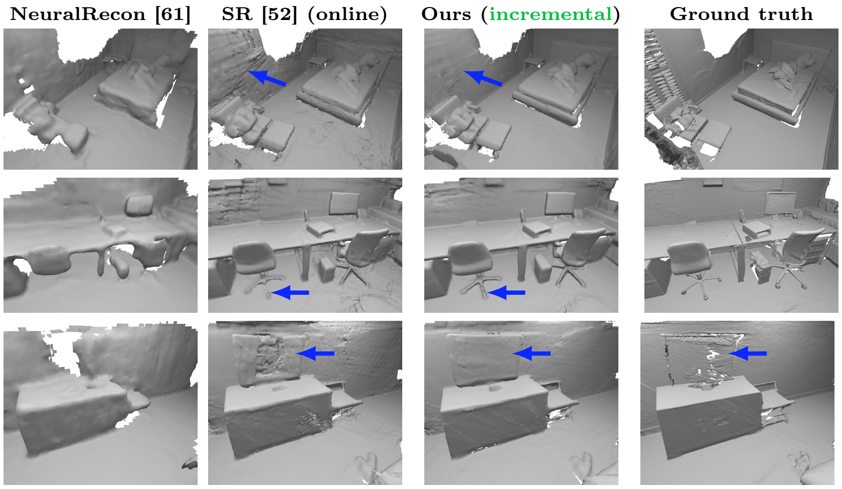


Fig. 7: Qualitative results for meshing (incremental). Our method gives higher quality meshes than baselines *e.g.* [52] by using existing geometry for future predictions.

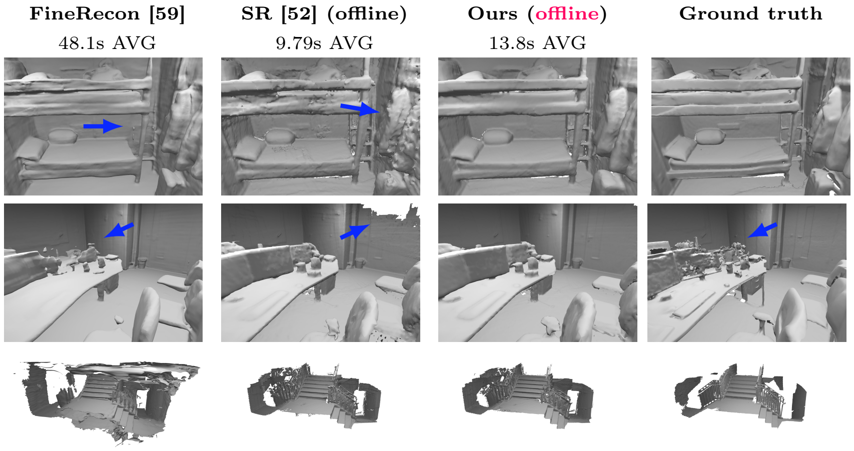
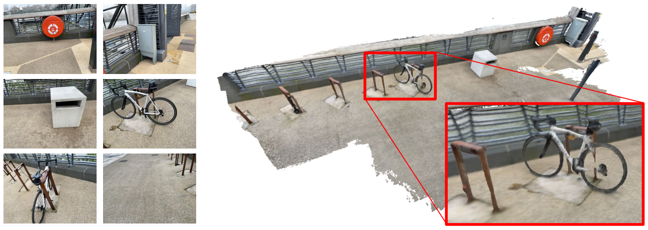


Fig. 8: Qualitative results for meshing (offline, with two passes). Our method enables higher quality meshes than baselines *e.g.* [52] by running it twice.

Fig. 9: Our method generalizes to new domains *e.g.* this outside scene, casually captured with a smartphone. Poses are from ARKit [1].



Acknowledgements

We'd like to thank the Niantic Raptor R&D infrastructure team - Saki Shinoda, Jakub Powierza, and Stanimir Vichev - for their valuable infrastructure support.

References

1. Apple: ARKit (2023), `https://developer.apple.com/documentation/arkit`, Accessed: 5 October 2023
2. Bozic, A., Palafox, P., Thies, J., Dai, A., Nießner, M.: TransformerFusion: Monocular RGB scene reconstruction using transformers. NeurIPS (2021)
3. Cai, C., Ji, P., Yan, Q., Xu, Y.: RIAV-MVS: Recurrent-indexing an asymmetric volume for multi-view stereo. In: CVPR (2023)
4. Casser, V., Pirk, S., Mahjourian, R., Angelova, A.: Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In: AAAI (2019)
5. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: CVPR (2018)
6. Chen, Y., Schmid, C., Sminchisescu, C.: Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In: ICCV (2019)
7. Cheng, X., Wang, P., Yang, R.: Learning depth with convolutional spatial propagation network. PAMI (2019)
8. Cheng, Z., Yang, J., Li, H.: Stereo matching in time: 100+ FPS video stereo matching for extended reality. In: WACV (2023)
9. Choe, J., Joo, K., Imtiaz, T., Kweon, I.S.: Volumetric propagation network: Stereo-lidar fusion for long-range depth estimation. IEEE Robotics and Automation Letters (2021)
10. Collins, R.T.: A space-sweep approach to true multi-image matching. In: CVPR (1996)
11. Conti, A., Poggi, M., Mattoccia, S.: Sparsity agnostic depth completion. In: WACV (2023)
12. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: CVPR (2017)
13. Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised NeRF: Fewer views and faster training for free. In: CVPR (2022)
14. Du, R., Turner, E., Dzitsiuk, M., Prasso, L., Duarte, I., Dourgarian, J., Afonso, J., Pascoal, J., Gladstone, J., Cruces, N., et al.: DepthLab: Real-time 3D interaction with depth maps for mobile augmented reality. In: ACM Symposium on User Interface Software and Technology (2020)
15. 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)
16. Fu, Q., Xu, Q., Ong, Y.S., Tao, W.: Geo-Neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. NeurIPS (2022)
17. Furukawa, Y., Hernández, C.: Multi-view stereo: A tutorial, foundations and trends® in computer graphics and vision (2015)
18. Gao, H., Mao, W., Liu, M.: VisFusion: Visibility-aware online 3D scene reconstruction from videos. In: CVPR (2023)

19. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
20. 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)
21. Guédon, A., Lepetit, V.: SuGaR: Surface-aligned Gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. arXiv preprint arXiv:2311.12775 (2023)
22. Guizilini, V., Ambrus, R., Burgard, W., Gaidon, A.: Sparse auxiliary networks for unified monocular depth prediction and completion. In: CVPR (2021)
23. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
24. Hou, Y., Kannala, J., Solin, A.: Multi-view stereo by temporal nonparametric fusion. In: ICCV (2019)
25. Huang, P.H., Matzen, K., Kopf, J., Ahuja, N., Huang, J.B.: DeepMVS: Learning multi-view stereopsis. In: CVPR (2018)
26. Im, S., Jeon, H.G., Lin, S., Kweon, I.S.: DPSNet: End-to-end deep plane sweep stereo. ICLR (2019)
27. Izquierdo, S., Civera, J.: SfM-TTR: Using structure from motion for test-time refinement of single-view depth networks. In: CVPR (2023)
28. Kähler, O., Prisacariu, V.A., Ren, C.Y., Sun, X., Torr, P.H.S., Murray, D.W.: Very High Frame Rate Volumetric Integration of Depth Images on Mobile Device. IEEE Transactions on Visualization and Computer Graphics (Proceedings International Symposium on Mixed and Augmented Reality 2015 **22**(11) (2015)
29. Kähler, O., Prisacariu, V.A., Murray, D.W.: Real-time large-scale dense 3d reconstruction with loop closure. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII. pp. 500–516 (2016)
30. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. In: ICCV (2017)
31. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
32. Khan, N., Penner, E., Lanman, D., Xiao, L.: Temporally consistent online depth estimation using point-based fusion. In: CVPR (2023)
33. Kulhanek, J., Sattler, T.: Tetra-NeRF: Representing neural radiance fields using tetrahedra. In: ICCV (2023)
34. Kuznietsov, Y., Proesmans, M., Van Gool, L.: CoMoDA: Continuous monocular depth adaptation using past experiences. In: WACV (2021)
35. Li, Z., Müller, T., Evans, A., Taylor, R.H., Unberath, M., Liu, M.Y., Lin, C.H.: Neuralangelo: High-fidelity neural surface reconstruction. In: CVPR (2023)
36. Lipson, L., Teed, Z., Deng, J.: Raft-stereo: Multilevel recurrent field transforms for stereo matching. In: 3DV (2021)
37. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3D surface construction algorithm. In: Seminal graphics: pioneering efforts that shaped the field (1998)
38. Luo, X., Huang, J.B., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. In: ACM SIGGRAPH (2020)
39. Ma, F., Cavalheiro, G.V., Karaman, S.: Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In: ICRA (2019)

40. Ma, F., Karaman, S.: Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In: ICRA (2018)
41. Ma, Z., Teed, Z., Deng, J.: Multiview stereo with cascaded epipolar RAFT. In: ECCV (2022)
42. McCraith, R., Neumann, L., Zisserman, A., Vedaldi, A.: Monocular depth estimation with self-supervised instance adaptation. arXiv:2004.05821 (2020)
43. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: CVPR (2015)
44. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
45. 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)
46. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: ECCV (2020)
47. Poggi, M., Conti, A., Mattoccia, S.: Multi-view guided multi-view stereo. In: IROS (2022)
48. Rakotosaona, M.J., Manhardt, F., Arroyo, D.M., Niemeyer, M., Kundu, A., Tombari, F.: NeRFMeshing: Distilling neural radiance fields into geometrically-accurate 3D meshes. In: 3DV (2023)
49. Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.Y., Johnson, J., Gkioxari, G.: Accelerating 3D deep learning with PyTorch3D. arXiv:2007.08501 (2020)
50. Rich, A., Stier, N., Sen, P., Höllerer, T.: 3DVNet: Multi-view depth prediction and volumetric refinement. In: 3DV (2021)
51. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views. In: CVPR (2022)
52. Sayed, M., Gibson, J., Watson, J., Prisacariu, V., Firman, M., Godard, C.: SimpleRecon: 3D reconstruction without 3D convolutions. In: ECCV (2022)
53. Schönberger, J.L., Zheng, E., Frahm, J.M., Pollefeys, M.: Pixelwise view selection for unstructured multi-view stereo. In: ECCV (2016)
54. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016)
55. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: CVPR (2013)
56. Shu, C., Yu, K., Duan, Z., Yang, K.: Feature-metric loss for self-supervised learning of depth and egomotion. In: ECCV (2020)
57. Sinha, A., Murez, Z., Bartolozzi, J., Badrinarayanan, V., Rabinovich, A.: DELTAS: Depth estimation by learning triangulation and densification of sparse points. In: ECCV (2020)
58. Song, S., Truong, K.G., Kim, D., Jo, S.: Prior depth-based multi-view stereo network for online 3D model reconstruction. Pattern Recognition (2023)
59. Stier, N., Ranjan, A., Colburn, A., Yan, Y., Yang, L., Ma, F., Angles, B.: Finerecon: Depth-aware feed-forward network for detailed 3D reconstruction. In: ICCV (2023)
60. Stier, N., Rich, A., Sen, P., Höllerer, T.: VoRTX: Volumetric 3D reconstruction with transformers for voxelwise view selection and fusion. In: 3DV (2021)
61. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: NeuralRecon: Real-time coherent 3D reconstruction from monocular video. In: CVPR (2021)
62. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: ICML (2021)
63. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant CNNs. In: 3DV (2017)

64. Uy, M.A., Martin-Brualla, R., Guibas, L., Li, K.: SCADE: NeRFs from space carving with ambiguity-aware depth estimates. In: CVPR (2023)
65. Valentin, J., Kowdle, A., Barron, J.T., Wadhwa, N., Dzitsiuk, M., Schoenberg, M., Verma, V., Csaszar, A., Turner, E., Dryanovski, I., et al.: Depth from motion for smartphone AR. Transactions on Graphics (2018)
66. Wald, J., Avetisyan, A., Navab, N., Tombari, F., Niessner, M.: RIO: 3D object instance re-localization in changing indoor environments. In: ICCV (2019)
67. Wang, K., Shen, S.: MVDepthNet: Real-time multiview depth estimation neural network. In: 3DV (2018)
68. Wenchao Du, Hu Chen, H.Y., zhang, Y.: Depth completion using geometry-aware embedding. In: ICRA (2022)
69. Wong, A., Soatto, S.: Unsupervised depth completion with calibrated backprojection layers. In: ICCV (2021)
70. Xin, Y., Zuo, X., Lu, D., Leutenegger, S.: SimpleMapping: Real-Time Visual-Inertial Dense Mapping with Deep Multi-View Stereo. In: ISMAR (2023)
71. Yang, J., Mao, W., Alvarez, J.M., Liu, M.: Cost volume pyramid based depth inference for multi-view stereo. In: CVPR (2020)
72. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: MVSNet: Depth inference for unstructured multi-view stereo. In: ECCV (2018)
73. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. In: NeurIPS (2021)
74. Yu, Z., Peng, S., Niemeyer, M., Sattler, T., Geiger, A.: MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. NeurIPS (2022)
75. Zhang, F., Prisacariu, V., Yang, R., Torr, P.H.: GA-Net: Guided aggregation net for end-to-end stereo matching. In: CVPR (2019)
76. Zhang, F., Qi, X., Yang, R., Prisacariu, V., Wah, B., Torr, P.: Domain-invariant stereo matching networks. In: ECCV (2020)
77. Zhang, Z., Peng, R., Hu, Y., Wang, R.: GeoMVSNet: Learning multi-view stereo with geometry perception. In: CVPR (2023)
78. 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)
79. Zuo, X., Yang, N., Merrill, N., Xu, B., Leutenegger, S.: Incremental dense reconstruction from monocular video with guided sparse feature volume fusion. IEEE Robotics and Automation Letters (2023)