DeepSmooth: Efficient and Smooth Depth Completion

Sriram Krishna
Samsung Research
Bangalore, India
sriram.sk@samsung.com

Basavaraja Shanthappa Vandrotti
Samsung Research
Bangalore, India
b.vandrotti@samsung.com

Abstract

Accurate and consistent depth maps are essential for numerous applications across domains such as robotics, Augmented Reality and others. High-quality depth maps that are spatially and temporally consistent enable tasks such as Spatial Mapping, Video Portrait effects and more generally, 3D Scene Understanding. Depth data acquired from sensors is often incomplete and contains holes whereas depth estimated from RGB images can be inaccurate. This work focuses on Depth Completion, the task of filling holes in depth data using color images. Most work in depth completion formulates the task at the frame level, individually filling each frame's depth. This results in undesirable flickering artifacts when the RGB-D video stream is viewed as a whole and has detrimental effects on downstream tasks. We propose DeepSmooth, a model that spatio-temporally propagates information to fill in depth maps. Using an EfficientNet and pseudo 3D-Conv based architecture, and a loss function which enforces consistency across space and time, the proposed solution produces smooth depth maps.

1. Introduction

Depth estimation and related tasks have been long-standing problems in the Computer Vision community. Accurate depth maps are a fundamental requirement for numerous downstream tasks across domains as varied as Robotics and Augmented Reality (AR). Filling the holes in sparse depth measurements *i.e. depth completion*, increases the quality of the depth maps. This is critical when a complete 3D representation of the world is required.

The classical approach of estimating depth - Multi View Stereo (MVS), uses color images captured from multiple views to predict depth. Recent work in MVS [9, 13, 21, 34] estimate depth by feeding in images from multiple viewpoints to a deep neural network. However, MVS methods fail in texture-less areas such as plain walls, where pixel correspondences cannot be established. Furthermore, it can be expensive to acquire stereo or multiple views of a scene. Monocular Depth Estimation (MDE) addresses these challenges by estimating depth from a single RGB image. Monocular Depth Estimation is known to be an ill-posed problem, as it is not possible to geometrically recover depth from a single image [8]. Nonetheless, recent works [38] are able to overcome the ill-posedness of the problem and estimate depth, due to the priors embedded in the neural networks typically used for the task.

While significant progress has been made on estimating depth purely from RGB images, it remains challenging. Sensor-based approaches, on the other hand, use hardware directly to acquire depth. Over the previous decade, starting with the Microsoft Kinect, access to depth sensors has become democratized. Depth sensors are now commercially viable, widely available and can provide accurate depth maps. These sensors are of various kinds such as Time-of-Flight (ToF), Structured Light or LIDAR, and work well in various practical scenarios. However, these sensors have their own limitations. ToF sensors fail on surfaces with low reflectivity, LIDAR sensors provide accurate but sparse depth, etc.

Depth Completion aims at recovering dense depth from the sparse sensor depth and the semantic cues provided by a color image. The depth sensors provide an initial depth estimate, and color images are used to enhance the result. Contemporary work in depth completion [12, 16, 29, 37] perform depth completion by feeding an RGB image and sparse depth to a neural network. However, they have key drawbacks. They are typically designed to work on *individual images* rather than video streams, resulting in flickering depth over time, especially if the holes in the input are large and varying. Naively applying frame-level models to a video sequence results in a series of depth maps, which, while plausible on their own, result in noisy depth when viewed as a whole video. The noisiness in the depth maps propagates errors in downstream tasks, such as 3D Scene Reconstruction and Semantic Segmentation. While there has been work leveraging temporal information [3, 26], they do not exploit the semantic information present in color images captured in indoor scenarios.

Towards this goal, we propose *DeepSmooth*, a lightweight model designed for stable depth completion on RGB-D streams. Firstly, we design our network to handle the noise and holes inherent in a depth sensor. Semi-dense depth maps have holes covering large areas of the image with no valid values present and naively using standard convolutions gives poor results [33]. We make use of Atrous [5] convolutions to cover a wider a receptive field, and Gated [35] convolutions to encode a depth representation robust to missing values in the input depth map. Additionally, the holes in the input depth may vary over time as well. To accommodate this, our model uses a gated R2+1D Conv [32] temporal encoder, to aid in smooth predictions over a RGB-D stream.

Secondly, we propose enforcing *smoothness* spatially and temporally in depth predictions by means of a novel **Temporal Planar Consistency Loss**. Our motivation is to optimize a network to predict *planar* depth values that are stable over time. Stable planar depth has significant benefits in tasks utilizing depth maps, such as clean meshes when used in 3D Scene Reconstruction. State-of-the-art approaches are capable of predicting accurate depth maps with minimal error. We argue that consistent predictions are more important for most real-world applications than reducing the error by a few centimetres. By enforcing planar consistency over time and effectively propagating spatio-temporal context, we acquire smooth depth maps.

The primary contributions of this work are:

- We propose a novel architecture that is carefully designed to model RGB-D video-streams of indoor scenes with semi-dense input depth. Our network makes use of Atrous Gated convolutions to encode semi-dense depth, and gated R2+1D convolutions for spatio-temporal fusion.
- We propose a novel loss function, **Temporal Planar Consistency Loss**, that propagates *planes* in predicted depth over time, and minimizes the distance between the planes in consecutive predictions, enforcing spatial and temporal consistency simultaneously.
- Finally, we conduct extensive experiments on ScanNet and NYUv2, showing competitive results while improving upon other models in terms of temporal consistency. We also present a qualitative analysis on RGB+ToF data captured from a smartphone to demonstrate real-world performance.

In Section 2, we review related literature on depth estimation and completion. Section 3 presents the approach and design of our model. Section 4 describes the experimental setup *i.e.* implementation details, datasets and metrics. Section 5 presents our results and Section 6 concludes the paper.

2. Related Work

Depth estimation and depth completion are closely related tasks aimed at understanding the world in 3D. We provide a brief overview of algorithmic and learning-based methods that have been used for estimating depth, densifying sparse depth, and commonly used datasets.

Depth Estimation: Depth Estimation is fundamental in understanding the 3D world and has seen immense progress over the decades. Early approaches include Structure-from-Motion (SfM) and Multi View Stereo (MVS) which, broadly speaking, perform feature matching over a sequence to estimate depth [28]. Methods based on neural networks have increasingly become popular in recent years as they outperform classical approaches. They take the form of Monocular Depth Estimation, where only a single RGB image is used, or an MVS pipeline, where a posed video sequence is the input. Li *et al*. [19] propose a temporal consistency loss, directly minimizing the distance between consecutive predictions. Li *et al*. [18] exploit the structural regularities in indoor areas using the Manhattan normal constraint (surfaces align with dominant directions), and the co-planar constraint. The co-planar constraint states that the depth values are well-fitted by a plane in planar regions, and they devise a loss function minimizing the distance from the detected plane. Our Temporal Planar Consistency Loss differs in the fact that we propagate our planar detections over time, enforcing that they remain consistent over an entire sequence of frames.

Zhao *et al*. [38] provide an overview of contemporary deep learning based methods for monocular depth estimation. Ranft *et al*. [27] propose a robust depth estimation model trained by mixing 10 datasets from various domains. Deep MVS methods have continued to see progress as well, since the 6DoF pose can be acquired easily from most sensors. Hou *et al*. [13] build on top of previous work by Wang and Shen [34] by introducing a Gaussian Process at the bottleneck layer for fusing information from previous views. Duzceker *et al*. [9] propagate temporal information in a more explicit manner, by introducing a Convolutional LSTM between the encoder and the decoder.

Depth Completion: Similar to depth estimation, early depth completion methods used algorithmic approaches to densify sparse depth measurements [11]. Uhrig *et al*. [33] presented the first neural depth completion method by designing a custom convolution for handling sparse inputs. Zhang and Funkhouser [36] render depth images from the Matterport3D [4] dataset and create a benchmark by comparing the semi-dense sensor depth input with the dense depth rendered from the mesh. Nguyen and Yoo [26] propose a spatio-temporal approach for depth completion using a ConvLSTM cell, without exploiting the semantic cues in the color image, or modeling the holes in the depth image. Hu *et al*. [14] present a survey on deep depth completion

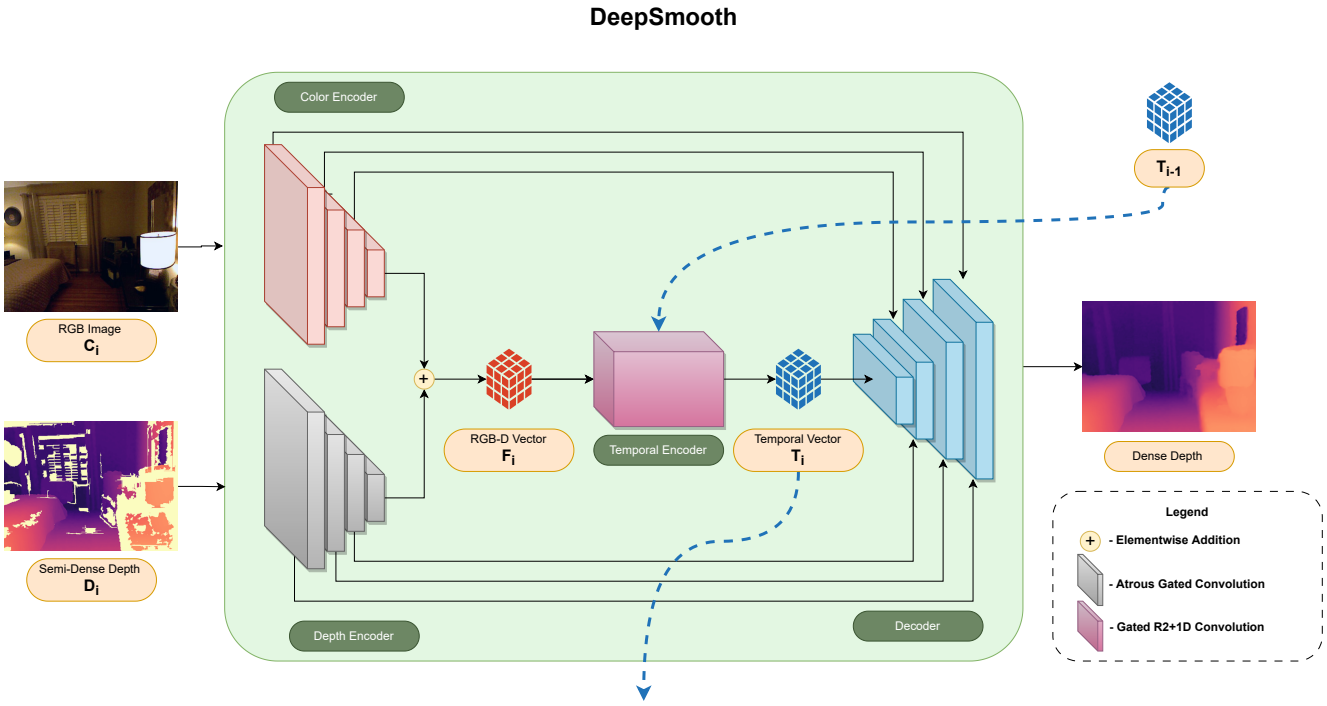

Figure 1. The architecture diagram of our proposed DeepSmooth model. At a given time-step i, the input to the network is the the RGB image C_i and the semi-dense sensor depth D_i. The RGB and the depth images are fed into EfficientNet-Lite backbones and the output is the RGB-D feature vector F_i. The temporal encoder combines F_i with the output of the temporal encoder at the previous time-step T_{i-1} to produce T_i which is provided as input to the succeeding time-step $i+1$. Finally, a decoder based on RefineNet produces a dense depth map as output.

and the evolution of the architectures used, primarily on outdoor depth completion. Outdoor depth completion is typically centered around autonomous driving where LIDAR sensors are commonly used. In indoor depth completion, commodity depth sensors such as ToF sensors are used, providing *semi-dense* depth *i.e.* depth with dense measurements in some areas and large holes in others. Senushkin *et al*. [29] propose decoder modulation specifically to handle the semi-dense input common in indoor depth completion. Zhang *et al*. [37] propose real-time depth completion for mobile devices. However, their system relies on an edge server *i.e.* it is not *on-device*, strictly limiting its applicability. Finally, Kam *et al*. [16] propose CostDCNet, a lightweight network that modifies the cost volume formulation in MVS setups for depth completion, achieving competitive results across different sparsity levels.

Datasets: Given the popularity of depth estimation/completion, a number of datasets and benchmarks have been proposed to evaluate proposed methods. The NYUv2 dataset [30] consists of RGB-D sequences from the Microsoft Kinect. Matterport3D [4] is commonly used, as semi-dense sensor depth is available from the dataset and dense depth is rendered from the mesh, as done by Zhang and Funkhouser [36]. However, Matterport3D captures are not continuous RGB-D sequences, and thus unsuitable for our task. ScanNet [7] is a large RGB-D video dataset of indoor scenes and our primary dataset for training. Like Matterport3D, ScanNet only provides sensor depths, and depth is rendered from the ground truth meshes.

3. DeepSmooth

In depth completion, the model is provided a color image and its corresponding sparse depth measurements to be filled. While this is sufficient for frame-level depth completion, we approach the task from the perspective of RGB-D video streams. We also make use of the camera pose while training and maintain temporal context to predict *smooth* depth.

Drawing on insights from depth completion over the years [14], we propose a lightweight dual branch encoder-decoder, enhanced with temporal propagation. Our architectural design is shown in Fig. 1, a dual branch encoder-decoder, which generates embeddings for RGB and depth separately while maintaining a reasonably sized model. A primary design decision concerns the way in which the

color and depth modalities are fused in a deep network. Naive early fusion approaches directly concatenate the two modalities [23], but nonetheless achieve good results. In late fusion, intermediate representations are fused and performance is generally better than early fusion due to explicitly handling the two modalities [14]. Our color and depth encoders are both networks based on EfficientNet-Lite [22]. For smoother depth completion over time, we integrate a *temporal encoder*. Finally, our decoder is based on the RefineNet [20,25] architecture, with skip connections from the encoder at multiple scales. At inference time, the input to our network is a color image and the semi-dense depth at that time instant, and outputs a dense depth map.

3.1. Color Encoder

The color encoder is fed the RGB image and it returns a feature representation. We opt to use a network trained for monocular depth estimation as our color backbone, generating features more conducive to the task of depth completion. We use a recent monocular depth estimation model MiDaS [27] as our color backbone. The MiDaS model is trained on a diverse set of datasets across domains, improving the quality of depth estimation. We use the EfficientNet-lite backbone of this model as our color encoder. The EfficientNet-lite architecture [22] optimize the EfficientNet family of models for real-time inference on a low-powered device.

3.2. Depth Encoder

In depth completion, there is a complication in encoding depth as the input depth is semi-dense *i.e.* contains holes. We encode the semi-dense depth by making use of atrous gated convolutions. Uhrig *et al*. [33] introduced sparse convolutions, which mask invalid pixels in the sparse input and normalize the convolutions appropriately. Yu *et al*. [35] generalize this idea to *Gated Convolutions*, wherein soft masks are automatically learned from data. Gated Convolutions have been used effectively in both image in-painting and depth completion [15]. The Gated Convolution is defined as follows:

$$Gate_X = \sigma(Conv_g(X)) \tag{1}$$

$$Feat_X = \phi(Conv_f(X)) \tag{2}$$

$$Out_X = Gate_X \odot Feat_X \tag{3}$$

Where, $Conv_g$ and $Conv_f$ are two convolutional filters, X the feature values, σ represents the sigmoid function, thus constraining the gating output in $[0-1]$, ϕ represents an arbitrary activation function and $\odot$ represents the element-wise multiplication between the gating output and the feature output. Our depth encoder is similar in structure to the color encoder, except that all convolutions have been replaced by gated convolutions. As gated convolutions effectively double the size of the depth encoder, we opt to use Atrous convolutions [5] to reduce the size of the model. For feature fusion, we opted against concatenation in favour of addition of features to keep our model lightweight.

3.3. Temporal Encoder

The outputs of the dual encoders are fused before being fed into the temporal encoder, designed to propagate information temporally, so as to provide smooth depth over time. The temporal encoder takes the form of a series of R2+1D convolutions. The R2+1D Conv is a factorization of the 3D Convolution into sequential 2D and 1D convs, operating over the spatial and temporal states independently [32]. This factorization allows for convolving over the 4D volume without the memory and computational requirements of a full 3D convolution. The holes in the input depth are unstable and vary over time, and in order to make our temporal representation robust to noise, we modify the R2+1D convolutions to utilize gated convolutions. Thus, we use Gated R2+1D convolutions to model the temporal noise. Along with the feature vector of the current vector, the output of the temporal encoder of the previous time-step is also provided as input.

3.4. Decoder

The output of the temporal encoders represents information stored in the incoming RGB-D video. This feature vector is fed into the decoder, based on a variant of RefineNet [20]. The RefineNet architecture introduces two components, the residual convolution unit (RCU), a residual unit without batch normalization, and chained residual pooling (CRP), a chain of convolutions and pooling layers. Nekraskov *et al*. [25] propose a lightweight version of RefineNet for real-time semantic segmentation, by reducing the size of the convolution kernels and dropping the RCU entirely. As in the original model, skip connections from the encoder at various levels to refine the decoder's output. RefineNet has been used effectively in various tasks, including depth estimation [27] and depth completion [29].

3.5. Temporal Planar Consistency

In training our network, we make use of a hybrid loss function, combining the traditional L1 loss with our novel **Temporal Planar Consistency** (TPC) to enforce smoother depth prediction. Li *et al*. [19] propose a temporal consistency loss as the difference between two consecutive predictions, the rationale being that they must not differ drastically and be *noisy*. Li *et al*. [18] propose a co-planar loss, on the observation that depth pixels lying in a plane should be well fitted by a plane equation.

We observe that these are both desired characteristics of an ideal depth completion model: the predicted depth must be consistent over time, and well-fitted by planes in the scene. This motivates our Temporal Planar Consistency Loss, to enforce these characteristics. The prediction at time $i-1$, D_{i-1} is warped forward ($\hat{D}_{i-1}$) as described in Duzceker *et al*. [9]. We make use of the available 6DoF pose to reproject the depth map to account for the change in viewpoints. After reprojection, we fit planes to the warped depth using RANSAC. For pixels in detected planes, the depth value is *flattened* such that it fits the plane equation exactly. The new depth value z^{flat} of a point in 3D space (x, y, z) is given by solving for z in the plane equation:

$$z^{flat} = -1(Ax + By + D)/C \tag{4}$$

Where, (A, B, C, D) are the coefficients of the detected plane $Ax + By + Cz + D = 0$. Finally, we compute the L1 loss between the warped "flattened" depth and the prediction at time i.

$$\mathcal{L}_{TPC} = ||D_i - \hat{D}_{i-1}^{flat}|| \tag{5}$$

We note here that we are optimizing our network to predict an *idealized* depth. This ideal depth flattens minor irregularities present in the real world into smooth planes. By design, this encourages our model to avoid predicting fine details leading to a slight increase in error. We argue that the increased consistency in output is more desirable for most real-world applications which do not require millimeter-level accuracy. Our complete training loss $\mathcal{L}$ is a weighted combination of the L1 loss and the temporal planar consistency loss:

$$\mathcal{L} = \lambda\mathcal{L}_{L_1} + (1 - \lambda)\mathcal{L}_{TPC} \tag{6}$$

Where λ controls the weight of the two loss terms. Empirically, we find that setting $\lambda = 0.9$ is sufficient to encourage consistency while ensuring the network is trained on the primary L1 loss objective.

3.6. Training Setup

We train on the ScanNet dataset, a large indoor RGB-D dataset where scenes are captured in a continuous video sequence. However, ScanNet does not make the ground truth depth available and only provides the sensor depth. Since many existing datasets don't provide both these types of data, Senushkin *et al*. [29] propose a *depth corruption strategy*, wherein the sensor depth is artificially corrupted and holes are introduced. This corrupted data serves as the training data and the provided sensor depth takes the place of the ground truth depth data. Senushkin *et al*. evaluate a number of algorithms for depth corruption and empirically recommend Felzenszwalb's graph-based segmentation algorithm [10]. The color image is segmented with Felzenszwalb's algorithm and segments with area less than a threshold are masked in the depth image. In our case, we mask all segments in the image whose area is less than the threshold. While training, the model is fed the *corrupted sensor* depth and the target is the *actual sensor* depth ($\mathcal{C} \rightarrow \mathbb{S}$). An alternative to this setup is to use the sensor depth for training and rendered depth as the target ($\mathcal{S} \rightarrow \mathbb{R}$). However, we find that rendered depth is similar to the sensor depth and does not provide a strong enough signal for learning. Furthermore, depth values of faraway objects were missing from the rendered depth and thus we opt to use the $\mathcal{C} \rightarrow \mathbb{S}$ setup instead.

4. Experiments

Our model was implemented in PyTorch, and trained end-to-end on ScanNet for 10 epochs. Training was performed on a cluster of 8 NVIDIA P40 GPUs. The Adam optimizer was used with a learning rate of $1e^{-4}$. As other methods are typically trained on Matterport3D, a direct comparison is unfair. To remedy this, we retrain all models using their official implementations on the ScanNet dataset.

4.1. Datasets

Our primary experiments are conducted on the following datasets: ScanNet [7] and NYUv2 [30]. Other popular datasets such as Matterport3D [4] and ToF18k [37] are not suited to our task as the dataset is not captured as an RGB-D video stream.

For evaluating on ScanNet, the sensor depth is provided as input and evaluated against depth rendered from the ground truth mesh ($\mathcal{S} \rightarrow \mathbb{R}$).

To evaluate the generalizability of the models, they are tested on the NYUv2 dataset without finetuning. Traditionally, models evaluated against NYUv2 use its *labeled* split, a subset of the raw dataset which provides semi-dense depth maps whose holes are filled using the colorization scheme of Levin *et al*. [17]. The labeled split of NYUv2 does not contain full sequences and is unsuitable. Instead, we evaluate with the *raw* dump of NYUv2, containing full sequences. Similar to existing evaluations, we in-paint the depths using the colorization of Levin *et al*. We take the raw sensor depth as input and evaluate it against the in-painted depth. NYUv2 also does not make the pose available, and following Teed *et al*. [31], we use pose estimated from ORB-SLAM [24].

Finally, we also evaluate against **ARCore**, an RGB-D sequence of an indoor room, captured on an Android smartphone (Samsung Note 10+ 5G) with the ARCore library [1].

4.2. Metrics

In line with recent work [29, 37, 38], we quantitatively evaluate using the following metrics - root mean squared error (RMSE), mean absolute error (MAE) and the δ metric.

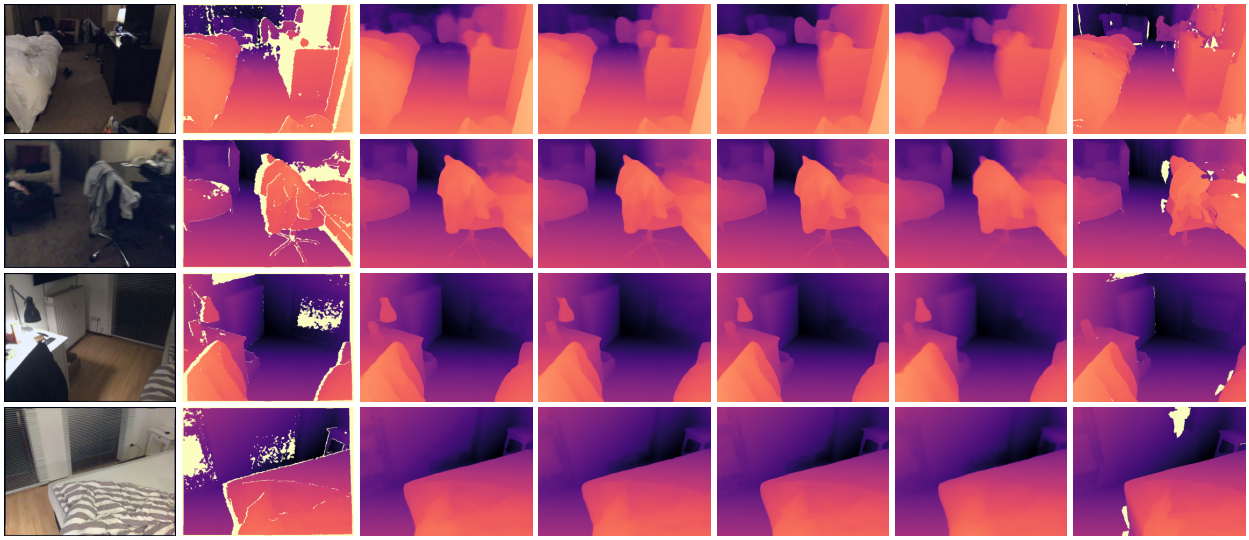

Figure 2. Qualitative Results on the ScanNet test set. Rows 1-2 belong to the same scene, with the image in Row 2 being captured after an arbitrary amount of time after Row 1, to illustrate temporal stability of predictions. The same holds true for Rows 3-4.

	TC↑	RMSE↓	MAE↓	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑	$\delta_{1.25^2}$ ↑	$\delta_{1.25^3}$ ↑
CostDCNet [16]	0.989	0.145	0.039	**0.928**	0.952	0.973	0.987	0.993
DM-LRN [29]	0.990	0.137	0.036	**0.928**	**0.954**	0.974	**0.988**	**0.994**
inDepth [37]	0.990	0.137	**0.035**	**0.928**	**0.954**	0.974	**0.988**	**0.994**
DeepSmooth-simp w/out TPC	0.991	0.138	0.039	0.907	0.948	0.974	**0.988**	**0.994**
DeepSmooth w/out TPC	0.991	**0.136**	0.038	0.910	0.951	**0.975**	**0.988**	**0.994**
DeepSmooth	**0.992**	0.142	0.043	0.886	0.942	0.973	0.987	**0.994**

Table 1. ScanNet: Quantitative results on the ScanNet test set. RMSE and MAE are measured in meters.

δ_i indicates the percentage of pixels where the relative error is less than a threshold i. It is evaluated at multiple levels, with i set to 1.05, 1.10, 1.25, 1.25^2 and 1.25^3. We also compute **temporal consistency** [19] to evaluate the *smoothness* of the depth prediction over time. It quantifies the percentage of pixels which are stable over time *i.e.* the ratio of change between consecutive frames does not go beyond a threshold. For a single image, the relative Temporal Consistency (rTC) is given by:

$$rTC = \frac{1}{sum(M)} M \left[max \left(\frac{D_i}{\hat{D}_{i-1}}, \frac{\hat{D}_{i-1}}{D_i} \right) < thr \right] \quad (7)$$

Where, D_i represents current depth prediction, $\hat{D}_{i-1}$ represents the depth prediction of the previous frame warped forward by the relative camera pose, and M is the occlusion mask. The occlusion mask M is generated from the color images. A pixel is considered as occluded and masked if there is a significant colour difference between corresponding pixels at time $i-1$ and i. Following Li *et al.* [19], we set the threshold $thr = 1.21$, *i.e.* only pixels whose values vary by less than 21% are counted as valid. The temporal consistency of the entire scene (TC) is given by averaging the relative temporal consistency rTC over all the frames of the scene.

$$TC = \sum_{i=0}^{n} rTC_i \quad (8)$$

5. Results

We evaluate the best performing methods of recent years on selected datasets. Senushkin *et al.* propose DM-LRN (Decoder Modulation - Lightweight RefineNet) [29], a network that modulates the decoder with the holes in the depth maps. Zhang *et al.* [37] propose inDepth, a DNN with dilated convolutions to have a larger receptive field. Kam *et al.* [16] develop CostDCNet, a depth completion network taking inspiration from the cost volume technique in Multi View Stereo pipelines. We also evaluate *DeepSmooth w/out TPC* - the DeepSmooth model trained with only the L1 Loss

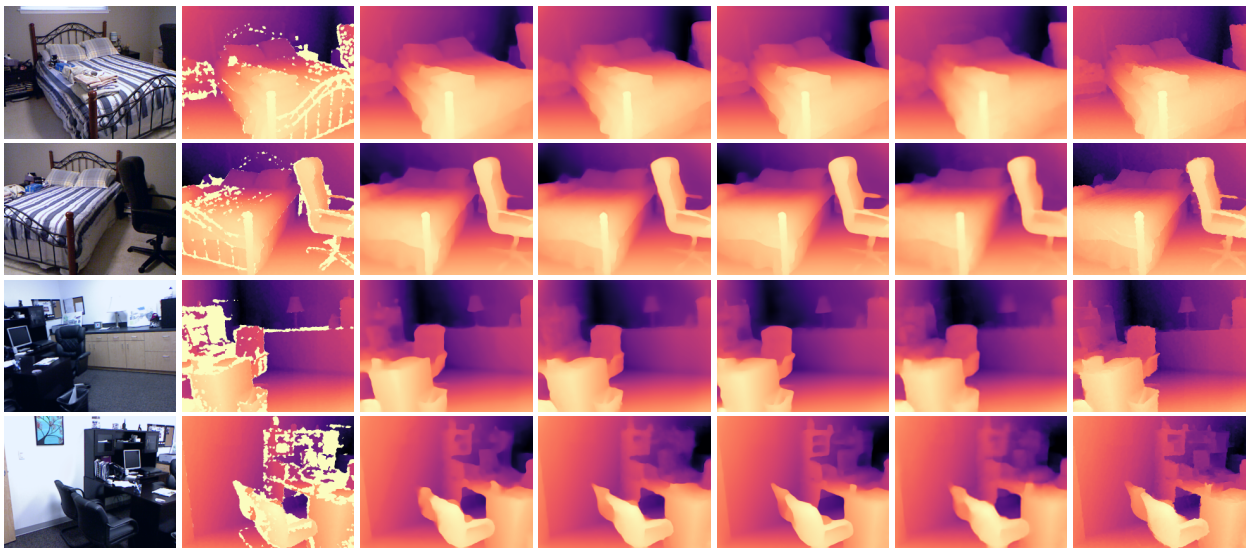

Figure 3. Qualitative Results on the NYUv2 Raw set. Rows 1-2 belong to the same scene, with the image in Row 2 being captured after an arbitrary amount of time after Row 1, to illustrate temporal stability of predictions. The same holds true for Rows 3-4.

	TC↑	RMSE↓	MAE↓	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑	$\delta_{1.25^2}$ ↑	$\delta_{1.25^3}$ ↑
CostDCNet [16]	0.993	**0.205**	**0.061**	**0.930**	**0.965**	**0.987**	**0.997**	**0.999**
DM-LRN [29]	0.994	0.235	0.069	0.921	0.958	0.981	0.994	0.998
inDepth [37]	0.993	0.227	0.069	0.919	0.961	0.981	0.993	0.998
DeepSmooth-simp w/out TPC	0.993	0.223	0.077	0.896	0.956	0.983	0.995	**0.999**
DeepSmooth w/out TPC	0.994	0.236	0.074	0.911	0.957	0.982	0.995	0.998
DeepSmooth	**0.995**	0.225	0.077	0.899	0.960	0.985	0.996	**0.999**

Table 2. NYUv2: Quantitative results on the NYUv2 raw dataset. RMSE and MAE are measured in meters.

- and *DeepSmooth-simp w/out TPC*. DeepSmooth-simp is a simplified version of DeepSmooth, with the temporal encoder removed.

	Parameters (million)
costDCNet [16]	1.8
DM-LRN [29]	22.3
inDepth [37]	54.6
DeepSmooth	20.4

Table 3. Comparison of model sizes

Table 3 shows a comparison of model sizes. inDepth [37] is the largest model by far, and is designed to run on a remote server for fast inference, rather than on-device. CostDCNet [16] is a much smaller model, but utilizes Minkowski convolutions [6], which does not support conversion to ONNX [2], a standard format for deployment of models across various runtimes and accelerators.

ScanNet: We evaluate the methods on the ScanNet test set, with the input being the sensor depth and the evaluation target being the depth rendered from the mesh. A visual representation of the results is in Fig. 2. Our method predicts smooth depth maps which are planar in nature. Compared to other methods, DeepSmooth predicts lesser fine details, but much less noise as well. Quantitative results are present in Table 1. DeepSmooth achieves state-of-the-art results on temporal consistency (TC). However, due to the smoothening nature of our Temporal Planar Consistency Loss, DeepSmooth has slightly higher error (of the order of a few millimeters) compared to other methods. However, the representation is much smoother. We argue that this is a more advantageous representation in real-world applications, maintaining high fidelity where required and flattening depth into planes when not needed.

NYUv2: Previous works evaluate on the NYUv2 labeled split, a subset containing around 1500 images and their inpainted depth maps. As mentioned in Sec. 4, this contains discrete images and not a continuous sequence. Thus, we

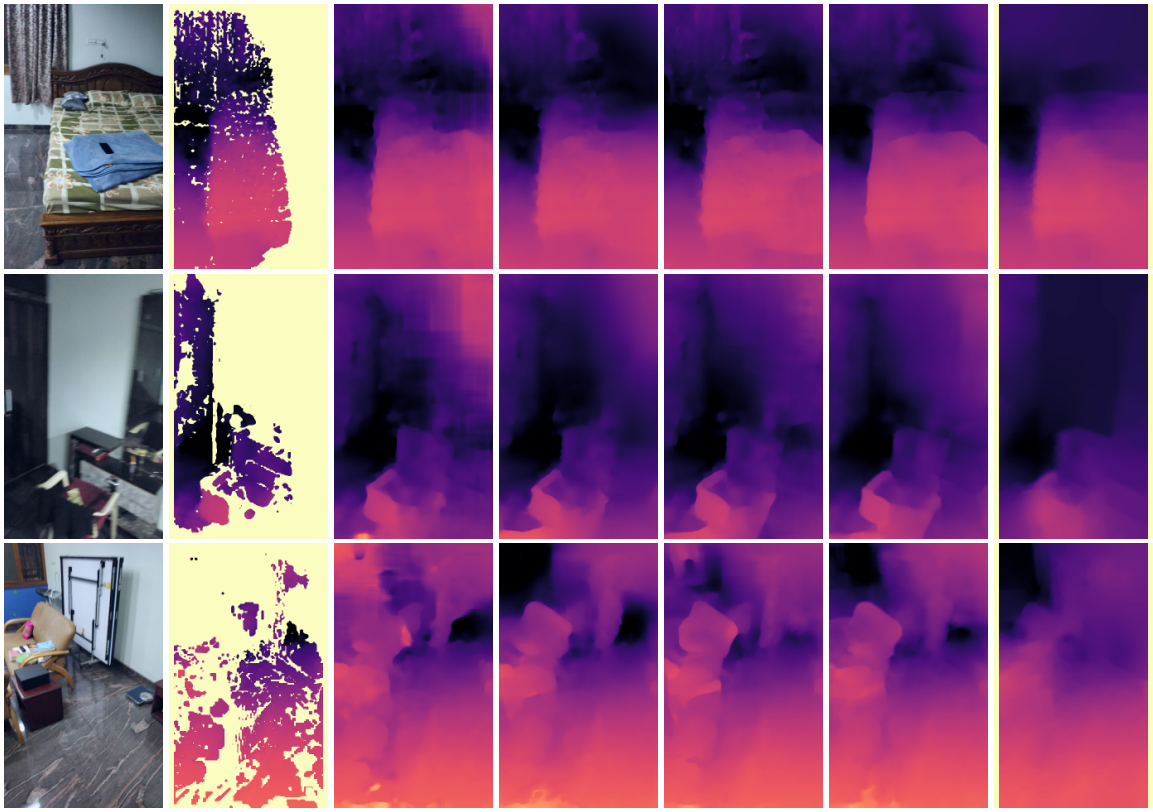

Figure 4. Qualitative Results on the ARCore dataset. Input depth is the depth provided by ARCore's **Raw Depth API**, provided with minimal processing. The Full Depth represents data acquired with ARCore's **Depth API**.

utilize the raw split and evaluate on a subset of the same. Qualitative and quantitative results are present in Fig. 3 and Table 2. While costDCNet [16] shows strong results across metrics, we note that that the difference is small in absolute terms. DeepSmooth shows higher temporal consistency and this is reflected in the planar output in the images in Fig. 3.

ARCore: Depth completion models work in tandem with depth sensors, and the quality of a model is determined by it's ability to complete the depth from a commodity depth sensor. We captured an indoor scene with an Android smartphone containing a ToF depth sensor (Samsung Note 10+ 5G). The input depth to the model comes from ARCore's *Raw Depth API*, which provides depth with minimal processing *i.e.* raw. Instead of ground truth, we have depth from ARCore's *Depth API*, which interpolates the raw depth using a proprietary algorithm. A qualitative comparison of the results are shown in Fig. 4. Due to the domain shift and a much larger number of missing pixels, we observe a significant degradation in performance across models. ARCore's Full Depth API aggressively smoothens the image, causing the loss of structure in the image. Our method is able to fill in large holes in the depth while preserving the structure of larger objects in the scene.

6. Conclusion

We propose DeepSmooth, a novel depth completion method that generates dense depth maps from semi-dense depth, using a novel dual-branch encoder-decoder network. Our model is designed for video streams by integrating R2+1D convolutions into the network for stable predictions over time. We enforce spatial and temporal consistency by means of a novel loss function, Temporal Planar Consistency Loss. Furthermore, our model is designed to be lightweight, a critical need for applications in Augmented Reality and Robotics.

We perform quantitative and qualitative experiments on the ScanNet and NYUv2 datasets and show competitive results across the board while achieving state-of-the-art results on temporal consistency. We argue that temporal consistency is a more desirable characteristic in many applications where millimeter-level accuracy is not required. Commodity depth sensors are limited in their accuracy due to engineering constraints and cost, and providing high quality depth maps through depth completion unlocks the way towards applications requiring 3D understanding of the world around us.

References

[1] ARCore - Google Developers. https://developers.google.com/ar. Accessed: 2022-08-15. 5

[2] Junjie Bai, Fang Lu, Ke Zhang, et al. Onnx: Open neural network exchange. https://github.com/onnx/onnx, 2019. 7

[3] Massimo Camplani and Luis Salgado. Efficient spatio-temporal hole filling strategy for kinect depth maps. In *Three-dimensional image processing (3DIP) and applications Ii*, volume 8290, pages 127–136. SPIE, 2012. 1

[4] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 2, 3, 5

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2, 4

[6] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019. 7

[7] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 3, 5

[8] Tom van Dijk and Guido de Croon. How do neural networks see depth in single images? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2183–2191, 2019. 1

[9] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15324–15333, 2021. 1, 2, 5

[10] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004. 5

[11] Simon Hawe, Martin Kleinsteuber, and Klaus Diepold. Dense disparity maps from sparse disparity measurements. In *2011 International Conference on Computer Vision*, pages 2126–2133. IEEE, 2011. 2

[12] Girish Hegde, Tushar Pharale, Soumya Jahagirdar, Vaishakh Nargund, Ramesh Ashok Tabib, Uma Mudenagudi, Basavaraja Vandrotti, and Ankit Dhiman. Deepdnet: Deep dense network for depth completion task. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2190–2199, 2021. 1

[13] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2651–2660, 2019. 1, 2

[14] Junjie Hu, Chenyu Bao, Mete Ozay, Chenyou Fan, Qing Gao, Honghai Liu, and Tin Lun Lam. Deep depth completion: A survey. *arXiv preprint arXiv:2205.05335*, 2022. 2, 3, 4

[15] Yu-Kai Huang, Tsung-Han Wu, Yueh-Cheng Liu, and Winston H Hsu. Indoor depth completion with boundary consistency and self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 4

[16] Jaewon Kam, Jungeon Kim, Soongjin Kim, Jaesik Park, and Seungyong Lee. Costdcnet: Cost volume based depth completion for a single rgb-d image. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 257–274. Springer, 2022. 1, 3, 6, 7, 8

[17] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. In *ACM SIGGRAPH 2004 Papers*, pages 689–694. ACM, 2004. 5

[18] Boying Li, Yuan Huang, Zeyu Liu, Danping Zou, and Wenxian Yu. Structdepth: Leveraging the structural regularities for self-supervised indoor depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12663–12673, 2021. 2, 4

[19] Siyuan Li, Yue Luo, Ye Zhu, Xun Zhao, Yu Li, and Ying Shan. Enforcing temporal consistency in video depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1145–1154, 2021. 2, 4, 6

[20] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1925–1934, 2017. 4

[21] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb (r) d sensing: Depth and uncertainty from a video camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10995, 2019. 1

[22] Renjie Liu. Higher accuracy on vision models with EfficientNet-Lite. https://blog.tensorflow.org/2020/03/higher-accuracy-on-vision-models-with-efficientnet-lite.html. Accessed: 2022-07-03. 4

[23] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4796–4803. IEEE, 2018. 4

[24] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 5

[25] Vladimir Nekrasov, Chunhua Shen, and Ian Reid. Lightweight refinenet for real-time semantic segmentation. *arXiv preprint arXiv:1810.03272*, 2018. 4

[26] Tri Nguyen and Myungsik Yoo. Dense-depth-net: a spatial-temporal approach on depth completion task. In *2021 IEEE Region 10 Symposium (TENSYMP)*, pages 1–3. IEEE, 2021. 1, 2

[27] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2, 4

[28] Fengxiang Rong, Dongfang Xie, Wei Zhu, Huiliang Shang, and Liang Song. A survey of multi view stereo. In *2021 International Conference on Networking Systems of AI (INSAI)*, pages 129–135. IEEE, 2021. 2

[29] Dmitry Senushkin, Mikhail Romanov, Ilia Belikov, Nikolay Patakin, and Anton Konushin. Decoder modulation for indoor depth completion. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2181–2188. IEEE, 2021. 1, 3, 4, 5, 6, 7, 8

[30] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012. 3, 5

[31] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 5

[32] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2, 4

[33] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 international conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017. 2, 4

[34] Kaixuan Wang and Shaojie Shen. Mvdepthnet: Real-time multiview depth estimation neural network. In *2018 International conference on 3d vision (3DV)*, pages 248–257. IEEE, 2018. 1, 2

[35] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4471–4480, 2019. 2, 4

[36] Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 175–185, 2018. 2, 3

[37] Yunfan Zhang, Tim Scargill, Ashutosh Vaishnav, Gopika Premsankar, Mario Di Francesco, and Maria Gorlatova. Indepth: Real-time depth inpainting for mobile augmented reality. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(1):1–25, 2022. 1, 3, 5, 6, 7, 8

[38] Chaoqiang Zhao, Qiyu Sun, Chongzhen Zhang, Yang Tang, and Feng Qian. Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences*, 63(9):1612–1627, 2020. 1, 2, 5