This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Recurrent MVSNet for High-resolution Multi-view Stereo Depth Inference

Yao Yao[1*] Zixin Luo[1] Shiwei Li[1] Tianwei Shen[1] Tian Fang[2†] Long Quan[1]

[1]The Hong Kong University of Science and Technology
{yyaoag, zluoag, slibc, tshenaa, quan}@cse.ust.hk
[2]Shenzhen Zhuke Innovation Technology (Altizure)
fangtian@altizure.com

Abstract

Deep learning has recently demonstrated its excellent performance for multi-view stereo (MVS). However, one major limitation of current learned MVS approaches is the scalability: the memory-consuming cost volume regularization makes the learned MVS hard to be applied to high-resolution scenes. In this paper, we introduce a scalable multi-view stereo framework based on the recurrent neural network. Instead of regularizing the entire 3D cost volume in one go, the proposed Recurrent Multi-view Stereo Network (R-MVSNet) sequentially regularizes the 2D cost maps along the depth direction via the gated recurrent unit (GRU). This reduces dramatically the memory consumption and makes high-resolution reconstruction feasible. We first show the state-of-the-art performance achieved by the proposed R-MVSNet on the recent MVS benchmarks. Then, we further demonstrate the scalability of the proposed method on several large-scale scenarios, where previous learned approaches often fail due to the memory constraint. Code is available at `https://github.com/YoYo000/MVSNet`.

1. Introduction

Multi-view stereo (MVS) aims to recover the dense representation of the scene given multi-view images and calibrated cameras. While traditional methods [24, 10, 29, 9] have achieved excellent reconstruction performance, recent works [14, 13, 30] show that learned approaches are able to produce results comparable to the traditional state-of-the-arts. In particular, MVSNet [30] proposed a deep architecture for depth map estimation, which significantly boosts the reconstruction completeness and the overall quality.

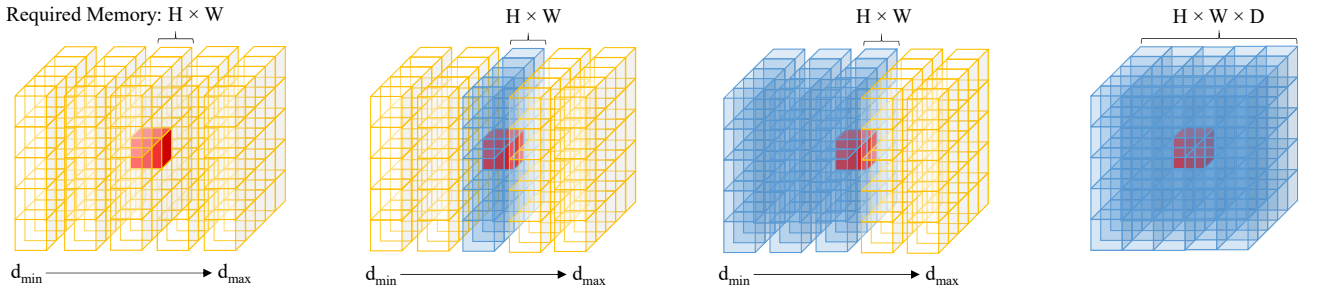
One of the key advantages of learning-based MVS is the cost volume regularization, where most networks apply multi-scale 3D CNNs [14, 15, 30] to regularize the 3D cost volume. However, this step is extremely memory expensive: it operates on 3D volumes and the memory requirement grows cubically with the model resolution (Fig. 1 (d)). Consequently, current learned MVS algorithms could hardly be scaled up to high-resolution scenarios.

Recent works on 3D with deep learning also acknowledge this problem. OctNet [23] and O-CNN [27] exploit the sparsity in 3D data and introduce the octree structure to 3D CNNs. SurfaceNet [14] and DeepMVS [13] apply the engineered divide-and-conquer strategy to the MVS reconstruction. MVSNet [30] builds the cost volume upon the reference camera frustum to decouple the reconstruction into smaller problems of per-view depth map estimation. However, when it comes to a high-resolution 3D reconstruction (e.g., volume size $> 512^3$ voxels), these methods will either fail or take a long time for processing.

To this end, we present a novel scalable multi-view stereo framework, dubbed as R-MVSNet, based on the recurrent neural network. The proposed network is built upon the MVSNet architecture [30], but regularizes the cost volume in a sequential manner using the convolutional gated recurrent unit (GRU) rather than 3D CNNs. With the sequential processing, the online memory requirement of the algorithm is reduced from cubic to quadratic to the model resolution (Fig. 1 (c)). As a result, the R-MVSNet is applicable to high resolution 3D reconstruction with *unlimited* depth-wise resolution.

We first evaluate the R-MVSNet on DTU [1], Tanks and Temples [17] and ETH3D [25] datasets, where our method produces results comparable or even outperforms the state-of-the-art MVSNet [30]. Next, we demonstrate the scalability of the proposed method on several large-scale scenarios with detailed analysis on the memory consumption. R-MVSNet is much more efficient than other methods in GPU memory and is the first learning-based approach applicable to such wide depth range scenes, e.g., the *advance* set of Tanks and Temples dataset [17].

*Intern at Shenzhen Zhuke Innovation Technology (Altizure).
†Corresponding author.



(a) Winner-take-all (b) Spatial Regularization (c) Recurrent Regularization (Proposed) (d) 3D CNNs Regularization

Figure 1: Illustrations of different regularization schemes. For the interest red voxel, we use voxels in blue to denote its receptive field during the cost volume regularization. The runtime memory requirement is also listed on top of the volume, where H, W and D denote the image height, width and depth sample number respectively. The 3D CNNs gather the cost information across the whole space, however, requires a runtime memory cubical to the model resolution

2. Related Work

Learning-based MVS Reconstruction Recent learning-based approaches have shown great potentials for MVS reconstruction. Multi-patch similarity [11] is proposed to replace the traditional cost metric with the learned one. SurfaceNet [14] and DeepMVS [13] pre-warp the multi-view images to 3D space, and regularize the cost volume using CNNs. LSM [15] proposes differentiable projection operations to enable the end-to-end MVS training. Our approach is mostly related to MVSNet [30], which encodes camera geometries in the network as differentiable homography and infers the depth map for the reference image. While some methods have achieved excellent performance in MVS benchmarks, aforementioned learning-based pipelines are restricted to small-scale MVS reconstructions due to the memory constraint.

Scalable MVS Reconstruction The memory requirement of learned cost volume regularizations [14, 15, 13, 5, 30] grows cubically with the model resolution, which will be intractable when large image sizes or wide depth ranges occur. Similar problem also exists in traditional MVS reconstructions (e.g., semi-global matching [12]) if the whole volume is taken as the input to the regularization. To mitigate the scalability issue, learning-based OctNet [23] and O-CNN [27] exploit the sparsity in 3D data and introduce the octree structure to 3D CNNs, but are still restricted to reconstructions with resolution $< 512^3$ voxels. Heuristic divide-and-conquer strategies are applied in both classical [18] and learned MVS approaches [14, 13], however, usually lead to the loss of global context information and the slow processing speed.

On the other hand, scalable traditional MVS algorithms all regularize the cost volume implicitly. They either apply local depth propagation [19, 9, 10, 24] to iteratively refine depth maps/point clouds, or sequentially regularize the cost volume using simple plane sweeping [7] and 2D spatial cost aggregation with depth-wise winner-take-all [28, 31]. In this work, we follow the idea of sequential processing, and propose to regularize the cost volume using the convolutional GRU [6]. GRU is a RNN architecture [8] initially proposed for learning sequential speech and text data, and is recently applied to 3D volume processing, e.g., video sequence analysis [3, 34]. For our task, the convolutional GRU gathers spatial as well as temporal context information in the depth direction, which is able to achieve comparable regularization results to 3D CNNs.

3. Network Architecture

This section describes the detailed network architecture of R-MVSNet. Our method can be viewed as an extension to the recent MVSNet [30] with cost volume regularization using convolutional GRU. We first review the MVSNet architecture in Sec. 3.1, and then introduce the recurrent regularization in Sec. 3.2 and the corresponding loss formulation in Sec. 3.3.

3.1. Review of MVSNet

Given a reference image $\mathbf{I}_1$ and a set of its neighboring source images $\{\mathbf{I}_i\}_{i=2}^N$, MVSNet [30] proposes an end-to-end deep neural network to infer the reference depth map $\mathbf{D}$. In its network, deep image features $\{\mathbf{F}_i\}_{i=1}^N$ are first extracted from input images through a 2D network. These 2D image features will then be warped into the reference camera frustum by differentiable homographies to build the feature volumes $\{\mathbf{V}_i\}_{i=1}^N$ in 3D space. To handle arbitrary N-view image input, a variance based cost metric is proposed to map N feature volumes to one cost volume $\mathbf{C}$. Similar to other stereo and MVS algorithms, MVSNet regularizes the cost volume using the multi-scale 3D CNNs, and regresses the reference depth map $\mathbf{D}$ through the soft argmin [16] operation. A refinement network is applied at

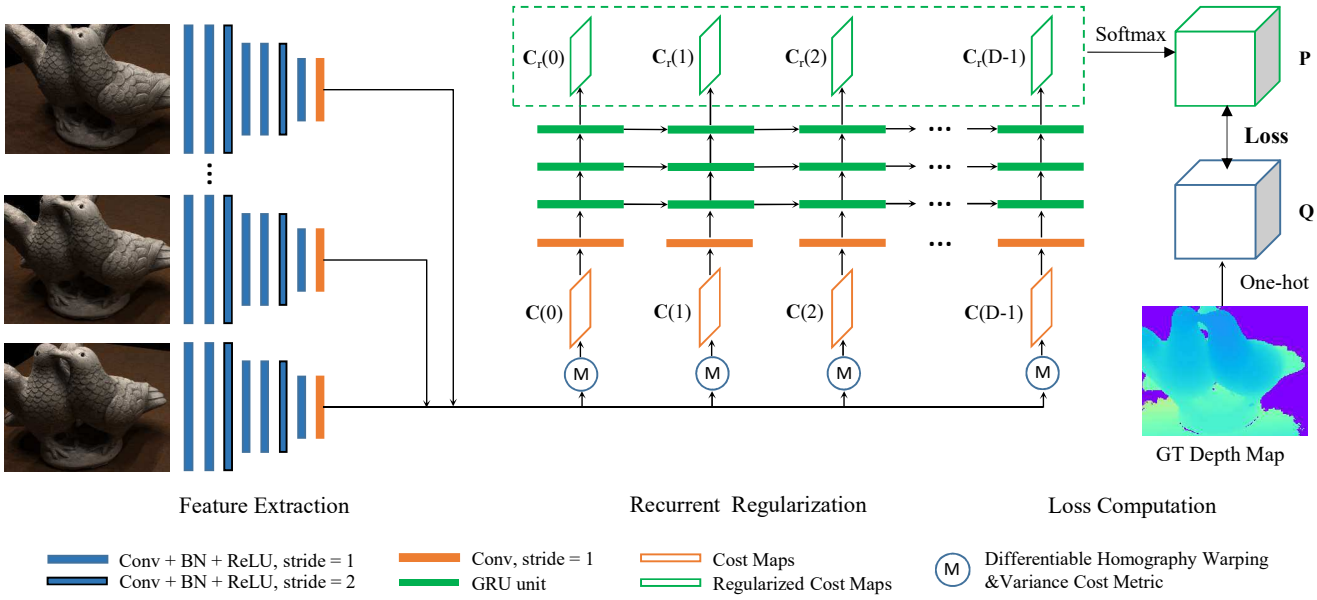


Figure 2: The R-MVSNet architecture. Deep image features are extracted from input images and then warped to the fronto-parallel planes of the reference camera frustum. The cost maps are computed at different depths and are sequentially regularized by the convolutional GRU. The network is trained as a classification problem with the cross-entropy loss

the end of MVSNet to further enhance the depth map quality. As deep image features $\{\mathbf{F}_i\}_{i=1}^{N}$ are downsized during the feature extraction, the output depth map size is $1/4$ to the original image size in each dimension.

MVSNet has shown state-of-the-art performance on DTU dataset [1] and the *intermediate* set of Tanks and Temples dataset [17], which contain scenes with outside-looking-in camera trajectories and small depth ranges. However, MVSNet can only handle a maximum reconstruction scale at $H \times W \times D = 1600 \times 1184 \times 256$ with the 16 GB large memory Tesla P100 GPU, and will fail at larger scenes e.g., the *advanced* set of Tanks and Temples. To resolve the scalability issue especially for the wide depth range reconstructions, we will introduce the novel recurrent cost volume regularization in the next section.

3.2. Recurrent Regularization

Sequential Processing An alternative to globally regularize the cost volume $\mathbf{C}$ in one go is to sequentially process the volume through the depth direction. The simplest sequential approach is the winner-take-all plane sweeping stereo [7], which crudely replaces the pixel-wise depth value with the better one and thus suffers from noise (Fig. 1 (a)). To improve, cost aggregation methods [28, 31] filter the matching cost $\mathbf{C}(d)$ at different depths (Fig. 1 (b)) so as to gather spatial context information for each cost estimation. In this work, we follow the idea of sequential processing, and propose a more powerful recurrent regularization scheme based on convolutional GRU. The proposed method is able to gather spatial as well as the uni-directional context information in the depth direction (Fig. 1 (c)), which achieves regularization results comparable to the full-space 3D CNNs but is much more efficient in runtime memory.

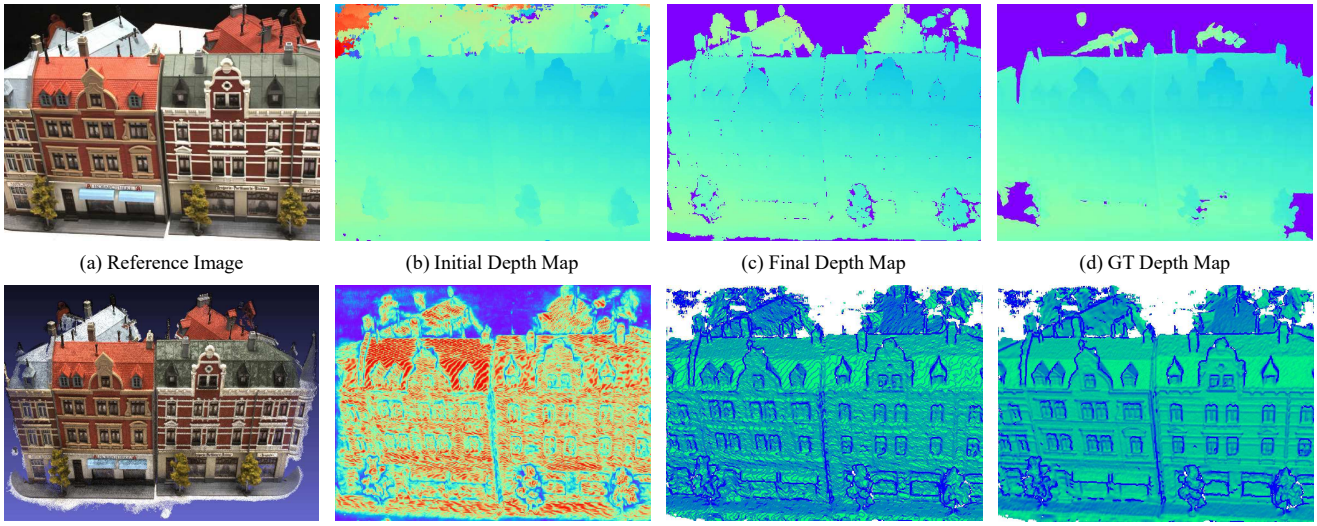
Convolutional GRU Cost volume $\mathbf{C}$ could be viewed as D cost maps $\{\mathbf{C}(i)\}_{i=1}^{D}$ concatenated in the depth direction. If we denote the output of regularized cost maps as $\{\mathbf{C}_r(i)\}_{i=1}^{D}$, for the ideal sequential processing at the t^{th} step, $\mathbf{C}_r(t)$ should be dependent on cost maps of the current step $\mathbf{C}(t)$ as well as all previous steps $\{\mathbf{C}(i)\}_{i=1}^{t-1}$. Specifically, in our network we apply a convolutional variant of GRU to aggregate such temporal context information in depth direction, which corresponds to the time direction in language processing. In the following, we denote '$\odot$' as the element-wise multiplication, '[]' the concatenation and '$*$' the convolution operation. Cost dependencies are formulated as:

$$\mathbf{C}_r(t) = (1 - \mathbf{U}(t)) \odot \mathbf{C}_r(t-1) + \mathbf{U}(t) \odot \mathbf{C}_u(t) \quad (1)$$

where $\mathbf{U}(t)$ is the update gate map to decide whether to update the output for current step, $\mathbf{C}_r(t-1)$ is the regularized cost map of late step, and $\mathbf{C}_u(t)$ could be viewed as the updated cost map in current step, which is defined as:

$$\mathbf{C}_u(t) = \sigma_c(\mathbf{W}_c * [\mathbf{C}(t), \mathbf{R}(t) \odot \mathbf{C}_r(t-1)] + \mathbf{b}_c) \quad (2)$$

$\mathbf{R}(t)$ here is the reset gate map to decide how much the previous $\mathbf{C}_r(t-1)$ should affect the current update. $\sigma_c(\cdot)$ is



(a) Reference Image (b) Initial Depth Map (c) Final Depth Map (d) GT Depth Map

(e) Final Point Cloud (f) Probability Map (g) Depth Gradient (h) Depth Gradient after Refinement

Figure 3: Reconstruction pipeline. (a) Image 24 of DTU [1] *scan 15*. (b) Initial depth map from the network. (c) Final depth map (Sec. 4.3). (d) Ground truth depth map. (e) Point cloud output. (f) Probability estimation for depth map filtering (Sec. 4.3). (g) The gradient visualization of the initial depth map. (h) The gradient visualization after the refinement (Sec. 4.2)

the nonlinear mapping, which is the element-wise sigmoid function. The update gate and reset gate maps are also related to the current input and previous output:

$$\mathbf{R}(t) = \sigma_g(\mathbf{W}_r * [\mathbf{C}(t), \mathbf{C}_r(t-1)] + \mathbf{b}_r) \tag{3}$$

$$\mathbf{U}(t) = \sigma_g(\mathbf{W}_u * [\mathbf{C}(t), \mathbf{C}_r(t-1)] + \mathbf{b}_u) \tag{4}$$

$\mathbf{W}$ and $\mathbf{b}$ are learned parameters. The nonlinear $\sigma_g(\cdot)$ is the hyperbolic tangent to make soft decisions for the updates.

The convolutional GRU architecture not only spatially regularizes the cost maps through 2D convolutions, but also aggregates the temporal context information in depth direction. We will show in the experiment section that our GRU regularization can significantly outperform the simple winner-take-all or only the spatial cost aggregation.

Stacked GRU The basic GRU model is comprised of a single layer. To further enhance the regularization ability, more GRU units could be stacked to make a deeper network. In our experiments, we adopt a 3-layer stacked GRU structure (Fig. 2). Specifically, we first apply a 2D convolutional layer to map the 32-channel cost map $\mathbf{C}(t)$ to 16-channel as the input to the first GRU layer. The output of each GRU layer will be used as the input to the next GRU layer, and the output channel numbers of the 3 layers are set to 16, 4, 1 respectively. The regularized cost maps $\{\mathbf{C}_r(i)\}_{i=1}^{D}$ will finally go through a softmax layer to generate the probability volume $\mathbf{P}$ for calculating the training loss.

3.3. Training Loss

Most deep stereo/MVS networks regress the disparity/depth outputs using the *soft argmin* operation [16], which can be interpreted as the expectation value along the depth direction [30]. The expectation formulation is valid if depth values are *uniformly* sampled within the depth range. However, in recurrent MVSNet, we apply the *inverse depth* to sample the depth values in order to efficiently handle reconstructions with wide depth ranges. Rather than treat the problem as a regression task, we train the network as a multi-class classification problem with cross entropy loss:

$$Loss = \sum_{\mathbf{p}} \Big(\sum_{i=1}^{D} -\mathbf{P}(i, \mathbf{p}) \cdot \log \mathbf{Q}(i, \mathbf{p}) \Big) \tag{5}$$

where $\mathbf{p}$ is the spatial image coordinate and $\mathbf{P}(i, \mathbf{p})$ is a voxel in the probability volume $\mathbf{P}$. $\mathbf{Q}$ is the ground truth binary occupancy volume, which is generated by the one-hot encoding of the ground truth depth map. $\mathbf{Q}(i, \mathbf{p})$ is the corresponding voxel to $\mathbf{P}(i, \mathbf{p})$.

One concern about the classification formulation is the discretized depth map output [32, 20, 13]. To achieve sub-pixel accuracy, a variational depth map refinement algorithm is proposed in Sec. 4.2 to further refine the depth map output. In addition, while we need to compute the whole probability volume during training, for testing, the depth map can be sequentially retrieved from the regularized cost maps using the winner-take-all selection.

4. Reconstruction Pipeline

The proposed network in the previous section generates the depth map per-view. This section describes the non-learning parts of our 3D reconstruction pipeline.

4.1. Preprocessing

To estimate the reference depth map using R-MVSNet, we need to prepare: 1) the source images $\{\mathbf{I}_i\}_{i=2}^N$ of the given reference image $\mathbf{I}_1$, 2) the depth range $[d_{min}, d_{max}]$ of the reference view and 3) the depth sample number D for sampling depth values using the *inverse depth* setting.

For selecting the source images, we follow MVSNet [30] to score each image pair using a piece-wise Gaussian function w.r.t. the baseline angle of the sparse point cloud [33]. The neighboring source images are selected according to the pair scores in descending order. The depth range is also determined by the sparse point cloud with the implementation of COLMAP [24]. Depth samples are chosen within $[d_{min}, d_{max}]$ using the inverse depth setting and we determine the total depth sample number D by adjusting the temporal depth resolution to the spatial image resolution (details are described in the supplementary material).

4.2. Variational Depth Map Refinement

As mentioned in Sec. 3.3, a depth map will be retrieved from the regularized cost maps through the winner-take-all selection. Compare to the *soft argmin* [16] operation, the *argmax* operation of winner-take-all cannot produce depth estimations with sub-pixel accuracy. To alleviate the stair effect (see Fig. 3 (g) and (h)), we propose to refine the depth map in a small depth range by enforcing the multi-view photo-consistency.

Given the reference image $\mathbf{I}_1$, the reference depth map $\mathbf{D}_1$ and one source image $\mathbf{I}_i$, we project $\mathbf{I}_i$ to $\mathbf{I}_1$ through $\mathbf{D}_1$ to form the reprojected image $\mathbf{I}_{i\to1}$. The image reprojection error between $\mathbf{I}_1$ and $\mathbf{I}_{i\to1}$ at pixel $\mathbf{p}$ is defined as:

$$\begin{aligned} E^i(\mathbf{p}) &= E^i_{photo}(\mathbf{p}) + E^i_{smooth}(\mathbf{p}) \\ &= \mathcal{C}(\mathbf{I}_1(\mathbf{p}), \mathbf{I}_{i\to1}(\mathbf{p})) + \sum_{\mathbf{p}'\in\mathcal{N}(\mathbf{p})} \mathcal{S}(\mathbf{p}, \mathbf{p}') \end{aligned} \tag{6}$$

where E^i_{photo} is the photo-metric error between two pixels, E^i_{smooth} is the regularization term to ensure the depth map smoothness. We choose the zero-mean normalized cross-correlation (ZNCC) to measure the photo-consistency $\mathcal{C}(\cdot)$, and use the bilateral squared depth difference $\mathcal{S}(\cdot)$ between $\mathbf{p}$ and its neighbors $\mathbf{p}' \in \mathcal{N}(\mathbf{p})$ for smoothness.

During the refinement, we iteratively minimize the total image reprojection error between the reference image and all source images $E = \sum_i \sum_{\mathbf{p}} E_{i\to1}(\mathbf{p})$ w.r.t. depth map $\mathbf{D}_1$. It is noteworthy that the initial depth map from R-MVSNet has already achieved satisfying result. The proposed variational refinement only fine-tunes the depth values within a small range to achieve sub-pixel depth accuracy, which is similar to the quadratic interpolation in stereo methods [32, 20] and the DenseCRF in DeepMVS [13].

4.3. Filtering and Fusion

Similar to other depth map based MVS approaches[10, 24, 30], we filter and fuse depth maps in R-MVSNet into a single 3D point cloud. The photo-metric and the geometric consistencies are considered in depth map filtering. As described in previous sections, the regularized cost maps will go through a softmax layer to generate the probability volume. In our experiments, we take the corresponding probability of the selected depth value as its confidence measurement (Fig. 3 (f)), and we will filter out pixels with probability lower than a threshold of 0.3. The geometric constraint measures the depth consistency among multiple views, and we follow the geometric criteria in MVSNet [30] that pixels should be at least three view visible. For depth map fusion, we apply the visibility-based depth map fusion [21] as well as the mean average fusion [30] to further enhance the depth map quality and produce the 3D point cloud. Illustrations of our reconstruction pipeline are shown in Fig. 3.

5. Experiments

5.1. Implementation

Training We train R-MVSNet on the DTU dataset [1], which contains over 100 scans taken under 7 different lighting conditions and fixed camera trajectories. While the dataset only provides the ground truth point clouds, we follow MVSNet [30] to generate the rendered depth maps for training. The training image size is set to $W \times H = 640 \times 512$ and the input view number is $N = 3$. The depth hypotheses are sampled from 425mm to 905mm with $D = 192$. In addition, to prevent depth maps from being biased on the GRU regularization order, each training sample is passed to the network with forward GRU regularization from d_{min} to d_{max} as well as the backward regularization from d_{max} to d_{min}. The dataset is split into the same training, validation and evaluation sets as previous works [14, 30]. We choose TensorFlow [2] for the network implementation, and the model is trained for 100k iterations with batch size of 1 on a GTX 1080Ti graphics card. RMSProp is chosen as the optimizer and the learning rate is set to 0.001 with an exponential decay of 0.9 for every 10k iterations.

Testing For testing, we use $N = 5$ images as input, and the inverse depth samples are adaptively selected as described in Sec. 4.1. For Tanks and Temples dataset, the camera parameters are computed from OpenMVG [22] as suggested by MVSNet [30]. Depth map refinement, filtering and fusion are implemented using OpenGL on the same GTX 1080Ti GPU.

5.2. Benchmarks

We first demonstrate the state-of-the-art performance of the proposed R-MVSNet, which produces results comparable to or outperforms the previous MVSNet [30].

DTU Dataset [1] We evaluate the proposed method on the DTU evaluation set. To compare R-MVSNet with MVSNet [30], we set $[d_{min}, d_{max}] = [425, 905]$ and $D = 256$ for all scans. Quantitative results are shown in Table 1. The accuracy and the completeness are calculated using the matlab script provided by the DTU dataset. To summarize the overall reconstruction quality, we calculate the average of the mean accuracy and the mean completeness as the *overall* score. Our R-MVSNet produces the best reconstruction completeness and overall score among all methods. Qualitative results can be found in Fig. 4.

Tanks and Temples Benchmark [17] Unlike the indoor DTU dataset, Tanks and Temples is a large dataset captured in more complex environments. Specifically, the dataset is divided into the *intermediate* and the *advanced* sets. The *intermediate* set contains scenes with outside-look-in camera trajectories, while the *advanced* set contains large scenes with complex geometric layouts, where almost all previous learned algorithms fail due to the memory constraint.

The proposed method ranks 3^{rd} on the *intermediate* set, which outperforms the original MVSNet [30]. Moreover, R-MVSNet successfully reconstructs all scenes and also ranks 3^{rd} on the *advanced* set. The reconstructed point clouds are shown in Fig. 5. It is noteworthy that the benchmarking result of Tanks and Temples is highly dependent on the point cloud density. Our depth map is of size $\frac{H}{4} \times \frac{W}{4}$, which is relatively low-resolution and will result in low reconstruction completeness. So for the evaluation, we linearly upsample the depth map from the network by two ($\frac{H}{2} \times \frac{W}{2}$) before the depth map refinement. The *f_scores* of *intermediate* and *advanced* sets increase from 43.48 to 48.40 and from 24.91 to 29.55 respectively.

ETH3D Benchmark [25] We also evaluate our method on the recent ETH3D benchmark. The dataset is divided into the *low-res* and the *high-res* scenes, and provides the ground truth depth maps for MVS training. We first fine-tune the model on the ETH3D *low-res* training set, however, observe no performance gain compared to the model only pre-trained on DTU. We suspect the problem may be some images in *low-res* training set are blurred and overexposed as they are captured using hand-held devices. Also, the scenes of ETH3D dataset are complicated in object occlusions, which are not explicitly handled in the proposed network. We evaluate on this benchmark without fine-tuning the network. Our method achieves similar performance to MVSNet [30] and ranks 6^{th} on the *low-res* benchmark.

	Mean Acc.	Mean Comp.	Overall (mm)
Camp [4]	0.835	0.554	0.695
Furu [9]	0.613	0.941	0.777
Tola [26]	0.342	1.19	0.766
Gipuma [10]	**0.283**	0.873	0.578
Colmap [10]	0.400	0.664	0.532
SurfaceNet [14]	0.450	1.04	0.745
MVSNet (D=256) [30]	0.396	0.527	0.462
R-MVSNet (D=256)	0.385	0.459	0.422
R-MVSNet (D=512)	0.383	**0.452**	**0.417**

Table 1: Quantitative results on the DTU evaluation scans [1]. R-MVSNet outperforms all methods in terms of reconstruction completeness and overall quality

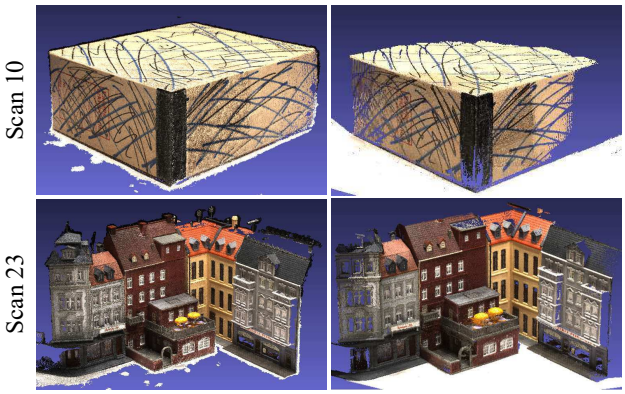


Figure 4: Our results and the ground truth point clouds of scans 10 and 23, DTU [1] dataset

5.3. Scalability

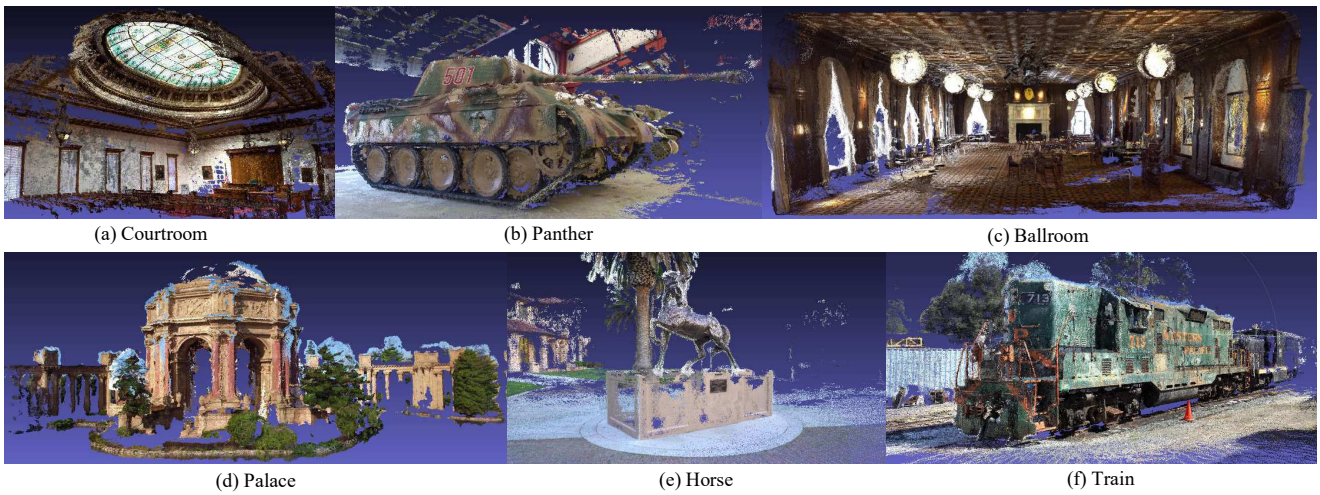
Next, we demonstrate the scalability of R-MVSNet from: 1) wide-range and 2) high-resolution depth reconstructions.

Wide-range Depth Reconstructions The memory requirement of R-MVSNet is independent to the depth sample number D, which enables the network to infer depth maps with large depth range that is unable to be recovered by previous learning-based MVS methods. Some large scale reconstructions of Tanks and Temples dataset are shown in Fig. 5. Table 2 compares MVSNet [30] and R-MVSNet in terms of benchmarking rankings, reconstruction scales and memory requirements. We define the algorithm's memory utility (Mem-Util) as the size of volume processed per memory unit ($\frac{H}{4} \times \frac{W}{4} \times D$ / runtime memory size). R-MVSNet is $\times 8$ more efficient than MVSNet in Mem-Util.

High-resolution Depth Reconstructions R-MVSNet can also produce high-resolution depth reconstructions by sampling denser in depth direction. For the DTU evaluation in Sec. 5.2, if we fix the depth range and change the depth sample number from $D = 256$ to $D = 512$, the *overall* distance score will be reduced from $0.422mm$ to $0.419mm$ (see last row of Table 1).

Dataset	MVSNet[30]						R-MVSNet (Ours)						Mem-Util
	Rank	H	W	Ave. D	Mem.	Mem-Util	Rank	H	W	Ave. D	Mem.	Mem-Util	Ratio
DTU [1]	2	1600	1184	256	15.4 GB	1.97 M	1	1600	1200	512	6.7 GB	9.17 M	4.7
T. Int. [17]	4	1920	1072	256	15.3 GB	2.15 M	3	1920	1080	898	6.7 GB	17.4 M	8.1
T. Adv. [17]	-	-	-	-	-	-	3	1920	1080	698	6.7 GB	13.5 M	-
ETH3D [25]	5	928	480	320	8.7 GB	1.02 M	6	928	480	351	2.1 GB	4.65 M	4.6

Table 2: Comparisons between MVSNet [30] and the proposed R-MVSNet on benchmarking rankings, reconstruction scales and GPU memory requirements on the three MVS datasets [1, 17, 25]. The memory utility (Mem-Util) measures the data size processed per memory unit, and the high ratio between the two algorithms reflects the scalability of R-MVSNet



(a) Courtroom (b) Panther (c) Ballroom

(d) Palace (e) Horse (f) Train

Figure 5: Point cloud reconstructions of Tanks and Temples dataset [17]

5.4. Ablation studies

5.4.1 Networks

This section studies how different components in the network affect the depth map reconstruction. We perform the study on DTU *validation* set with $W \times H \times D = 640 \times 512 \times 256$, and use the average absolute difference between the inferred and the ground truth depth maps for the quantitative comparison. We denote the learned 2D image features as 2D CNNs. The comparison results of following settings are shown Fig. 6 and Fig. 7:

2D CNNs + 3D CNNs Replace the GRU regularization with the same 3D CNNs regularization in MVSNet [30]. As shown in Fig. 6 and Fig. 7, 3D CNNs produces the best depth map reconstructions.

2D CNNs + GRU The setting of the proposed R-MVSNet, which produces the 2^{nd} best depth map results among all settings. The qualitative comparison between 3D CNNs and GRU is shown in Fig. 7 (d) and (e).

2D CNNs + Spatial Replace the GRU regularization with the simple spatial regularization. We approach the spatial regularization by a simple 3-layer, 32-channel 2D network on the cost map. The depth map error of spatial regularization is larger than the GRU regularization.

2D CNNs + Winner-Take-All Replace the GRU regularization with simple the winner-take-all selection. We apply a single layer, 1-channel 2D CNN to directly map the cost map to the regularized cost map. The depth map error is further larger than the spatial regularization.

ZNCC + Winner-Take-All Replace the learned image feature and cost metric with the engineered $ZNCC$ (window size of 7×7). This setting is also referred to the classical plane sweeping [7]. As expected, plane sweeping produces the highest depth map error among all methods.

5.4.2 Post-processing

Next, we study the influences of post processing steps on the final point cloud reconstruction. We reconstruct the DTU *evaluation* without the variational refinement, photo-metric filtering, geometric filtering or depth map fusion. Quantitative results are shown in Table 3.

Without Variational Refinement This setting is similar to the post-processing of MVSNet [30]. The *f_score* is changed to a larger number of 0.465, demonstrating the effectiveness of the proposed depth map refinement.

Without Photo-metric Filtering Table 3 shows that the *f_score* without photo-metric filtering is increased to a larger

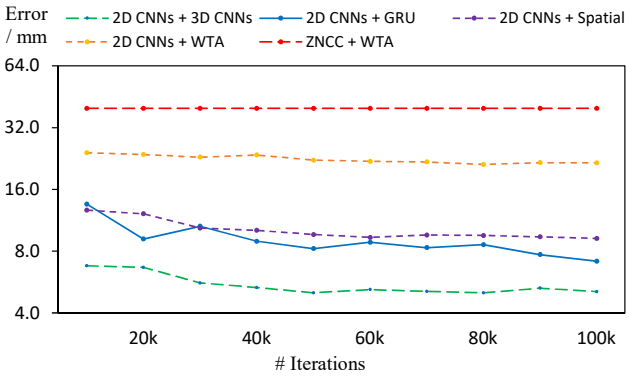
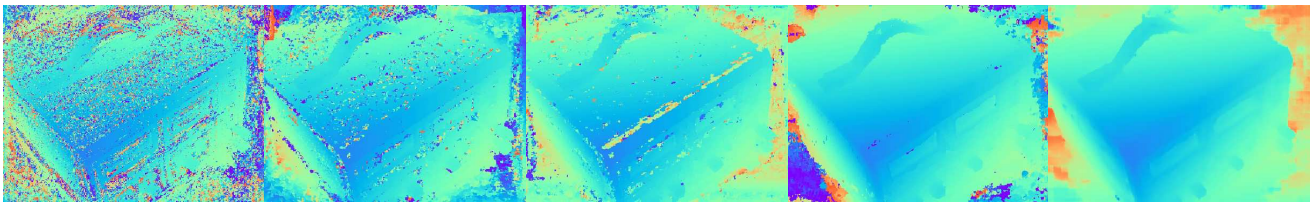


Figure 6: Ablation studies on network architectures, which demonstrate the importance of learned features and learned regularization. WTA is referred to **W**inner-**T**ake-**A**ll and the figure records testing depth map errors during training

Ref.	Pho.	Geo.	Fus.	Acc.	Comp.	Overall
√	√	√	√	**0.385**	0.459	**0.422**
×	√	√	√	0.444	0.486	0.465
√	×	√	√	0.550	**0.384**	0.467
√	√	×	√	0.479	0.385	0.432
√	√	√	×	0.498	0.364	0.431
×	√	√	×	0.605	0.373	0.489
×	×	√	√	0.591	0.411	0.501

Table 3: Ablation studies on different combinations of variational **Ref**inement, **Pho**to-metic filtering, **Geo**metry filtering and depth map **Fus**ion for post-processing. Tested on DTU [1] evaluation set



(a) ZNCC + WTA (b) 2D CNNs + WTA (c) 2D CNNs + spatial (d) 2D CNNs + GRU (e) 2D CNNs + 3D CNNs

Figure 7: Depth map reconstructions of *scan 11*, DTU dataset [1] using different image features and cost volume regularization methods. All models are trained for 100k iterations

number of 0.467, which demonstrates the importance of the probability map for photo-metric filtering (Fig. 3 (f)).

Without Geo-metric Filtering The *f_score* is increased to 0.432, showing the effectiveness of depth consistency.

Without Depth Map Fusion The *f_score* is also increased to 0.431, showing the effectiveness of depth fusion.

5.5. Discussion

Running Time For DTU evaluation with $D = 256$, R-MVSNet generates the depth map at a speed of $9.1s$ / view. Specifically, it takes $2.9s$ to infer the initial depth map and $6.2s$ to perform the depth map refinement. It is noteworthy that the runtime of depth map refinement only relates to refinement iterations and the input image size. Filtering and fusion takes neglectable runtime.

Generalization R-MVSNet is trained with fixed input size of $N \times W \times H \times D = 3 \times 640 \times 512 \times 256$, but it is applicable to arbitrary input size during testing. It is noteworthy that we use the model trained on the DTU dataset [1] for all our experiments without fine-tuning. While R-MVSNet has shown satisfying generalizability to the other two datasets [17, 25], we hope to train R-MVSNet on a more diverse MVS dataset, and expect better performances on Tanks and Temples [17] and ETH3D [25] benchmarks in the future.

Limitation on Image Resolution While R-MVSNet is applicable to reconstructions with unlimited depth-wise resolution, the reconstruction scale is still restricted to the input image size. Currently R-MVSNet can handle a maximum input image size of 3072×2048 on a 11GB GPU, which covers all modern MVS benchmarks except for the ETH3D high-res benchmark (6000×4000).

6. Conclusions

We presented a scalable deep architecture for high-resolution multi-view stereo reconstruction. Instead of using 3D CNNs, the proposed R-MVSNet sequentially regularizes the cost volume through the depth direction with the convolutional GRU, which dramatically reduces the memory requirement for learning-based MVS reconstructions. Experiments show that with the proposed post-processing, R-MVSNet is able to produce high quality benchmarking results as the original MVSNet [30]. Also, R-MVSNet is applicable to large-scale reconstructions which cannot be handled by the previous learning-based MVS approaches.

7. Acknowledgement

This work is supported by Hong Kong RGC GRF 16203518, Hong Kong T22-603/15N, ITC PSKL12EG02. We thank the support of Google Cloud Platform.

References

[1] H. Aanæs, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision (IJCV)*, 2016.

[2] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving deeper into convolutional networks for learning video representations. *International Conference on Learning Representations (ICLR)*, 2016.

[4] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2008.

[5] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. *Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[7] R. T. Collins. A space-sweep approach to true multi-image matching. *Computer Vision and Pattern Recognition (CVPR)*, 1996.

[8] J. L. Elman. Finding structure in time. *Cognitive science*, 1990.

[9] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

[10] S. Galliani, K. Lasinger, and K. Schindler. Massively parallel multiview stereopsis by surface normal diffusion. *International Conference on Computer Vision (ICCV)*, 2015.

[11] W. Hartmann, S. Galliani, M. Havlena, L. Van Gool, and K. Schindler. Learned multi-patch similarity. *International Conference on Computer Vision (ICCV)*, 2017.

[12] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2008.

[13] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang. Deepmvs: Learning multi-view stereopsis. *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[14] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. *International Conference on Computer Vision (ICCV)*, 2017.

[15] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[16] A. Kendall, H. Martirosyan, S. Dasgupta, and P. Henry. End-to-end learning of geometry and context for deep stereo regression. *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[17] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (TOG)*, 2017.

[18] A. Kuhn, H. Hirschmüller, D. Scharstein, and H. Mayer. A tv prior for high-quality scalable multi-view stereo reconstruction. *International Journal of Computer Vision (IJCV)*, 2017.

[19] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2005.

[20] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[21] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nistér, and M. Pollefeys. Real-time visibility-based fusion of depth maps. *International Conference on Computer Vision (ICCV)*, 2007.

[22] P. Moulon, P. Monasse, R. Marlet, and Others. Openmvg. an open multiple view geometry library. `https://github.com/openMVG/openMVG`.

[23] G. Riegler, A. O. Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[24] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2016.

[25] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] E. Tola, C. Strecha, and P. Fua. Efficient large-scale multi-view stereo for ultra high-resolution image sets. *Machine Vision and Applications (MVA)*, 2012.

[27] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 2017.

[28] Q. Yang. A non-local cost aggregation method for stereo matching. *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[29] Y. Yao, S. Li, S. Zhu, H. Deng, T. Fang, and L. Quan. Relative camera refinement for accurate dense reconstruction. *3D Vision (3DV)*, 2017.

[30] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2018.

[31] K.-J. Yoon and I. S. Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2006.

[32] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research (JMLR)*, 2016.

[33] R. Zhang, S. Li, T. Fang, S. Zhu, and L. Quan. Joint camera clustering and surface segmentation for large-scale multi-view stereo. *International Conference on Computer Vision (ICCV)*, 2015.

[34] X. Zhu, J. Dai, X. Zhu, Y. Wei, and L. Yuan. Towards high performance video object detection for mobiles. *arXiv preprint arXiv 1804.05830*, 2018.