SimpleRecon: 3D Reconstruction Without 3D Convolutions

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

Abstract. Traditionally, 3D indoor scene reconstruction from posed images happens in two phases: per-image depth estimation, followed by depth merging and surface reconstruction. Recently, a family of methods have emerged that perform reconstruction directly in final 3D volumetric feature space. While these methods have shown impressive reconstruction results, they rely on expensive 3D convolutional layers, limiting their application in resource-constrained environments. In this work, we instead go back to the traditional route, and show how focusing on high quality multi-view depth prediction leads to highly accurate 3D reconstructions using simple off-the-shelf depth fusion. We propose a simple state-of-the-art multi-view depth estimator with two main contributions: 1) a carefully-designed 2D CNN which utilizes strong image priors alongside a plane-sweep feature volume and geometric losses, combined with 2) the integration of keyframe and geometric metadata into the cost volume which allows informed depth plane scoring. Our method achieves a significant lead over the current state-of-the-art for depth estimation and close or better for 3D reconstruction on ScanNet and 7-Scenes, yet still allows for online real-time low-memory reconstruction. Code, models and results are available at `https://nianticlabs.github.io/simplerecon`

1 Introduction

Generating 3D reconstructions of a scene is a challenging problem in computer vision which is useful for tasks such as robotic navigation, autonomous driving, content placement for augmented reality and historical preservation [47, 77]. Traditionally, such 3D reconstructions are generated from 2D depth maps obtained using multi-view stereo (MVS) [56, 11], which are then fused into a 3D representation from which a surface is extracted [57, 31]. Recent advances in deep learning have enabled convolutional methods to outperform classical methods for depth prediction from multiple stereo images, spearheaded by GC-Net [32] and MVSNet [78]. Key to these methods is the use of 3D convolutions to smooth and regularize a 4D ($C \times D \times H \times W$) cost volume, which performs well in practice but is expensive in both time and memory. This could preclude their use on low

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

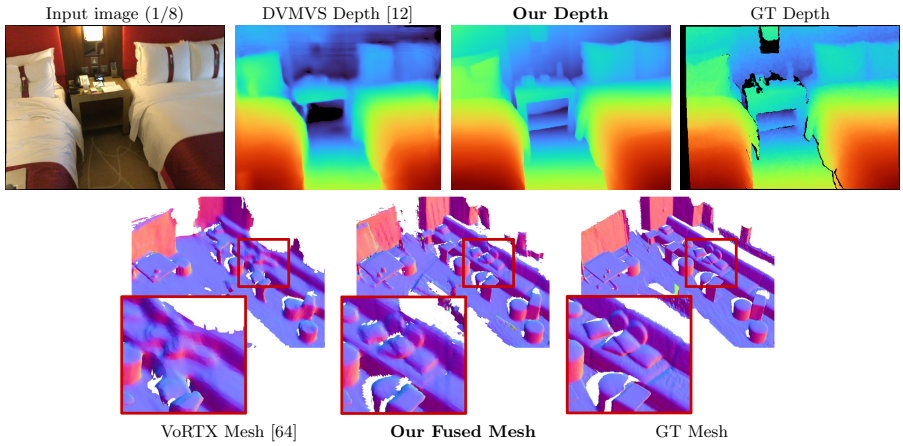

Fig. 1. Qualitative preview of our method. Our method significantly improves upon previous state-of-the-art monocular MVS methods [12] in depth prediction and matches the current volumetric state-of-the-art in full scene reconstruction [64].

power hardware e.g. smartphones, where overall compute energy and memory are limited. The same is true of recent depth estimators which use LSTMs and Gaussian processes for improved depth accuracy [12, 23].

A new stream of work started by ATLAS [46] (and extended by e.g. [65, 2, 7]) performs the reconstruction directly in 3D space by predicting a truncated signed distance function (TSDF) from a 4D feature volume computed from the input images. Again, these works give good results but use expensive 3D convolutions.

In this paper we go *back to basics*, showing that, surprisingly, it is possible to obtain state-of-the-art depth accuracy with a simple 2D CNN augmented with a cost volume. Our method also gives competitive scores in 3D scene reconstruction using off-the-shelf TSDF fusion [47], all without expensive 3D convolutions. Key to our method is the novel incorporation of cheaply available *metadata* into the cost volume, which we show significantly improves depth and reconstruction quality. Our main contributions are: (1) The integration of keyframe and geometric metadata into the cost volume using a multi-level perceptron (MLP), which allows informed depth plane scoring, and (2) A carefully-designed 2D CNN that utilizes strong image priors alongside a plane-sweep 3D feature volume and geometric losses. We evaluate our 'back-to-basics' method against all recent published methods on the challenging ScanNetv2 [10] dataset on both depth estimation and 3D scene reconstruction (Sec. 4), and show it generalizes on 7-Scenes [18] data (Table 1) and casually captured footage (Fig. 6).

By combining our novel cost volume metadata with principled architectural decisions that result in better depth predictions, we can avoid the computational cost associated with 3D convolutions, potentially enabling use in embedded and resource-constrained environments. We have released code, models and precomputed results at `https://nianticlabs.github.io/simplerecon`

2 Related Work

Our method is related to prior work in *stereo* depth estimation, *multi-view* depth estimation, and 3D reconstruction.

2.1 Depth from Calibrated Stereo Pairs

Many methods for estimating depth use calibrated stereo pairs of images in order to estimate disparity, which can be translated into depth using camera parameters and the intra-axial distance between the camera positions. Early methods compare patches [22, 82, 44], similar to work in optical flow estimation [16]. This laid the groundwork for GCNet [32], which built on earlier plane-sweep stereo works [8, 29] to develop now-ubiquitous cost-volume-based depth estimation. The typical architecture is feature extraction from input images, then feature matching and reduction into a cost volume, followed by convolutional layers to output the final disparity. Further improvements include post-processing the cost volume [4, 83, 84, 6] using multiscale information, carefully designed network layers that mimic classical refinement methods, and spatial pyramid pooling. The best results typically come from running 3D convolutions on a 4D ($C \times D \times H \times W$) cost volume, pioneered by Chang et al. in PSMNet [4]; this can be very computationally expensive. A more attractive option is to create a 3D cost volume ($D \times H \times W$) by reducing along the feature dimension, meaning 2D convolutions can be used for further processing [71, 79]; however, this typically comes at the expense of depth quality. In this work we show how, with simple tricks and clever reduction techniques, a method with a 3D cost volume can outperform existing 4D cost volume methods for both depth estimation and 3D scene reconstruction.

2.2 Multi-view Stereo Depth

Multi-View Stereo (MVS) is a more general problem which aims to estimate depth at a *reference* viewpoint using one or more additional *source* viewpoints captured from arbitrary locations. Knowledge of camera intrinsics and extrinsics for both reference and source views are generally assumed, but can also be estimated offline using e.g. structure-from-motion [57] or on-line using inertial and camera tracking, like that provided by ARKit or ARCore.

Classical MVS methods typically use patch matching with photometric consistency to estimate a depth map followed by depth fusion and refinement [17, 58]. In contrast, early learning-based methods backprojected dense image features from multiple viewpoints into 3D volumes representing the entire scene and then predicted voxel occupancy or surface probability from a fused 3D volume [27, 30]. Recent methods, inspired by binocular stereo matching techniques, combine these approaches, performing epipolar-geometry-consistent matching on image pixels (e.g. in MVDepthNet [70] and DeepMVS [25]), or extracted features (e.g. in DPSNet [26] and DeepVideoMVS [12]) to produce a matching cost volume. The cost volume can optionally be reduced using a dot product [12] or mean absolute difference [71, 48], and then processed using convolutional

layers. Further works incorporate additional scene information to regularize the cost volume and refine the final output by using reference image features [78], by taking into account occlusions [39] and moving objects [76], or with a Gaussian process prior [23]. Others have proposed methods to combine multiple reference views, e.g. by pooling in DeepMVS [25] or averaging the feature volumes in DPSNet [26, 39]. Aside from use of keyframe image values and features in a cost volume, depth-estimation approaches have used temporal information in varying ways, such as LSTMs to fuse volumes over multiple frames [68, 12, 51], or by a test-time optimization of reprojection error [3, 5, 42, 45, 61, 33]. However, all these approaches use only the color image as inputs, discarding additional information such as viewing direction and relative pose estimation after the cost volume is computed. In this work, we extend the matching cost volume into a matching feature volume, which uses readily-available metadata to produce higher-quality depth maps.

2.3 3D Scene Reconstruction from Posed Views

Classical methods for creating dense 3D reconstructions from images typically compute dense depth per-view, such as [58], followed by a surface reconstruction such as Delaunay triangulation [34] or Poisson surface reconstruction [31]. The seminal work Kinect Fusion [47] demonstrated *real-time* 3D scene reconstruction from depth-maps using a volumetric truncated signed distance field (TSDF) representation [9], from which a mesh can be obtained using marching cubes [40]. A family of methods improved on it, allowing it to work more efficiently on larger scenes [73, 49, 28, 52], to handle moving objects [59, 55], or to perform loop closure [74], all of which solidified TSDF fusion as a key component of real-time mapping.

Recent deep learning methods forego depth estimation, instead extracting 2D image features from keyframes and backprojecting these features into 3D space to produce a 4D feature volume [63]. In ATLAS [46], 3D convolutions on such a feature volume are used to regress a TSDF for the scene, which significantly improved reconstruction quality over the then-state-of-the-art method of learning-based MVS followed by traditional TSDF fusion [47]. NeuralRecon [65] extended this to refine the TSDF in a coarse-to-fine manner using recurrent layers, while TransformerFusion [2] and VoRTX [64] further improved performance using transformers [69] to learn feature matching. Recently methods proposed combining volumetric reasoning – via a 3D encoder-decoder – with MVS reconstruction; either iteratively in the case of 3DVNet [54] or using pose-invariant 3D convolutional layers in VolumeFusion [7].

Although these methods produce high-quality reconstructions, the use of 3D convolutions, transformers, or recurrent layers makes them computationally expensive and memory-intensive. Furthermore, they predict the whole scene TSDF at once, making real-time use impossible, or rely on complex sparsification [65] or attention mechanisms [2] to allow for progressive updates. In contrast, we take a simpler approach: by focusing on predicting high-quality depth maps, we are able to use efficient off-the-shelf TSDF fusion methods such as Infinitam [52]. This allows our method to achieve real-time and progressive 3D reconstructions

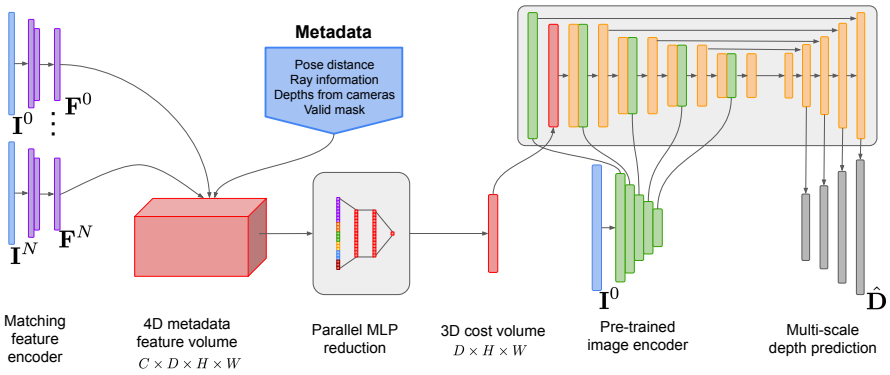

Fig. 2. Overview of our method. Our key contribution is the injection of cheaply-available *metadata* into the feature volume. Each volumetric cell is then reduced in parallel with an MLP into a feature map before input into a 2D encoder-decoder [87].

at low compute and memory footprints, with accuracy competitive with volumetric methods but without the use of 3D convolutions.

3 Method

We take as input a reference image $\mathbf{I}^0$, a set of source images $\mathbf{I}^{n\in\{1,\dots,N-1\}}$, as well as their intrinsics and relative camera poses. At training time we also assume access to a ground truth depth map $\mathbf{D}^{\text{gt}}$ aligned with each RGB image; at test time our aim is to predict dense depth maps $\hat{\mathbf{D}}$ for each reference image.

3.1 Method Overview

Our depth estimation model sits at the intersection of monocular depth estimation [13, 19] and MVS via plane sweep [8]. We augment a depth prediction encoder-decoder architecture with a cost volume; see Figure 2. Our image encoder extracts matching features from the reference and source images for input to a cost volume. The output of the cost volume is processed using a 2D convolutional encoder-decoder network, augmented with image level features extracted using a separate pretrained image encoder.

Our key insight is to inject readily available *metadata* into the cost volume alongside the typical deep image features, allowing the network access to useful information such as geometric and relative camera pose information. Figure 3 shows in detail the construction of our feature volume. By incorporating this previously unexploited information, our model is able to significantly outperform previous methods on depth prediction without the need for costly 4D cost volume reductions [32, 78], complex temporal fusion [12], or Gaussian processes [23].

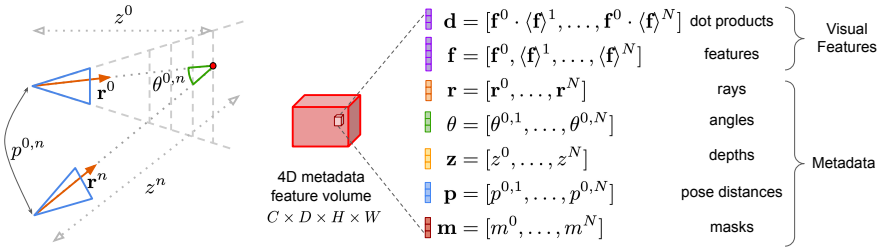

Fig. 3. Metadata insertion. Typical MVS systems predict depth from warped *features* or differences between features e.g. dot products. We additionally include cheaply-available *metadata* for improved performance. Indices (k, i, j) are omitted for clarity.

We first describe our novel metadata component and explain how it is incorporated into the network (Section 3.2). We then set out our network architecture and losses (Sections 3.3 and 3.4), giving best practices for depth estimation.

3.2 Improving the Cost Volume with Metadata

In traditional stereo techniques, there exists important information which is typically ignored. In this work, we incorporate readily available *metadata* into the cost volume, allowing our network to aggregate information across views in an informed manner. This can be done both *explicitly* via appending extra feature channels and *implicitly* via enforcing specific feature ordering.

We propose injecting metadata into our network by augmenting image-level features inside the cost volume with additional metadata channels. These channels encode information about the 3D relationship between the images used to build the cost volume, allowing our network to better reason about the relative importance of each source image for estimating depth for a particular pixel.

Our cost volume is therefore a 4D tensor of dimension $C \times D \times H \times W$, where for each spatial location (k, i, j), k is the depth plane index, we have a C dimensional feature vector. This vector comprises reference image features $\mathbf{f}^0_{k,i,j}$ and a set of warped source image features $\langle \mathbf{f} \rangle^n_{k,i,j}$ for $n \in [1, N]$, where $\langle\ \rangle$ indicates that the features are perspective-warped into the reference camera frame, along with the following metadata components:

Feature dot product — The dot product between reference image features and warped source image features, i.e. $\mathbf{f}^0 \cdot \langle \mathbf{f} \rangle^n$. This is commonly used as the *only* matching affinity in the cost volume.

Ray directions $\mathbf{r}^0_{k,i,j}$ and $\mathbf{r}^n_{k,i,j} \in \mathbb{R}^3$ — The normalized direction to the 3D location of a point (k, i, j) in the plane sweep from the camera origins.

Reference plane depth $z^0_{k,i,j}$ — The perpendicular depth from the reference camera to the point at position k, i, j in the cost volume.

Reference frame reprojected depths $z^n_{k,i,j}$ — The perpendicular depth of the 3D point at position k, i, j in the cost volume to source camera n.

Relative ray angles $\theta^{0,n}$ — The angle between $\mathbf{r}^0_{k,i,j}$ and $\mathbf{r}^n_{k,i,j}$.

Relative pose distance $p^{0,n}$ — A measure of the relative pose distance between the pose of the reference camera and each source frame [12]:

$$p^{0,n} = \sqrt{||\mathbf{t}^{0,n}|| + \frac{2}{3}\text{tr}(\mathbb{I} - \mathbf{R}^{0,n})} \tag{1}$$

Depth validity masks $m^n_{k,i,j}$ — This binary mask indicates if point (k, i, j) in the cost volume projects in front of the source camera n or not.

An overview of these features is given in Fig 3. Each resulting $\mathbf{f}_{k,i,j}$ is processed by a simple multi layer perceptron (MLP), outputting a single scalar value for each location (k, i, j). This scalar can be thought of an initial estimate of the likelihood that the depth of pixel i, j is equal to the kth depth plane.

Metadata motivation — We argue that by appending metadata-derived features into our cost volume, the MLP can learn to correctly weigh the contribution of each source frame at each pixel location. Consider for instance the pose distance $p^{s,n}$; it is clear that for depths farther from the camera, the matching features from source frames with a greater baseline would be more informative. Similarly, ray information can be useful for reasoning about occlusions; if features from the reference frame disagree with those from a source frame but there is a large angle between camera rays, then this could be explained by an occlusion rather than incorrect depth. Depth validity masks can help the network to know whether to trust features from source camera n at (k, i, j), By allowing our network access to this kind of information, we give it the ability to conduct such geometric reasoning when aggregating information from multiple source frames.

Implicit metadata incorporation — In addition to explicitly providing metadata as extra features, we also propose *implicitly* encoding metadata via feature ordering. This is made possible by the inherent order dependence of MLP networks, which we exploit by choosing the ordering in which we stack our source features $\mathbf{f}^n$. We advocate ordering $\mathbf{f}^n$ by frame pose distance $p^{s,n}$, a measure shown by [12, 23] to be effective for optimal keyframe selection. This ordering allows the MLP to learn a prior on pose distance and feature relevance.

Our experiments show that by including metadata in our network, both *explicitly* via extra feature channels and *implicitly* via feature ordering, we can obtain a significant boost to depth estimation accuracy, bringing with it improved 3D reconstruction quality; see Table 4. Whilst previous works have included tensors related to camera intrinsics [14] and extrinsics [86] for monocular depth estimation, we believe that our use of metadata is a novel innovation for multi-view-stereo depth estimation.

3.3 Network Architecture Design

Our network is based on a 2D convolutional encoder-decoder architecture similar to prior works such as [12, 71]. When constructing such networks, we find that

there are important design choices which can give significant improvements to depth prediction accuracy. We specifically aim to keep the overall architecture simple, avoiding complex structures such as LSTMs [12] or GPs [23], and making our baseline model lightweight and interpretable.

Baseline cost volume fusion — While RNN-based temporal fusion methods are often used [65, 12], they significantly increase the complexity of the system. We instead make our baseline cost-volume fusion as simple as possible and find that simply summing the dot-product matching costs between the reference view and each source view leads to results competitive with state-of-the-art depth estimation techniques, as shown in Table 1 with the heading "no metadata".

Image encoder and feature matching encoder — Prior depth estimation works have shown the impact of more powerful image encoders for the task of depth estimation, both in monocular [20, 72, 53] and multi-view estimation [71]. DeepVideoMVS [12] make use of an MnasNet [66] as their image encoder, chosen for its relatively low latency. We propose instead utilizing a still-small but more powerful EfficientNetv2 S encoder [67], the smallest of its family. While this does come with a cost of increased parameter count and 10% slower execution, it yields a sizeable improvement to depth estimation accuracy, especially for precise metrics such as Sq Rel and $\delta < 1.05$. See Table 4 for full results.

For producing matching feature maps, we use the first two blocks from ResNet18 [21] for efficiency, we experimented with FPN [37] following [12], which slightly improved accuracy at the expense of a 50% slower overall run-time.

Fuse multi-scale image features into the cost volume encoder — In 2D CNN based deep stereo and multi-view stereo, image features are typically combined with the output of the cost volume at a single scale [71, 36].

More recently, DeepVideoMVS [12] instead proposed concatenating deep image features at *multiple* scales, adding skip connections between the image encoder and cost volume encoder at all resolutions. Whilst this has been shown to be helpful for their LSTM-based fusion network, we find that it is similarly important for our architecture.

Number of source images — While other methods show diminishing returns as additional source frames are added [12], our method is better able to incorporate this additional information and displays increased performance with up to 8 views. We posit that incorporating additional metadata for each frame allows the network to make a more informed decision about the relative weightings of each frame's features when inferring the final cost. In contrast, methods such as MVDepthNet [70], MVSNet [78], ManyDepth [71] and ATLAS [46] give each frame equal weight during a update, thus potentially overwhelming the most useful information with lower-quality features.

3.4 Loss

We supervise our training using a combination of geometric losses, inspired by recent MVS methods [12, 13, 78, 25, 78] as well as monocular depth estimation

techniques [20, 35, 81, 53]. We find that careful choice of loss function is required for best performance, and that supervising intermediate predictions at lower output scales substantially improves results.

Depth regression loss — We follow [13] and densely supervise predictions using log-depth, but use an absolute error on log depth for each scale s,

$$\mathcal{L}_{\text{depth}} = \frac{1}{HW} \sum_{s=1}^{4} \sum_{i,j} \frac{1}{s^2} | \uparrow_{gt} \log \hat{\mathbf{D}}^{s}_{i,j} - \log \mathbf{D}^{\text{gt}}_{i,j} |, \tag{2}$$

where we upsample each lower scale depth using nearest neighbor upsampling [12] to the highest scale we predict at with the $\uparrow_{gt}$ operator. We average this loss per pixel, per scale and per batch. Our experiments found this loss to perform better than the scale-invariant formulation of Eigen et al. [13, 1], while producing much sharper depth boundaries, resulting in higher fused reconstruction quality.

Multi-scale gradient and normal losses — We follow [35, 81, 53] and use a multi-scale gradient loss on our highest resolution network output

$$\mathcal{L}_{\text{grad}} = \frac{1}{HW} \sum_{s=1}^{4} \sum_{i,j} | \nabla \downarrow_s \hat{\mathbf{D}}_{i,j} - \nabla \downarrow_s \mathbf{D}^{\text{gt}}_{i,j} |, \tag{3}$$

where ∇ is first order spatial gradients and $\downarrow_s$ represents downsampling to scale s. Inspired by [80] we also use a simplified normal loss, where $\mathbf{N}$ is the normal map computed using the depth and intrinsics (see supp. mat. for details),

$$\mathcal{L}_{\text{normals}} = \frac{1}{2HW} \sum_{i,j} 1 - \hat{\mathbf{N}}_{i,j} \cdot \mathbf{N}_{i,j}. \tag{4}$$

Multi-view depth regression loss — We use ground-truth depth maps for each source view as additional supervision by projecting predicted depth $\hat{D}$ into each source view and averaging absolute error on log depth over all valid points,

$$\mathcal{L}_{\text{mv}} = \frac{1}{NHW} \sum_{n} \sum_{i,j} | \log \hat{\mathbf{D}}^{0 \rightarrow n}_{i,j} - \log \mathbf{D}^{\text{gt}}_{n,i,j} | \tag{5}$$

where $\hat{\mathbf{D}}^{0 \rightarrow n}$ is the depth predicted for the reference image of index 0, projected into source view n. This is similar in concept to the depth regression loss above, but for simplicity is applied only on the final output scale.

Total loss — Overall our total loss is:

$$\mathcal{L} = \mathcal{L}_{\text{depth}} + \alpha_{\text{grad}} \mathcal{L}_{\text{grad}} + \alpha_{\text{normals}} \mathcal{L}_{\text{normals}} + \alpha_{\text{mv}} \mathcal{L}_{\text{mv}}, \tag{6}$$

with $\alpha_{\text{grad}} = 1.0 = \alpha_{\text{normals}} = 1.0$, and $\alpha_{\text{mv}} = 0.2$, chosen experimentally using the validation set.

	ScanNetv2					7Scenes				
	Abs Diff↓	Abs Rel↓	Sq Rel↓	$\delta < 1.05 \uparrow$	$\delta < 1.25 \uparrow$	Abs Diff↓	Abs Rel↓	Sq Rel↓	$\delta < 1.05 \uparrow$	$\delta < 1.25 \uparrow$
DPSNet [26]	0.1552	0.0795	0.0299	49.36	93.27	0.1966	0.1147	0.0550	38.81	87.07
MVDepthNet [70]	0.1648	0.0848	0.0343	46.71	92.77	0.2009	0.1161	0.0623	38.81	87.70
DELTAS [62]	0.1497	0.0786	0.0276	48.64	93.78	0.1915	0.1140	0.0490	36.36	88.13
GPMVS [23]	0.1494	0.0757	0.0292	51.04	93.96	0.1739	0.1003	0.0462	42.71	90.32
DeepVideoMVS, fusion [12]*	0.1186	0.0583	0.0190	60.20	96.76	0.1448	0.0828	0.0335	47.96	93.79
Ours (no metadata)	0.0941	0.0467	0.0139	70.48	97.84	0.1105	0.0617	0.0175	57.30	97.02
Ours	0.0885	0.0434	0.0125	73.16	98.09	0.1045	0.0575	0.0153	59.78	97.38

Table 1. Depth evaluation. 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 [12], or evaluated for each method using their keyframes. *We boosted [12]'s scores by using three inference frames instead of two. [12] also use a custom 90/10 split; we show SimpleRecon results using this in the supplementary.

3.5 Implementation Details

We implemented the method using PyTorch [50, 15, 75] and we use an EfficientNetV2 S backbone [67], with a decoder similar to UNet++ [87], and use the first 2 blocks of ResNet18 (R18) for matching feature extraction. Please see supplementary material for a detailed architecture description. We train with the AdamW optimizer [41] for 100k steps – approximately 9 epochs – with a weight decay of 10^{-4}, and a learning rate of 10^{-4} for 70k steps, 10^{-5} until 80k, then dropped to 10^{-6} for remainder, which takes 36 hours on two 40GB A100 GPUs. Models with the lowest validation loss are used for evaluation. We resize images to 512×384 and predict depth at half that resolution. When training, random color augmentations to brightness, contrast, saturation, and hue are applied per image using TorchVision [43] with $\delta = 0.2$ for all parameters, and horizontal flips with a probability of 50%. Keyframes are selected following DeepVideoMVS [12].

4 Experiments

We train and evaluate our method on the 3D scene reconstruction dataset ScanNetv2 [10], which comprises 1,201 training, 312 validation, and 100 testing scans of indoor scenes, all captured with a handheld RGBD sensor. We also evaluate our ScanNetv2 models without fine-tuning on 7-Scenes [60] using [12]'s test split.

4.1 Depth Estimation

In Table 1, we evaluate the depth predictions from our network using the metrics established in Eigen et al. [13]. We also introduce a tighter threshold tolerance $\delta < 1.05$ to differentiate between high quality models. We directly compare to previously published results, including DeepVideoMVS [12], on ScanNetv2 and 7-Scenes (Table 1).

We use the standard test split for ScanNetv2 and the test split defined by [12] for 7-Scenes. We compute depth metrics for every keyframe as in [12] and average

	Volumetric	Comp↓	Acc↓	Chamfer↓	Prec↑	Recall ↑	F-Score ↑
RevisitingSI [24]	No	14.29	16.19	15.24	0.346	0.293	0.314
MVDepthNet [70]	No	12.94	8.34	10.64	0.443	0.487	0.460
GPMVS [23]	No	12.90	8.02	10.46	0.453	0.510	0.477
ESTDepth [38]	No	12.71	7.54	10.12	0.456	0.542	0.491
DPSNet [26]	No	11.94	7.58	9.77	0.474	0.519	0.492
DELTAS [62]	No	11.95	7.46	9.71	0.478	0.533	0.501
DeepVideoMVS [12]	No	10.68	6.90	8.79	0.541	0.592	0.563
COLMAP [58]	No	10.22	11.88	11.05	0.509	0.474	0.489
ATLAS [46]	Yes	7.16	7.61	7.38	0.675	0.605	0.636
NeuralRecon [65]	Yes	5.09	9.13	7.11	0.630	0.612	0.619
3DVNet [54]	Yes	7.72	6.73	7.22	0.655	0.596	0.621
TransformerFusion [2]	Yes	5.52	8.27	6.89	0.728	0.600	0.655
VoRTX [64]	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

Table 2. 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.

across all keyframes in the test sets. Surprisingly, our model, which uses no 3D convolutions, outperforms all baselines on depth prediction metrics. In addition, our baseline model with no metadata encoding (i.e. using only the dot product between reference and source image features) also performs well in comparison to previous methods, showing that a carefully designed and trained 2D network is sufficient for high-quality depth estimation. We show qualitative results for depth and normals in Fig. 4 and Fig. 5 respectively.

4.2 3D Reconstruction Evaluation

Our 3D reconstructions are evaluated using the standard protocol established by TransformerFusion [2]. Their evaluation uses a ground truth mesh based prediction mask to cull away parts of the prediction such that methods are not unfairly penalized for predicting potentially correct geometry that is missing in the ground truth. Scores are shown in Table 2. Our simple depth-based method outperforms state-of-the-art depth estimators for fusion by a wide margin. Although we do not perform global refinement of the resulting volume after fusion, we are still able to outperform more expensive *volumetric* methods in some metrics, showing overall competitive performance with lower complexity.

We also compute scores using the ATLAS [46] mesh evaluation protocol. However, we find this evaluation is inconsistent; comparing a ground truth mesh against itself does not result in a score of zero, nor does performance on metrics match inspection for visual quality or correlate well across methods. These scores, and more details on this discrepancy, are given in the supplementary.

4.3 3D Reconstruction Latency

For online and interactive 3D reconstruction applications, reducing the latency from sensor reading to 3D representation update is crucial. Most recent recon-

	Volume Update Mode	Breakdown	Update Latency (ms)↓	F-Score↑
ATLAS [46]	Volume 3D CNN	2D CNN (29ms) + 3D CNN (353ms)	382ms	0.636
NeuralRecon* [65]	3D Chunk Fusion + GRU	2D CNN (12ms) + GRU (78ms)	90ms	0.619
3DVNet [54]	Iterative 3D CNN	Refine Depths and Feature Cloud (23875ms)	23875ms	0.621
TransformerFusion [2]	Transformer Fusion + 3D CNN	2D CNN (131ms) + Refinement (195ms)	326ms	0.655
VoRTX [64]	Transformer Fusion + 3D CNN	2D CNN (23ms) + Refinement (4527ms)	4550ms	0.703
Ours	TSDF Fusion	2D Depth CNN (70ms) + TSDF fuse (2ms)	72ms	0.671

Table 3. Frame integration latencies for 3D reconstruction. We measure latency as the time to incorporate a new image measurement to a 3D representation. Note that NR reports time amortized over all keyframes. *requires sparse 3D convolutions.

struction methods use 3D CNN architectures [65, 46, 64, 2] that require expensive and often specialized hardware for sparse matrix computation. This makes them prohibitive for applications on low power devices (smartphones, IoE devices) where both compute and power are limited, or may simply not support the operations. Reconstruction methods often report amortized frame time, where the total compute time for select keyframes is averaged over all frames in a sequence. While this is a useful metric for full offline scene reconstruction performance, it is not indicative of online performance, especially when considering latency.

In Table 3 we compute the per-frame integration time given a new RGB frame. Some methods may not be designed to run on every keyframe. Notably, NeuralRecon [65] updates a chunk in world space when 9 keyframes have been received. However, for fairness across methods, we do not count the time spent waiting to satisfy a keyframe requirement and assume that the output of immediately available frames with potentially subpar pose distances is comparable to how the method was intended to perform. For methods that require a 3D CNN, we report the time for one 2D keyframe integration and a complete pass of their 3D CNN network. Although our method is slower than methods such as [65] on a per-keyframe basis, we can quickly perform updates to the reconstructed volume using online TSDF fusion methods, resulting in low update latencies.

4.4 Ablations

In order to show the importance of our best practices and novel contributions, we ablate different parts of our network and training routine. Results for depth estimation and mesh reconstruction metrics on ScanNet [10] are shown for ablations in Table 4, following the evaluation procedures in Section 4.

Baseline — We first show that using *no* MLP and 16 feature channels, reduced using a dot product, our performance greatly suffers. Interestingly, using 64 feature channels instead of 16 degrades accuracy while being significantly slower.

Frame ordering — We compare two models where we shuffle the ordering of the keyframes, instead of relying on the pose distance. As we can see while both models suffer from random ordering, the full model, which has access to the pose distance as metadata, does not suffer as much.

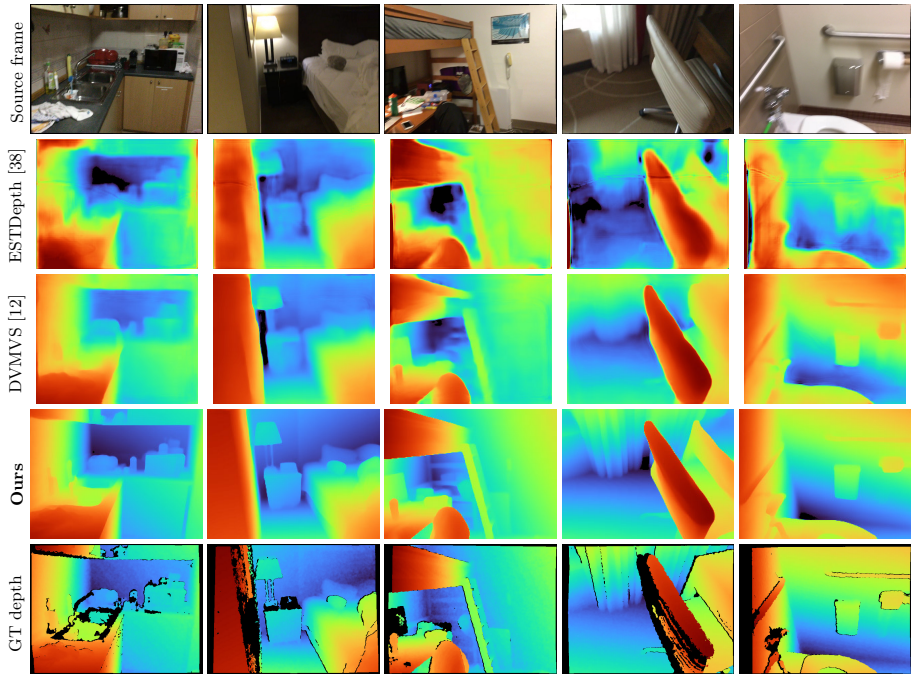

Fig. 4. Depth predictions on ScanNet. Our model produces significantly sharper and more accurate depths than the baselines. See sup. mat. for additional results.

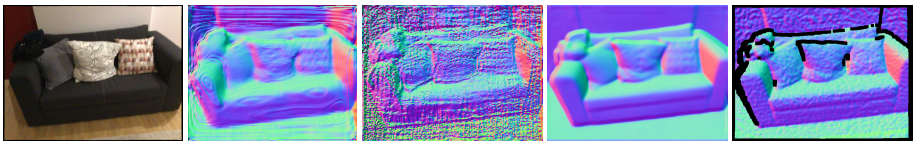

Fig. 5. Estimated Normals on ScanNet. Our model produces significantly sharper normals. We compute the estimated normals from depth, see supp. mat. for details.

Metadata — In this section all the models make use of the MLP cost volume reduction, but we vary the input of that MLP. We start with our baseline model, using only the feature dot products aggregated using a sum. We then add the features, their depth and validity mask, reduced using our MLP. We keep adding more metadata until we reach our full model. Accuracy increases with the amount of information provided to the model.

Views — In addition, we show that our method can incorporate information from many source views. As we increase from 2 views to 8, our performance continues to improve. In contrast, DeepVideoMVS's performance remains relatively constant when using more than three source frames [12]. In addition, we ablate the cost volume entirely by zeroing its output (creating a monocular method),

	Depth evaluation					Mesh eval	
	Abs Diff↓	Sq Rel↓	RMSE↓	$\delta < 1.05$ ↑	$\delta < 1.25$ ↑	Chamfer↓	F-score↑
Ours w/ all metadata, 8 ordered frames, dot prod CV 16c, ENv2S + R18	**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/ dot product CV 64c	0.0944	0.0140	0.1548	70.49	97.84	6.08	65.4
Ours w/o metadata, shuffled frames	0.0920	0.0135	0.1521	71.59	97.91	6.04	65.6
Ours w/ metadata, shuffled frames	0.0906	0.0129	0.1490	72.09	98.03	5.92	66.3
Ours baseline w/ dot product CV 16c	0.0941	0.0139	0.1544	70.48	97.84	6.29	64.2
Ours dot + feats + mask + depth	0.0904	0.0132	0.1509	72.63	98.03	5.92	66.5
Ours dot + feats + mask + depth + ray + angle	0.0896	0.0127	0.1481	72.76	98.09	5.88	66.6
Ours dot + feats + mask + depth + ray + angle + pose distance	**0.0885**	**0.0125**	**0.1468**	**73.16**	**98.09**	**5.81**	**67.1**
Ours w/ 1 frame – w/o CV	0.1742	0.0374	0.2330	40.96	90.03	9.26	47.0
Ours w/ 2 frames	0.1230	0.0198	0.1803	57.15	96.21	7.51	56.7
Ours w/ 4 frames	0.1036	0.0151	0.1611	65.62	97.60	6.57	62.3
Ours w/ metadata but w/ MnasNet at 320×256 (matching [12])	0.0947	0.0146	0.1587	71.24	97.68	5.92	66.3

Table 4. Ablation Evaluation. Ablation evaluation on depth and reconstruction metrics using DVMVS keyframes for ScanNet. Scores for our full method are bolded.

leading to greatly decreased performance, showing that a strong metric depth estimate from the cost volume is required to resolve scale ambiguity.

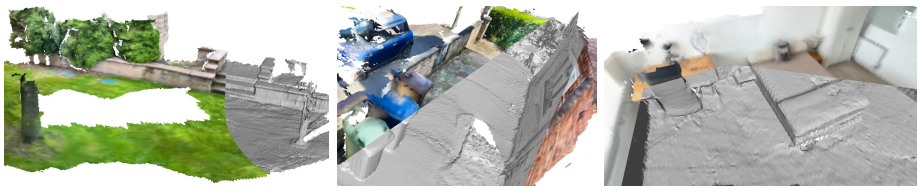

Fig. 6. Fused meshes on unseen data. Our model generalizes on unseen environments, including outdoors, captured on smartphone. See video for live reconstruction.

5 Conclusion

We propose SimpleRecon, which produces state-of-the-art depth estimations and 3D reconstructions, all without the use of expensive 3D convolutions. Our key contribution is to inject cheaply available *metadata* into the cost volume. Our evaluation shows that metadata boosts scores, and in partnership with our set of careful architecture design choices leads to our state-of-the-art depths. Moreover, our method does not preclude the use of 3D convolutions or additional cost volume and depth refinement techniques, allowing room for further improvements when compute is less restricted. Ultimately, our back-to-basics approach shows that high-quality depths are all you need for high-quality reconstructions.

Acknowledgments — We thank Aljaž Božič [2], Jiaming Sun [65] and Arda Düzçeker [12] for quickly providing useful information to help with baselines. Mohamed is funded by a Microsoft Research PhD Scholarship (MRL 2018-085).

References

1. Bhat, S.F., Alhashim, I., Wonka, P.: AdaBins: Depth estimation using adaptive bins. In: CVPR (2021)
2. Bozic, A., Palafox, P., Thies, J., Dai, A., Nießner, M.: TransformerFusion: Monocular RGB scene reconstruction using transformers. NeurIPS (2021)
3. Casser, V., Pirk, S., Mahjourian, R., Angelova, A.: Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In: AAAI (2019)
4. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: CVPR (2018)
5. Chen, Y., Schmid, C., Sminchisescu, C.: Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In: ICCV (2019)
6. Cheng, X., Wang, P., Yang, R.: Learning depth with convolutional spatial propagation network. PAMI (2019)
7. Choe, J., Im, S., Rameau, F., Kang, M., Kweon, I.S.: VolumeFusion: Deep depth fusion for 3D scene reconstruction. In: ICCV (2021)
8. Collins, R.T.: A space-sweep approach to true multi-image matching. In: CVPR (1996)
9. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (1996)
10. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: CVPR (2017)
11. Drory, A., Haubold, C., Avidan, S., Hamprecht, F.: Semi-global matching: A principled derivation in terms of message passing. In: German Conference on Pattern Recognition (2014)
12. 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)
13. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NeurIPS (2014)
14. Facil, J.M., Ummenhofer, B., Zhou, H., Montesano, L., Brox, T., Civera, J.: CAM-Convs: Camera-aware multi-scale convolutions for single-view depth. In: CVPR (2019)
15. Falcon, W., et al.: Pytorch lightning. GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning (2019)
16. Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., van der Smagt, P., Cremers, D., Brox, T.: FlowNet: Learning optical flow with convolutional networks. In: ICCV (2015)
17. Furukawa, Y., Hernández, C.: Multi-view stereo: A tutorial. Foundations and Trends in Computer Graphics and Vision (2015)
18. Glocker, B., Izadi, S., Shotton, J., Criminisi, A.: Real-time RGB-D camera relocalization. In: International Symposium on Mixed and Augmented Reality (ISMAR). IEEE (October 2013)
19. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR (2017)
20. Godard, C., Mac Aodha, O., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: ICCV (2019)

21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
22. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. PAMI (2007)
23. Hou, Y., Kannala, J., Solin, A.: Multi-view stereo by temporal nonparametric fusion. In: ICCV (2019)
24. Hu, J., Ozay, M., Zhang, Y., Okatani, T.: Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In: WACV (2018)
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. Ji, M., Gall, J., Zheng, H., Liu, Y., Fang, L.: SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. In: ICCV (2017)
28. Kähler, O., Prisacariu, V., Valentin, J., Murray, D.: Hierarchical voxel block hashing for efficient integration of depth images. IEEE Robotics and Automation Letters **1**(1), 192–197 (2015)
29. Kang, S.B., Szeliski, R., Chai, J.: Handling occlusions in dense multi-view stereo. In: CVPR (2001)
30. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. In: NeurIPS (2017)
31. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Eurographics. SGP '06, Eurographics Association (2006)
32. 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)
33. Kuznietsov, Y., Proesmans, M., Van Gool, L.: CoMoDA: Continuous monocular depth adaptation using past experiences. In: WACV (2021)
34. Lee, D.T., Schachter, B.J.: Two algorithms for constructing a delaunay triangulation. International Journal of Computer & Information Sciences **9**(3), 219–242 (1980)
35. Li, Z., Snavely, N.: MegaDepth: Learning single-view depth prediction from internet photos. In: CVPR (2018)
36. Liang, Z., Feng, Y., Guo, Y., Liu, H., Chen, W., Qiao, L., Zhou, L., Zhang, J.: Learning for disparity estimation through feature constancy. In: CVPR (2018)
37. 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)
38. Long, X., Liu, L., Li, W., Theobalt, C., Wang, W.: Multi-view depth estimation using epipolar spatio-temporal networks. In: CVPR (2021)
39. Long, X., Liu, L., Theobalt, C., Wang, W.: Occlusion-aware depth estimation with adaptive normal constraints. In: ECCV (2020)
40. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3D surface construction algorithm. ACM siggraph computer graphics (1987)
41. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
42. Luo, X., Huang, J.B., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. In: ACM SIGGRAPH (2020)
43. Marcel, S., Rodriguez, Y.: Torchvision the machine-vision package of torch. In: Proceedings of the 18th ACM International Conference on Multimedia. p. 1485–1488 (2010)

44. Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: CVPR (2016)
45. McCraith, R., Neumann, L., Zisserman, A., Vedaldi, A.: Monocular depth estimation with self-supervised instance adaptation. arXiv:2004.05821 (2020)
46. 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)
47. Newcombe, R.A., Izadi, S., Hilliges, O.: KinectFusion: Real-time dense surface mapping and tracking. In: UIST (2011)
48. Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: DTAM: Dense tracking and mapping in real-time. In: ICCV (2011)
49. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3D reconstruction at scale using voxel hashing. ACM Transactions on Graphics (ToG) (2013)
50. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: PyTorch: An imperative style, high-performance deep learning library. NeurIPS (2019)
51. Patil, V., Van Gansbeke, W., Dai, D., Van Gool, L.: Don't forget the past: Recurrent depth estimation from monocular video. In: IEEE Robotics and Automation Letters (2020)
52. 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)
53. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. PAMI (2020)
54. Rich, A., Stier, N., Sen, P., Höllerer, T.: 3dvnet: Multi-view depth prediction and volumetric refinement. In: International Conference on 3D Vision (3DV) (2021)
55. Runz, M., Buffier, M., Agapito, L.: MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In: ISMAR (2018)
56. Scharstein, D., Szeliski, R., Zabih, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In: IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001) (2001)
57. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: CVPR (2016)
58. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV) (2016)
59. Scona, R., Jaimez, M., Petillot, Y.R., Fallon, M., Cremers, D.: StaticFusion: Background reconstruction for dense RGB-D SLAM in dynamic environments. In: ICRA (2018)
60. 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)
61. Shu, C., Yu, K., Duan, Z., Yang, K.: Feature-metric loss for self-supervised learning of depth and egomotion. In: ECCV (2020)
62. Sinha, A., Murez, Z., Bartolozzi, J., Badrinarayanan, V., Rabinovich, A.: Deltas: Depth estimation by learning triangulation and densification of sparse points. In: ECCV (2020)
63. Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollhöfer, M.: DeepVoxels: Learning persistent 3D feature embeddings. In: CVPR (2019)

64. 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)
65. Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: NeuralRecon: Real-time coherent 3D reconstruction from monocular video. In: CVPR (2021)
66. Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V.: Mnasnet: Platform-aware neural architecture search for mobile. In: CVPR (2019)
67. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: ICML (2021)
68. Tananaev, D., Zhou, H., Ummenhofer, B., Brox, T.: Temporally consistent depth estimation in videos with recurrent architectures. In: ECCV Workshops (2018)
69. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. NeurIPS (2017)
70. Wang, K., Shen, S.: MVDepthNet: Real-time multiview depth estimation neural network. In: 3DV (2018)
71. Watson, J., Aodha, O.M., Prisacariu, V., Brostow, G., Firman, M.: The Temporal Opportunist: Self-Supervised Multi-Frame Monocular Depth. In: CVPR (2021)
72. Watson, J., Firman, M., Brostow, G.J., Turmukhambetov, D.: Self-supervised monocular depth hints. In: ICCV (2019)
73. Whelan, T., Kaess, M., Fallon, M., Johannsson, H., Leonard, J., McDonald, J.: Kintinuous: Spatially extended KinectFusion. In: RSS Workshop on RGB-D: Advanced Reasoning with Depth Camera (2012)
74. Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., Davison, A.: ElasticFusion: Dense SLAM without a pose graph. In: Robotics: Science and Systems (2015)
75. Wightman, R.: Pytorch image models. `https://github.com/rwightman/pytorch-image-models` (2019). https://doi.org/10.5281/zenodo.4414861
76. Wimbauer, F., Yang, N., von Stumberg, L., Zeller, N., Cremers, D.: MonoRec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera. In: CVPR (2021)
77. Yang, X., Zhou, L., Jiang, H., Tang, Z., Wang, Y., Bao, H., Zhang, G.: Mobile3DRecon: Real-time monocular 3D reconstruction on a mobile phone. IEEE Transactions on Visualization and Computer Graphics (2020)
78. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: MVSNet: Depth inference for unstructured multi-view stereo. In: ECCV (2018)
79. Yee, K., Chakrabarti, A.: Fast deep stereo with 2D convolutional processing of cost signatures. In: WACV (2020)
80. Yin, W., Liu, Y., Shen, C., Yan, Y.: Enforcing geometric constraints of virtual normal for depth prediction. In: ICCV (2019)
81. Yin, W., Zhang, J., Wang, O., Niklaus, S., Mai, L., Chen, S., Shen, C.: Learning to recover 3D scene shape from a single image. In: CVPR (2021)
82. Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. JMLR (2016)
83. Zhang, F., Prisacariu, V., Yang, R., Torr, P.H.: GA-Net: Guided aggregation net for end-to-end stereo matching. In: CVPR (2019)
84. Zhang, F., Qi, X., Yang, R., Prisacariu, V., Wah, B., Torr, P.: Domain-invariant stereo matching networks. In: ECCV (2020)
85. 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)

86. Zhao, Y., Kong, S., Fowlkes, C.: Camera pose matters: Improving depth prediction by mitigating pose distribution bias. In: CVPR (2021)
87. 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)