Atlas: End-to-End 3D Scene Reconstruction from Posed Images

Zak Murez[1], Tarrence van As[2*], James Bartolozzi[*], Ayan Sinha[1], Vijay Badrinarayanan[3*], and Andrew Rabinovich[2*]

[1] Magic Leap Inc., CA, USA zak@murez.com,asinha@magicleap.com
*Work done at Magic Leap bartolozzij@gmail.com
[2] InsideIQ Inc., CA, USA {tarrence,andrew}@insideiq.team
[3] Wayve.ai, London, UK vijay@wayve.ai

Abstract. We present an end-to-end 3D reconstruction method for a scene by directly regressing a truncated signed distance function (TSDF) from a set of posed RGB images. Traditional approaches to 3D reconstruction rely on an intermediate representation of depth maps prior to estimating a full 3D model of a scene. We hypothesize that a direct regression to 3D is more effective. A 2D CNN extracts features from each image independently which are then back-projected and accumulated into a voxel volume using the camera intrinsics and extrinsics. After accumulation, a 3D CNN refines the accumulated features and predicts the TSDF values. Additionally, semantic segmentation of the 3D model is obtained without significant computation. This approach is evaluated on the Scannet dataset where we significantly outperform state-of-the-art baselines (deep multiview stereo followed by traditional TSDF fusion) both quantitatively and qualitatively. We compare our 3D semantic segmentation to prior methods that use a depth sensor since no previous work attempts the problem with only RGB input.

Keywords: Multiview Stereo; TSDF; 3D Reconstruction

1 Introduction

Reconstructing the world around us is a long standing goal of computer vision. Recently many applications have emerged, such as autonomous driving and augmented reality, which rely heavily upon accurate 3D reconstructions of the surrounding environment. These reconstructions are often estimated by fusing depth measurements from special sensors, such as structured light, time of flight, or LIDAR, into 3D models. While these sensors can be extremely effective, they require special hardware making them more cumbersome and expensive than systems that rely solely on RGB cameras. Furthermore, they often suffer from noise and missing measurements due to low albedo and glossy surfaces as well as occlusion.

Another approach to 3D reconstruction is to use monocular [18,31,32], binocular [3,5] or multivew [23,27,28,51] stereo methods which take RGB images (one,

two, or multiple respectively) and predict depth maps for the images. Despite the plethora of recent research, these methods are still much less accurate than depth sensors, and do not produce satisfactory results when fused into a 3D model.

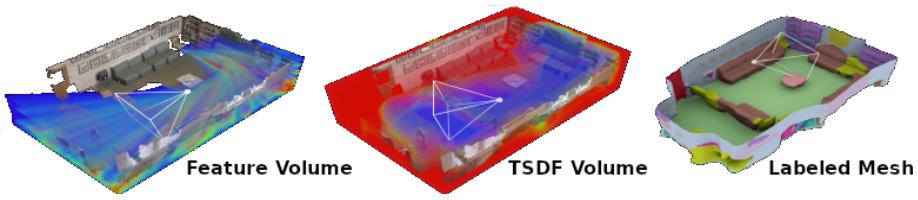

Fig. 1: Overview of our method. Features from each image are backprojected along rays and accumulated into a feature volume. Then a 3D CNN refines the features and regresses a TSDF volume. Finally a mesh is extracted from the TSDF. Semantic Labels can also be output.

In this work, we observe that depth maps are often just intermediate representations that are then fused with other depth maps into a full 3D model. As such, we propose a method that takes a sequence of RGB images and directly predicts a full 3D model in an end-to-end trainable manner. This allows the network to fuse more information and learn better geometric priors about the world, producing much better reconstructions. Furthermore, it reduces the complexity of the system by eliminating steps like frame selection, as well as reducing the required compute by amortizing the cost over the entire sequence.

Our method is inspired by two main lines of work: cost volume based multi view stereo [28,57] and Truncated Signed Distance Function (TSDF) refinement [12, 15]. Cost volume based multi view stereo methods construct a cost volume using a plane sweep. Here, a reference image is warped onto the target image for each of a fixed set of depth planes and stacked into a 3D cost volume. For the correct depth plane, the reference and target images will match while for other depth planes they will not. As such, the depth is computed by taking the argmin over the planes. This is made more robust by warping image features extracted by a CNN instead of the raw pixel measurements, and by filtering the cost volume with another CNN prior to taking the argmin.

TSDF refinement starts by fusing depth maps from a depth sensor into an initial voxel volume using TSDF fusion [10], in which each voxel stores the truncated signed distance to the nearest surface. Note that a triangulated mesh can then be extracted from this implicit representation by finding the zero crossing surface using marching cubes [34]. TSDF refinement methods [12, 15] take this noisy, incomplete TSDF as input and refine it by passing it through a 3D convolutional encoder-decoder network.

Similar to cost volume multi view stereo approaches, we start by using a 2D CNN to extract features from a sequence of RGB images. These features

are then back projected into a 3D volume using the known camera intrinsics and extrinsics. However, unlike cost volume approaches which back project the features into a target view frustum using image warping, we back project into a canonical voxel volume, where each pixel gets mapped to a ray in the volume (similar to [46]). This avoids the need to choose a target image and allows us to fuse an entire sequence of frames into a single volume. We fuse all the frames into the volume using a simple running average. Next, as in both cost volume and TSDF refinement, we pass our voxel volume through a 3D convolutional encoder-decoder to refine the features. Finally, as in TSDF refinement, our feature volume is used to regress the TSDF values at each voxel (see Figure 1).

We train and evaluate our network on real scans of indoor rooms from the Scannet [11] dataset. Our method significantly outperforms state-of-the-art multi view stereo baselines [28, 51] producing accurate and complete meshes.

As an additional bonus, for minimal extra compute, we can add an additional head to our 3D CNN and perform 3D semantic segmentation. While the problems of 3D semantic and instance segmentation have received a lot of attention recently [21, 25], all previous methods assume the depth was acquired using a depth sensor. Although our 3D segmentations are not competitive with the top performers on the Scannet benchmark leader board, we establish a strong baseline for the new task of 3D semantic segmentation from multi view RGB.

2 Related Work

2.1 3D reconstruction

Reconstructing a 3D model of a scene usually involves acquiring depth for a sequence of images and fusing the depth maps using a 3D data structure. The most common 3D structure for depth accumulation is the voxel volume used by TSDF fusion [10]. However, surfels (oriented point clouds) are starting to gain popularity [44, 55]. These methods are usually used with a depth sensor, but can also be applied to depth maps predicted from monocular or stereo images.

With the rise of deep learning, monocular depth estimation has seen huge improvements [18, 31, 32], however their accuracy is still far below state-of-the-art stereo methods. A popular classical approach to stereo [23] uses mutual information and semi global matching to compute the disparity between two images. Similar approaches have been incorporated into SLAM systems such as COLMAP [42, 43] and CNN-SLAM [50]. More recently, several end-to-end plane sweep algorithms have been proposed. DeepMVS [27] uses a patch matching network. MVDepthNet [51] constructs the cost volume from raw pixel measurements and performs 2D convolutions, treating the planes as feature channels. GPMVS [26] builds upon this and aggregates information into the cost volume over long sequences using a Gaussian process. MVSNet [57] and DPSNet [28] construct the cost volume from features extracted from the images using a 2D CNN. They then filter the cost volume using 3D convolutions on the 4D tensor. R-MVSNet [58] reduces the memory requirements of MVSNet by replacing the

3D CNN with a recurrent CNN, while P-MVSNet [6] starts with a low resolution MVSNet and then iteratively refines the estimate using their point flow module. All of these methods require choosing a target image to predict depth for and then finding suitable neighboring reference images. Recent binocular stereo methods [3, 5] use a similar cost volume approach, but avoid frame selection by using a fixed baseline stereo pair. Depth maps over a sequence are computed independently (or weakly coupled in the case of [26]). In contrast to these approaches, our method constructs a single coherent 3D model from a sequence of input images directly.

While TSDF fusion is simple and effective, it cannot reconstruct partially occluded geometry and requires averaging many measurements to reduce noise. As such, learned methods have been proposed to improve the fusion. OctNetFusion [40] uses a 3D encoder-decoder to aggregate multiple depth maps into a TSDF and shows results on single objects and portions of scans. ScanComplete [15] builds upon this and shows results for entire rooms. SG-NN [12] improves upon ScanComplete by increasing the resolution using sparse convolutions [21] and training using a novel self-supervised training scheme. 3D-SIC [24] focuses on 3D instance segmentation using region proposals and adds a per instance completion head. Routed fusion [54] uses 2D filtering and 3D convolutions in view frustums to improve aggregation of depth maps.

More similar in spirit to ours are networks that take one or more images and directly predict a 3D representation. 3D-R2N2 [9] encodes images to a latent space and then decodes a voxel occupancy volume. Octtree-Gen [49] increases the resolution by using an octtree data structure to improve the efficiency of 3D voxel volumes. Deep SDF [38] chooses to learn a generative model that can output an SDF value for any input position instead of discretizing the volume. These methods encode the input to a small latent code and report results on single objects, mostly from shapenet [4]. This small latent code is unlikely to contain enough information to be able to reconstruct an entire scene (follow up work [2], concurrent with ours, addresses this problem, but they do not apply it to RGB only reconstruction). Pix2Vox [56] encodes each image to a latent code and then decodes a voxel representation for each and then fuses them. This is similar to ours, but we explicitly model the 3D geometry of camera rays allowing us to learn better representations and scale to full scenes. SurfNet [45] learns a 3D offset from a template UV map of a surface. Point set generating networks [17] learns to generate point clouds with a fixed number of points. Pixel2Mesh++ [52] uses a graph convolutional network to directly predict a triangulated mesh. Mesh-RCNN [20] builds upon 2D object detection [22] and adds an additional head to predict a voxel occupancy grid for each instance and then refines them using a graph convolutional network on a mesh.

Back projecting image features into a voxel volume and then refining them using a 3D CNN has also been used for human pose estimation [29, 59]. These works regress 3D heat maps that are used to localize joint locations.

Deep Voxels [46] and the follow up work of scene representation networks [47] accumulate features into a 3D volume forming an unsupervised representation

of the world which can then be used to render novel views without the need to form explicit geometric intermediate representations.

2.2 3D Semantic Segmentation

In addition to reconstructing geometry, many applications require semantic labeling of the reconstruction to provide a richer representation. Broadly speaking, there are two approaches to solving this problem: 1) Predict semantics on 2D input images using a 2D segmentation network [1,7,22] and back project the labels to 3D [35–37] 2) Directly predict the semantic labels in the 3D space. All of these methods assume depth is provided by a depth sensor. A notable exception is Kimera [41], which uses multiview stereo [23] to predict depth, however, they only show results on synthetic data and ground truth 2D segmentations.

SGPN [53] formulates instance segmentation as a 3D point cloud clustering problem. Predicting a similarity matrix and clustering the 3D point cloud to derive semantic and instance labels. 3D-SIS [25] improves upon these approaches by fusing 2D features in a 3D representation. RGB images are encoded using a 2D CNN and back projected onto the 3D geometry reconstructed from depth maps. A 3D CNN is then used to predict 3D object bounding boxes and semantic labels. SSCN [21] predicts semantics on a high resolution voxel volume enabled by sparse convolutions.

In contrast to these approaches, we propose a strong baseline to the relatively untouched problem of 3D semantic segmentation without a depth sensor.

3 Method

Our method takes as input an arbitrary length sequence of RGB images, each with known intrinsics and pose. These images are passed through a 2D CNN backbone to extract features. The features are then back projected into a 3D voxel volume and accumulated using a running average. Once the image features have been fused into 3D, we regress a TSDF directly using a 3D CNN (See Fig. 2). We also experiment with adding an additional head to predict semantic segmentation.

3.1 Feature Volume Construction

Let $I_t \in \mathbb{R}^{3\times h\times w}$ be an image in a sequence of T RGB images. We extract features $F_t = F(I_t) \in \mathbb{R}^{c\times h\times w}$ using a standard 2D CNN where c is the feature dimension. These 2D features are then back projected into a 3D voxel volume using the known camera intrinsics and extrinsics, assuming a pinhole camera model. Consider a voxel volume $V \in \mathbb{R}^{c\times H\times W\times D}$

$$V_t(:, i, j, k) = F_t(:, \hat{i}, \hat{j}), \quad \text{with} \tag{1}$$

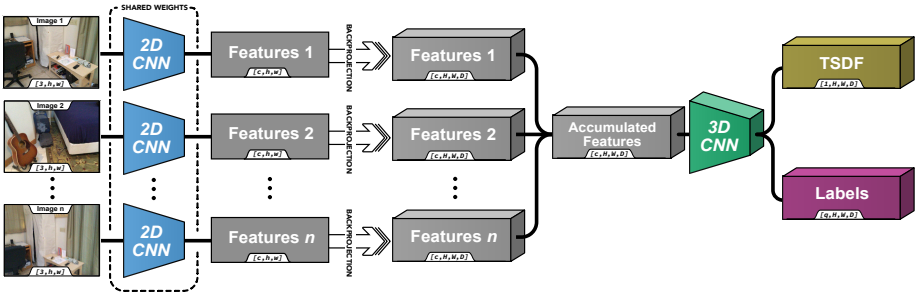

Fig. 2: Schematic of our method. Features are extracted from a sequence of images using a 2D CNN and then back projected into a 3D volume. These volumes are accumulated and then passed through a 3D CNN to directly regress a TSDF reconstruction of the scene. We can also jointly predict the 3D semantic segmentation of the scene.

$$\begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix} = \Pi K_t P_t \begin{bmatrix} i \\ j \\ k \\ 1 \end{bmatrix}, \tag{2}$$

where P_t and K_t are the extrinsics and intrinsics matrices for image t respectively, Π is the perspective mapping and : is the slice operator. Here (i, j, k) are the voxel coordinates in world space and $(\hat{i}, \hat{j})$ are the pixel coordinates in image space. Note that this means that all voxels along a camera ray are filled with the same features corresponding to that pixel.

These feature volumes are accumulated over the entire sequence using a weighted running average similar to TSDF fusion as follows:

$$\bar{V}_t = \frac{\bar{V}_{t-1}\bar{W}_{t-1} + V_t}{\bar{W}_{t-1} + W_t}, \tag{3}$$

$$\bar{W}_t = \bar{W}_{t-1} + W_t. \tag{4}$$

For the weights we use a binary mask $W_t(i, j, k) \in \{0, 1\}$ which stores if voxel (i, j, k) is inside or outside the view frustum of the camera.

3.2 3D Encoder-Decoder

Once the features are accumulated into the voxel volume, we use a 3D convolutional encoder-decoder network to refine the features and regress the output TSDF (Fig. 3). Each layer of the encoder and decoder uses a set of 3x3x3 residual blocks. Downsampling is implemented with 3x3x3 stride 2 convolution, while upsampling uses trilinear interpolation followed by a 1x1x1 convolution to change the feature dimension. The feature dimension is doubled with each downsampling and halved with each upsampling. All convolution layers are followed by

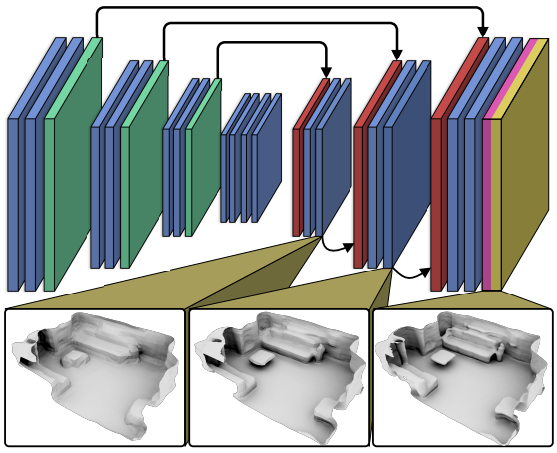

Fig. 3: Our 3D encoder-decoder architecture. Blue boxes denote residual blocks, green boxes are stride 2 convolutions and red boxes are trilinear upsampling. The arrows from the encoder to the decoder indicate skip connections. Our network predicts TSDFs in a coarse to fine manner with the previous resolution being used to sparsify the next resolution (shown as small arrows in the decoder).

batchnorm and relu. We also include additive skip connections from the encoder to the decoder.

At the topmost layer of the encoder-decoder, we use a 1x1x1 convolution followed by a tanh activation to regress the final TSDF values. For our semantic segmentation models we also include an additional 1x1x1 convolution to predict the segmentation logits.

We also include intermediate output heads at each decoded resolution prior to upsampling. These additional predictions are used both for intermediate supervision to help the network train faster, as well as to guide the later resolutions to focus on refining predictions near surfaces. At each resolution, any voxel that is predicted beyond a fraction (.99) of the truncation distance is clamped to one at the following resolutions. Furthermore, loss is only backpropageted for non-clamped voxels. Without this, the loss at the higher resolutions is dominated by the large number of empty space voxels and the network has a harder time learning fine details.

Note that since our features are back projected along entire rays, the voxel volume is filled densely and thus we cannot take advantage of sparse convolutions [21] in the encoder. However, the multiscale outputs can be used to sparsify the feature volumes in the decoder allowing for the use of sparse convolutions similar to [12]. In practice, we found that we were able to train our models at $4cm^3$ voxel resolution without the need for sparse convolutions.

4 Implementation Details

We use a Resnet50-FPN [33] followed by the merging method of [30] with 32 output feature channels as our 2D backbone. Our 3D CNN consists of a four scale resolution pyramid where we double the number of channels each time we half the resolution. The encoder consists of (1,2,3,4) residual blocks at each scale respectively, and the decoder consists of (3,2,1) residual blocks.

We supervise the multiscale TSDF reconstructions using ℓ_1 loss to the ground truth TSDF values. Following [14], we log-transform the predicted and target values before applying the ℓ_1 loss, and only backpropagate loss for voxels that were observed in the ground truth (i.e. have TSDF values strictly less than 1.) However, to prevent the network from hallucinating artifacts behind walls, outside the room, we also mark all the voxels where their entire vertical column is equal to 1 and penalize in these areas too. The intuition for this is that if the entire vertical column was not observed it was probably not within the room. To construct the ground truth TSDFs we run TSDF fusion at each resolution on the full sequences, prior to training.

We train the network end-to-end using 50 images selected randomly throughout the full sequence. We use a voxel size of $4cm^3$ with a grid of $(160 \times 160 \times 64)$ voxels, corresponding to a volume of $(6.4 \times 6.4 \times 2.56)$ meters. At test time, we accumulate the feature volumes in place (since we do not need to store the intermediate activations for backpropagation), allowing us to operate on arbitrary length sequences (often thousands of frames for ScanNet) and we use a 400x400x104 sized voxel grid corresponding to a volume of $(16 \times 16 \times 4.16)$ meters. We use the ADAM optimizer with a learning rate of 5e$-$4 and 16bit mixed precision operations. Training the network takes around 24 hours on 8 Titan RTX GPUs with a batch size of 8 (1 sequence per GPU) and synchronized batchnorm. Our model is implemented with PyTorch and PyTorch Lightning [16].

5 Results

We evaluate our method on ScanNet [11], which consists of 2.5M images across 707 distinct spaces. Standard train/validation/test splits are adopted. The 3D reconstructions are benchmarked using standard 2D depth metrics (Table 2) and 3D metrics (Table 3), which are defined in Table 1. We also show qualitative comparisons in Figure 6 where our method really stands out.

We compare our method to 4 state-of-the-art baselines: COLMAP [42, 43], MVDepthNet [51], GPMVS [26], and DPSNet [28]. For COLMAP we use the default dense reconstruction parameters but use the ground truth poses provided by Scannet. For each of the learned methods we fine tuned the models provided by the authors on Scannet. At inference time, 6 reference frames were selected temporally with stride 10 centered around the target view. We also mask the boundary pixels since the networks have visible edge effects that cause poor depth predictions here (leading to 92.8% completeness).

To evaluate these in 3D we fuse the predicted depth maps using two techniques: TSDF Fusion [10] and point cloud fusion. For COLMAP we use their

default point cloud fusion, while for the other methods we use the implementation of [19]. We found point cloud fusion was more robust to the outliers present in the depth predictions than our implementation of TSDF Fusion. As such, we only report the point cloud fusion results in Table 3 which are strictly better than the TSDF Fusion results (Note that the L_1 metric is computed using the TSDF Fusion approach as it is not computed in the point cloud fusion approach).

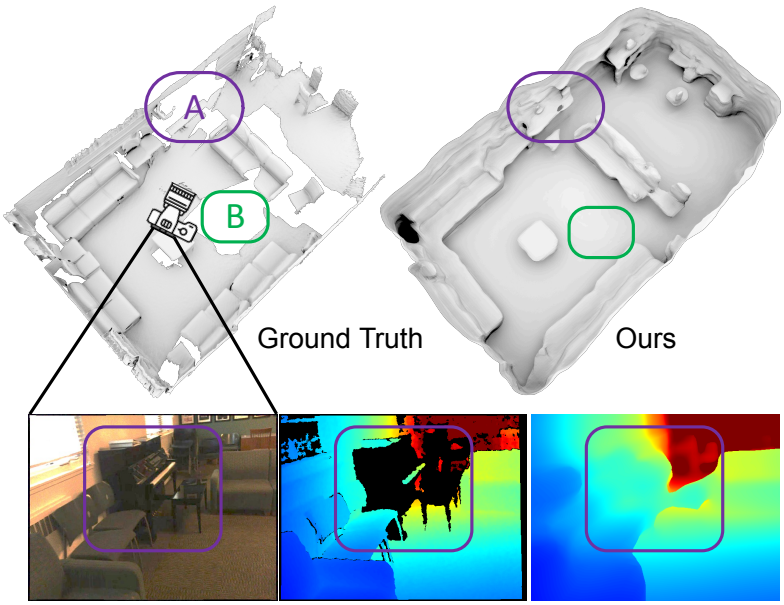

Fig. 4: Our method learns to fill holes that are missing from the ground truth. These holes arise from two causes: A) limitations of depth sensors on low albedo and specular surfaces, and B) unobserved regions caused by occlusion and incomplete scans. While other multiview stereo method often learn to predict depth for these troublesome surfaces, they are not able to complete unobserved geometry.

As seen in Figure 4 our method is able to fill holes that are missing from the ground truth. These holes arise from two causes: A) limitations of depth sensors on low albedo and specular surfaces, and B) unobserved regions caused by occlusion and incomplete scans. While other multiview stereo method often learn to predict depth for these troublesome surfaces, they are not able to complete unobserved geometry. On the other hand, since our method directly regresses the full TSDF for a scene, it is able to reason about and complete unobserved regions. However, this means that we must take extra care when evaluating the point cloud metrics, otherwise we will be falsely penalized in these regions. We remove geometry that was not observed in the ground truth by taking the rendered depth maps from our predicted mesh and re-fuse them using TSDF

Fusion into a trimmed mesh. This guarantees that there is no mesh in areas that were not observed in the ground truth.

Our method achieves state-of-the-art on about half of the metrics and is competitive on all metrics. However, as seen in Figure 6, qualitatively our results our significantly better than previous methods. While the L_1 metric on the TSDF seems to reflect this performance gap better, the inability of the other metrics to capture this indicates a need for additional more perceptual metrics.

As mentioned previously, we augment the existing 3D-CNN with a semantic segmentation head, requiring only a single $1 \times 1 \times 1$ convolution, to be able to not only reconstruct the 3D structure of the scene but also provide semantic labels to the surfaces. Since no prior work attempts to do 3D semantic segmentation from only RGB images, and there are no established benchmarks, we propose a new evaluation procedure. The semantic labels from the predicted mesh are transferred onto the ground truth mesh using nearest neighbor lookup on the vertices, and then the standard IOU metric can be used. The results are reported in Table 4 and Fig. 7 (note that this is an unfair comparison since all prior methods include depth as input).

Table 1: Definitions of metrics: n is the number of pixels with both valid ground truth and predictions, d and d^* are the predicted and ground truth depths (the predicted depth from our method is computed by rendering the predicted mesh). t and t^* are the predicted and ground truth TSDFs while p and p^* are the predicted and ground truth point clouds.

2D		3D	
Abs Rel	$\frac{1}{n}\sum \lvert d-d^*\rvert/d^*$	L1	$\text{mean}_{t^*<1}\lvert t-t^*\rvert$
Abs Diff	$\frac{1}{n}\sum \lvert d-d^*\rvert$	Acc	$\text{mean}_{p\in P}(\min_{p^*\in P^*} \lvert\lvert p-p^*\rvert\rvert)$
Sq Rel	$\frac{1}{n}\sum \lvert d-d^*\rvert^2/d^*$	Comp	$\text{mean}_{p^*\in P^*}(\min_{p\in P} \lvert\lvert p-p^*\rvert\rvert)$
RMSE	$\sqrt{\frac{1}{n}\sum \lvert d-d^*\rvert^2}$	Prec	$\text{mean}_{p\in P}(\min_{p^*\in P^*} \lvert\lvert p-p^*\rvert\rvert < .05)$
$\delta < 1.25^i$	$\frac{1}{n}\sum (\max(\frac{d}{d^*}, \frac{d^*}{d}) < 1.25^i)$	Recal	$\text{mean}_{p^*\in P^*}(\min_{p\in P} \lvert\lvert p-p^*\rvert\rvert < .05)$
Comp	% valid predictions	F-score	$\frac{2\times\text{Perc}\times\text{Recal}}{\text{Perc}+\text{Recal}}$

Table 2: 2D Depth Metrics

Method	AbsRel	AbsDiff	SqRel	RMSE	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Comp
COLMAP [43]	.137	.264	.138	.502	.834	.908	.938	.871
MVDepthNet [51]	.098	.191	.061	.293	.896	.977	.994	.928
GPMVS [26]	.130	.239	.339	.472	.906	.967	.980	.928
DPSNet [28]	.087	.158	**.035**	**.232**	.925	**.984**	**.995**	.928
Ours (plain)	**.061**	**.120**	.042	.248	**.940**	.972	.985	**.999**
Ours (semseg)	.065	.124	.043	.251	.936	.971	.986	**.999**

Table 3: 3D Geometry Metrics

Method	L_1	Acc	Comp	Prec	Recal	F-score
COLMAP [43]	.599	.069	.135	.634	**.505**	**.558**
MVDepthNet [51]	.518	.040	.240	.831	.208	.329
GPMVS [26]	.475	**.031**	.879	**.871**	.188	.304
DPSNet [28]	.421	.045	.284	.793	.223	.344
Ours (plain)	**.162**	.065	.130	.725	.383	.499
Ours (semseg)	.172	.074	**.124**	.711	.413	.520

Table 4: 3D Semantic Label Benchmark

Method	mIOU
ScanNet [11]	30.6
PointNet++ [39]	33.9
SPLATNet [48]	39.3
3DMV [13]	48.4
3DMV-FTSDF	50.1
PointNet++SW	52.3
SparseConvNet [21]	72.5
MinkowskiNet [8]	**73.4**
Ours	34.0

ScanNet 3D Semantic Segmentation metrics. We transfer our labels from the predicted mesh to the ground truth mesh using nearest neighbors.

From the results in Table 4 we see that our approach is surprisingly competitive with (and even beats some) prior methods that include depth as input. Having depth as an input makes the problem significantly easier because the only source of error is from the semantic predictions. In our case, in order to correctly label a vertex we must both predict the geometry correct as well as the semantic label. From Fig. 7 we can see that mistakes in geometry compounds with mistakes in semantics which leads to lower IOUs.

In Figure 5 we show an example of how our method degrades as the number of frames is reduced at inference time. We see that there is almost no degradation with as few as 25 frames. See accompanying video for more examples.

5.1 Inference Time

Since our method only requires running a small 2D CNN on each frame, the cost of running the large 3D CNN is amortized over a sequence of images. On the other hand, MVS methods must run all their compute on every frame. Note that they must also run depth map fusion to accumulate the depth maps into a mesh, but we do not include this additional time here. We report inference times using 2 neighbors. All models are run on a single NVidia TiTan RTX GPU. From

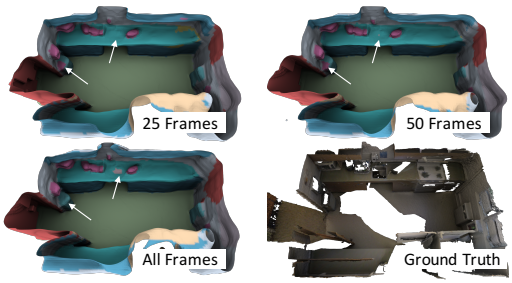

Fig. 5: Quality as a function of number of input frames at inference time. There is almost no degradation with as few as 25 frames (out of 784 total).

Table 5 we can see that after approximately 4 frames, ours becomes faster than DPSNet (note that most Scannet scenes are a few thousands of frames).

Table 5: Inference Time

Method	Per Frame Time (sec)	Per Sequence Time (sec)
COLMAP [43]	2.076	0
MVDepthNet [51]	0.048	0
GPMVS [26]	0.051	0
DPSNet [28]	0.322	0
Ours	.071	.840

6 Conclusions

In this work, we present a novel approach to 3D scene reconstruction. Notably, our approach does not require depth inputs; is unbounded temporally, allowing the integration of long frame sequences; completes unobserved geometry; and supports the efficient prediction of other quantities such as semantics. We have experimentally verified that the classical approach to 3D reconstruction via per view depth estimation is inferior to direct regression to a 3D model from an input RGB sequence. We have also demonstrated that without significant additional compute, a semantic segmentation objective can be added to the model to accurately label the resultant surfaces. In our future work, we aim to improve the back projection and accumulation process. One approach is to allow the network to learn where along a ray to place the features (instead of uniformly). This will improve the models ability to handle occlusions and large multi room scenes. We also plan to add additional tasks such as instance segmentation and intrinsic image decomposition. Our method is particularly well suited for intrinsic image decomposition because the network has the ability to reason with information from multiple views in 3D.

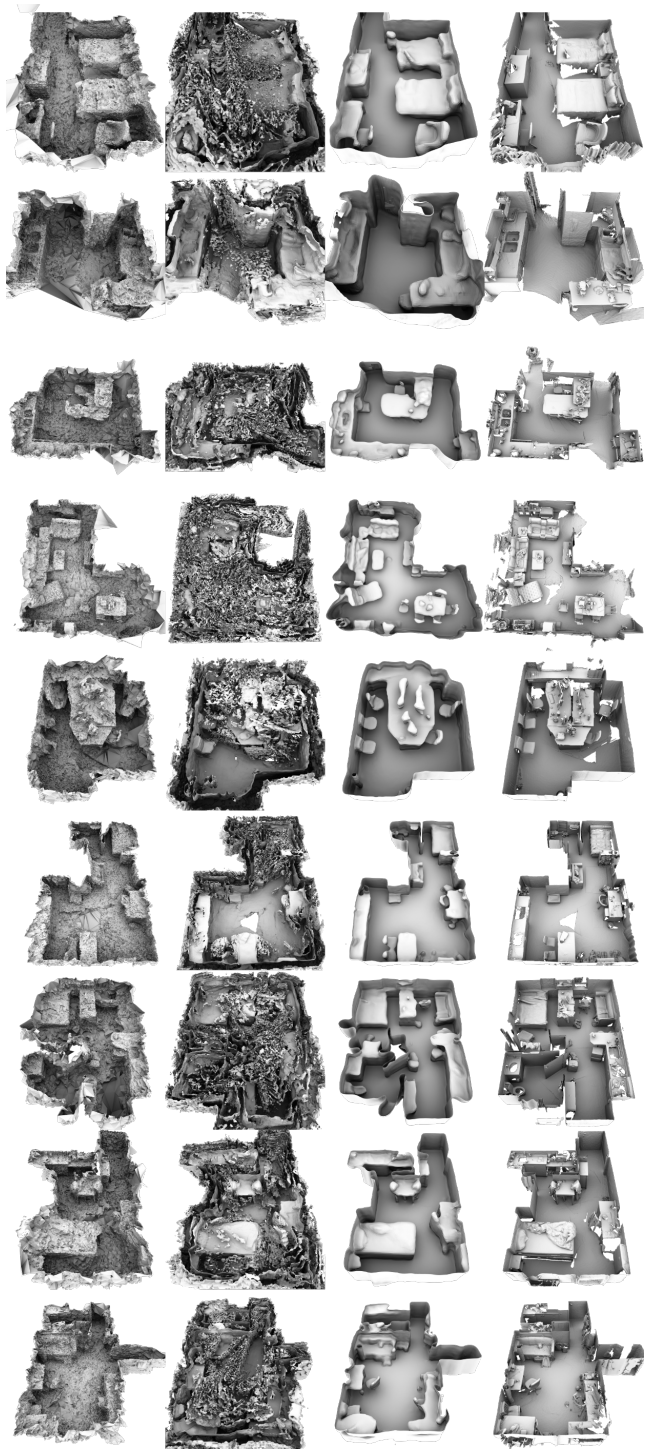

Fig. 6: Qualitative 3D reconstruction results.

Fig. 7: Qualitative 3D semantic segmentations. Left to right: Ours, our labels transferred to the ground truth mesh, ground truth labels. We are able to accurately segment the 3D scene despite not using a depth sensor.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation (2015)
2. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. arXiv preprint arXiv:2003.10983 (2020)
3. Chabra, R., Straub, J., Sweeney, C., Newcombe, R., Fuchs, H.: Stereodrnet: Dilated residual stereonet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 11786–11795 (2019)
4. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
5. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5410–5418 (2018)
6. Chen, R., Han, S., Xu, J., Su, H.: Point-based multi-view stereo network. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1538–1547 (2019)
7. Cheng, B., Collins, M.D., Zhu, Y., Liu, T., Huang, T.S., Adam, H., Chen, L.C.: Panoptic-deeplab. arXiv preprint arXiv:1910.04751 (2019)
8. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks (2019)
9. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: European conference on computer vision. pp. 628–644. Springer (2016)
10. 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. pp. 303–312 (1996)
11. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
12. Dai, A., Diller, C., Nießner, M.: Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans. arXiv preprint arXiv:1912.00036 (2019)
13. Dai, A., Nießner, M.: 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation (2018)
14. Dai, A., Qi, C.R., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis (2016)
15. Dai, A., Ritchie, D., Bokeloh, M., Reed, S., Sturm, J., Nießner, M.: Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4578–4587 (2018)
16. Falcon, W.: Pytorch lightning. GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning Cited by **3** (2019)
17. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
18. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2002–2011 (2018)

19. Galliani, S., Lasinger, K., Schindler, K.: Massively parallel multiview stereopsis by surface normal diffusion. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 873–881 (2015)
20. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9785–9795 (2019)
21. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9224–9232 (2018)
22. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
23. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. IEEE Transactions on pattern analysis and machine intelligence **30**(2), 328–341 (2007)
24. Hou, J., Dai, A., Nießner, M.: 3d-sic: 3d semantic instance completion for rgb-d scans. arXiv preprint arXiv:1904.12012 (2019)
25. Hou, J., Dai, A., Nießner, M.: 3d-sis: 3d semantic instance segmentation of rgb-d scans (2018)
26. Hou, Y., Kannala, J., Solin, A.: Multi-view stereo by temporal nonparametric fusion. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2651–2660 (2019)
27. Huang, P.H., Matzen, K., Kopf, J., Ahuja, N., Huang, J.B.: Deepmvs: Learning multi-view stereopsis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2821–2830 (2018)
28. Im, S., Jeon, H.G., Lin, S., Kweon, I.S.: Dpsnet: End-to-end deep plane sweep stereo. In: 7th International Conference on Learning Representations, ICLR 2019. International Conference on Learning Representations, ICLR (2019)
29. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7718–7727 (2019)
30. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6399–6408 (2019)
31. Lasinger, K., Ranftl, R., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. arXiv preprint arXiv:1907.01341 (2019)
32. Lee, J.H., Han, M.K., Ko, D.W., Suh, I.H.: From big to small: Multi-scale local planar guidance for monocular depth estimation. arXiv preprint arXiv:1907.10326 (2019)
33. 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)
34. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics **21**(4), 163–169 (1987)
35. McCormac, J., Clark, R., Bloesch, M., Davison, A., Leutenegger, S.: Fusion++: Volumetric object-level slam. In: 2018 international conference on 3D vision (3DV). pp. 32–41. IEEE (2018)
36. McCormac, J., Handa, A., Davison, A., Leutenegger, S.: Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In: 2017 IEEE International Conference on Robotics and automation (ICRA). pp. 4628–4635. IEEE (2017)

37. Narita, G., Seno, T., Ishikawa, T., Kaji, Y.: Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. arXiv preprint arXiv:1903.01177 (2019)
38. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019)
39. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space (2017)
40. Riegler, G., Osman Ulusoy, A., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3577–3586 (2017)
41. Rosinol, A., Abate, M., Chang, Y., Carlone, L.: Kimera: an open-source library for real-time metric-semantic localization and mapping. In: IEEE Intl. Conf. on Robotics and Automation (ICRA) (2020)
42. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
43. 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)
44. Schöps, T., Sattler, T., Pollefeys, M.: Surfelmeshing: Online surfel-based mesh reconstruction. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
45. Sinha, A., Unmesh, A., Huang, Q., Ramani, K.: Surfnet: Generating 3d shape surfaces using deep residual networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6040–6049 (2017)
46. Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollhofer, M.: Deepvoxels: Learning persistent 3d feature embeddings. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2437–2446 (2019)
47. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In: Advances in Neural Information Processing Systems (2019)
48. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: Splatnet: Sparse lattice networks for point cloud processing (2018)
49. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2088–2096 (2017)
50. Tateno, K., Tombari, F., Laina, I., Navab, N.: Cnn-slam: Real-time dense monocular slam with learned depth prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6243–6252 (2017)
51. Wang, K., Shen, S.: Mvdepthnet: real-time multiview depth estimation neural network. In: 2018 International Conference on 3D Vision (3DV). pp. 248–257. IEEE (2018)
52. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 52–67 (2018)
53. Wang, W., Yu, R., Huang, Q., Neumann, U.: Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In: In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. p. 2569–2578 (2018)
54. Weder, S., Schönberger, J., Pollefeys, M., Oswald, M.R.: Routedfusion: Learning real-time depth map fusion. arXiv preprint arXiv:2001.04388 (2020)

55. Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., Davison, A.: Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems
56. Xie, H., Yao, H., Sun, X., Zhou, S., Zhang, S.: Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2690–2698 (2019)
57. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 767–783 (2018)
58. Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., Quan, L.: Recurrent mvsnet for high-resolution multi-view stereo depth inference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5525–5534 (2019)
59. Zimmermann, C., Ceylan, D., Yang, J., Russell, B., Argus, M., Brox, T.: Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 813–822 (2019)