DeepMVS: Learning Multi-view Stereopsis

Po-Han Huang[1]   Kevin Matzen[2]   Johannes Kopf[2]   Narendra Ahuja[1]   Jia-Bin Huang[3]

[1]University of Illinois, Urbana-Champaign   [2]Facebook   [3]Virginia Tech

{phuang17,n-ahuja}@illinois.edu   {matzen,jkopf}@fb.com   jbhuang@vt.edu

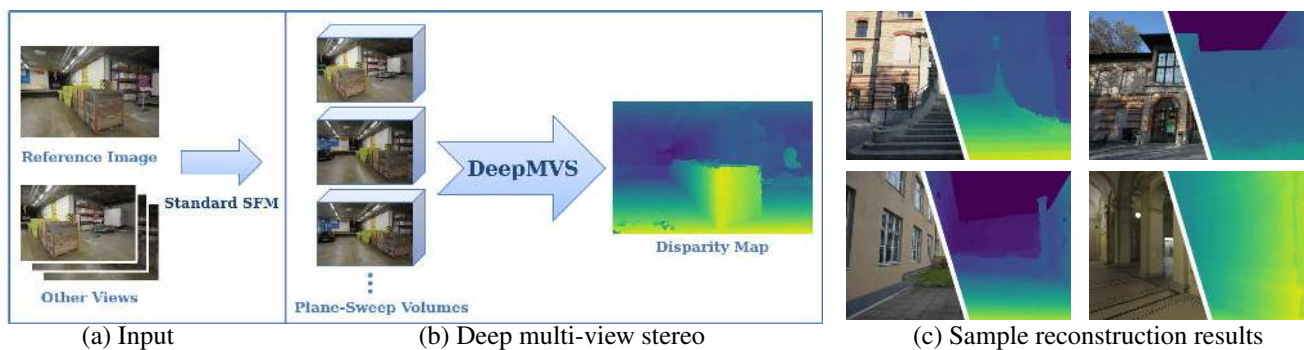(a) Input   (b) Deep multi-view stereo   (c) Sample reconstruction results

Figure 1. (a) Our network takes a set of images with known camera poses and calibration as input; (b) we produce a set of plane-sweep volumes for a reference view and feed these into a convolutional neural network that predicts a disparity map; (c) our network produces high-quality disparity maps even for challenging cases containing poorly textured regions and thin structures.

Abstract

We present DeepMVS, a deep convolutional neural network (ConvNet) for multi-view stereo reconstruction. Taking an arbitrary number of posed images as input, we first produce a set of plane-sweep volumes and use the proposed DeepMVS network to predict high-quality disparity maps. The key contributions that enable these results are (1) supervised pretraining on a photorealistic synthetic dataset, (2) an effective method for aggregating information across a set of unordered images, and (3) integrating multi-layer feature activations from the pre-trained VGG-19 network. We validate the efficacy of DeepMVS using the ETH3D Benchmark. Our results show that DeepMVS compares favorably against state-of-the-art conventional MVS algorithms and other ConvNet based methods, particularly for near-textureless regions and thin structures.

1. Introduction

Multi-view Stereo (MVS) methods aim at reconstructing disparity maps from a collection of images with known camera poses and calibration, possibly estimated using Structure from Motion (SFM) algorithms.[1] MVS is one of the fundamental computer vision problems that have seen decades of research and it is a core component in numerous important applications, including 3D reconstruction, novel view synthesis, augmented reality, and medical imaging [9].

Conventional MVS algorithms often estimate the disparity map by computing plane-sweep volumes and optimizing photometric consistency with handcrafted error functions to measure similarity between patches [9]. Aside from photometric consistency, other 3D cues such as lighting [39, 24], shadows [1], color [11], geometric structures [7], and semantic cues [14] have been incorporated into the MVS pipeline for improving the reconstruction accuracy. However, designing algorithms that make explicit use of all these cues is a non-trivial task. Despite extensive research, the results of state-of-the-art MVS algorithms often still contain numerous artifacts, in particular around poorly textured regions, thin structures, and reflective or transparent surfaces [9].

Deep Convolutional Neural Networks (ConvNets) have shown great success in many visual recognition tasks including image classification [23] and object detection [12], as well as in dense pixel-level prediction tasks such as semantic segmentation [25] and optical flow [4, 16].

---

[1]Throughout this work, we always refer to "disparities" rather than "depths". Disparities are defined as the reciprocal of depths.

For the use of ConvNets in visual reconstruction problems, early work focuses on learning patch similarity for stereo matching [42, 41, 26]. More recent work performs stereo reconstruction using end-to-end learning. However, these methods either impose constraints on relative camera poses [17, 19] or the number of input images [5, 37], or produce a coarse volumetric reconstruction [3, 18].

In this paper, we present *DeepMVS*, a deep ConvNet for multi-view stereo that addresses these limitations. Given a reference image and an arbitrary number of neighbor views of the scene, we first perform a standard SFM reconstruction to recover the camera calibration and pose for each image. We then produce a disparity map for the reference image in three stages, as illustrated in Figure 2. First, we generate a plane-sweep volume for each neighbor image that contains the warped neighbor colors at every disparity, and let our network extract features from each patch pair (reference patch vs. patch in plane-sweep volume). Second, we use an encoder-decoder architecture with skip connections to aggregate the features across large spatial regions. We incorporate the feature activations from a VGG-Net [35] pretrained on ImageNet to guide our decoder for disparity predictions. Third, we fuse the information extracted by each neighbor image with a max-pooling layer and produce the final disparity prediction. In contrast to Recurrent Network-based approaches [3, 18], the use of max-pooling allows us to process an arbitrary number of unordered input images

Training deep ConvNets for disparity reconstruction requires a large number of ground truth disparity maps. A solution is to train the network on the combination of a large-scale synthetic dataset and a smaller real-world dataset [27]. Synthetic datasets provide dense pixel-wise ground truth labels for training, but they do not reflect the complexity of realistic photometric effects, illumination, and natural image noise. On the other hand, real-world datasets are limited in scale and often do not have labels for regions in which it is difficult to obtain ground-truth data, such as sky and reflective surfaces. To address this issue, we introduce the MVS-SYNTH dataset — a set of 120 photorealistic sequences of synthetic urban scenes for learning-based MVS algorithms. We show that the use of a photorealistic synthetic dataset greatly improves the quality of disparity prediction.

We validate the effectiveness of DeepMVS on the recently introduced ETH3D benchmark dataset [34]. Our results show that DeepMVS outperforms DeMoN [37] in the setting of multi-view stereo, and achieves competitive performance with COLMAP [33], the state-of-the-art among conventional MVS algorithms. In particular, we observe that our network is often able to produce correct disparities in poorly textured regions, such as sky, walls, floors, and desk surfaces, where conventional algorithms fail.

In summary, we make the following contributions:

- We propose DeepMVS, a novel learning-based method

for multi-view stereo.

- Unlike existing work [5, 37, 19], DeepMVS can process an *arbitrary* number of input images. The disparity estimation result is invariant to the order in which the inputs are processed.

- Through extensive evaluation, we show that the incorporation of semantic features, training on photorealistic synthetic MVS-SYNTH dataset, and encoder-decoder architecture for aggregating features over large areas all contribute to the improved performance.

2. Related Work

Multi-view stereo reconstruction. Conventional MVS algorithms focus on designing neighbor selection algorithms and photometric error measures. Recent advances include robust neighbor view selection [24], incorporation of visibility consistency [10], and clustering-based techniques for efficient reconstruction [13, 8]. Recently, Schönberger et al. present a MVS system — COLMAP [33] — that jointly estimates depth and surface normal, leverages photometric and geometric priors for pixelwise view selection, and uses geometric consistency for simultaneous refinement. Through a tight integration of multiple techniques, COLMAP performs among the best algorithms in several public multi-view stereo benchmarks. We refer readers to [9] for a comprehensive overview of multi-view stereo reconstruction algorithms.

While impressive results have been shown, conventional MVS algorithms rely heavily on photometric consistency and often have difficulty in handling poorly textured and reflective surfaces where photometric consistency is unreliable. In addition, these algorithms do not consider other visual cues for depth perceptions such as lighting, shadows, and semantics (e.g., a building has a planar structure). Incorporating such information through hand-crafted objective functions is non-trivial. In this work, we aim at implicitly leveraging these cues through learning from data.

Learning-based MVS. A line of work focuses on learning a good similarity measure for patch matching across two views [42] and multiple views [15] using ConvNets. With the learned stereo matching cost, these methods produce disparity maps by a series of post-processing steps. In contrast, DeepMVS produces disparity maps directly from a set of posed images.

Another line of recent work uses ConvNets that take a plane-sweep volume as input and produce disparity maps (or synthesizes novel view) for the reference images. However, these approaches assume a fixed number of input images [19, 5, 37]. Our proposed DeepMVS can take an *arbitrary* number of images to produce high-quality disparity maps. Several recent works approach multi-view recon-

struction with volumetric methods [3, 18]. These methods take a sequence of images captured from different views and generate a 3D shape of the object using a voxel occupancy grid. Nevertheless, the dimension of the voxel grid is quite constrained by the available GPU memory (e.g., coarse grids of $32\times32\times32$ voxels). It is unclear how the volumetric algorithms can be generalized and applied to high-resolution stereo reconstruction in the real-world.

Learning from simulation. Synthetic datasets alleviate the difficulty and the cost of collecting large-scale training datasets from the real world. Examples of synthetic datasets for training and evaluating computer vision algorithms include indoor scene understanding [43], semantic segmentation [31, 30, 29], and depth and flow estimation [27, 29]. We also found that training with a synthetic dataset improves performance in our context. Our newly collected MVS-SYNTH dataset complements the missing ground truth depth measurements in the real-world such as sky and reflective surfaces like windows.

3. Learning Multi-view Stereopsis

The entire pipeline of our algorithm can be broken into four steps. We first preprocess the input image sequence (Section 3.1), and then generate plane-sweep volumes (Section 3.2). Next, our network estimates disparity maps from the plane-sweep volumes (Section 3.3), followed by final refinement to improve the results (Section 3.4).

3.1. Input

The input to our algorithm is a sequence of images and their camera poses and calibration (if necessary, we use the SFM algorithm in COLMAP [32] to estimate them). One of the input images is designated as the reference image, for which we seek to obtain a disparity map.

We start by selecting a subset of neighbor images for the reference to be used in the stereopsis using a similar approach to COLMAP [33]. The images which share the most common features with the reference are chosen to be neighbor images. However, unlike COLMAP, we do not discard the neighbor images which have small triangulation angles with the reference, and we do not estimate per-image weights, since we intend to train the network so it automatically determines whether a plane-sweep volume is reliable or not by comparing it with the reference image.

We also estimate the disparity range of the reference image. Following the approach as COLMAP, we estimate the maximum disparity by projecting all the features in the sparse reconstruction model to the reference view and computing the disparities of the features.

3.2. Plane-sweep Volume Generation

For each neighbor image we compute a plane-sweep volume with respect to the reference image as follows. We assume that the scene geometry is an infinite plane, fronto-parallel to the reference view, and at specific disparities: $\{0, \delta, 2\delta, \ldots, (D-1)\delta\}$. The disparity step, δ, is chosen such that $(D-1)\delta$ equals to the estimated maximum disparity of the reference image. We warp the neighbor image accordingly and store the result as a layer in the volume. If any of the assumed disparity is correct and that portion of the scene is not occluded in the neighbor image, we expect that the warped neighbor image matches the reference image well.

By doing this with all the neighbor images, we obtain a stack of plane-sweep volumes with $N \times D$ images, which we denote as $\mathcal{V} = \{V_{n,d} : 0 \leq n < N, 0 \leq d < D\}$. We normalize the RGB values to the range $[-0.5, 0.5]$ and fill the parts in the plane-sweep volumes that are not visible to the corresponding neighbor image with zeros.

The number of disparity levels, D, is predetermined. Increasing D allows us to use a smaller disparity step δ to reduce the quantization errors in the results, but also increases the number of parameters in the network and thus the GPU memory. As a compromise, we choose disparity level $D = 100$.

3.3. Network Architecture

Figure 2 and Figure 3 illustrate the architecture of Deep-MVS with the hyper-parameters. Our network can be broken into three parts: 1) the patch matching network, 2) the intra-volume feature aggregation network, and 3) the inter-volume feature aggregation network. Except for the very last layer of the network, all the convolutional layers in the network are followed by a Scaled Exponential Linear Unit (SELU) layer [21].

Patch matching. The goal of our patch matching network is to extract a set of per-pixel features that can better aid in the comparison of patches than hand-crafted photometric descriptors could do alone. The patch matching network takes a patch from the reference image I_R and a single patch $V_{n,d}$ from the plane-sweep volume that corresponds to the n-th neighbor image at d-th disparity level as input. The first convolutional layer extracts 64-channel features from the two patches. The features are then concatenated and passed through three more convolutional layers before turning into 4-channel patch matching features. We repeat this process for all $N \times D$ plane-swept images.

Intra-volume feature aggregation. For each neighbor image, we concatenate the 4-channel patch matching features of all D disparity levels to form a $4 \times D$-channel vol-

Figure 2. DeepMVS network architecture.

Figure 3. Network architecture of the intra-volume feature aggregation network.

ume. Following that is a U-Net structure composed of an encoder, a decoder, and skip connections. Each level of the encoder is formed by a stride-2 convolutional layer followed by an ordinary convolutional layer; each level of the decoder is formed by two convolutional layers followed by a bilinear upsampling layer. We show in Figure 3 the detailed structures and hyper-parameters of the proposed intra-volume feature aggregation network.

In addition, we add semantic features at each level of the decoder. We pass the reference image into the VGG-19 [35] network pre-trained on ImageNet, and take the layers *conv1_2*, *conv2_2*, *conv3_2*, *conv4_2*, and *conv5_2* as semantic features. These semantic features are first multiplied by 0.01 and passed through a convolutional layer so as to reduce dimensionality and to improve numerical stability. Finally, these feature maps are concatenated to each level of the decoder as shown in Figure 3.

This part of the network is intended to pass the features to larger spatial regions and enable the network to make predictions with non-local information. It also aids the disparity predictions using the VGG feature inputs. The output of the intra-volume feature aggregation network is an 800-channel volume F_n containing the disparity prediction information gathered from the n-th neighbor image.

Inter-volume feature aggregation.   In this step, we take the N volumes, $\{F_0, \ldots, F_{N-1}\}$, generated from each of the neighbor images and aggregate them using element-wise max-pooling. The use of max-pooling enables the network to gather information from an arbitrary number of neighbor images, and also ensures that the results are invariant with respect to the order of the neighbor images. This technique was previously used in PointNet [28] and in the work by Hartmann *et al.* in [15] to allow inputs with varying sizes. Finally, we use two convolutional layers converting the aggregated volume into the pixel-wise disparity predictions.

During training, we randomly select the number of neighbor images N from $\{1, 2, 3, 4\}$. By varying N, the network learns to make use of the max-pooling to collect only the useful information from each neighbor image. Even though N is restricted to be no larger than 4 during training (due to the limited size of the GPU memory), we show that our trained network can be applied to an arbitrary number of neighbor images in Section 4.4.

Training loss.   We pose disparity prediction as a multi-class classification problem, and use the cross-entropy loss to train the network. The predicted disparity map can be made by taking the disparity level at which the predicted probability is the highest for each pixel. Namely, for the
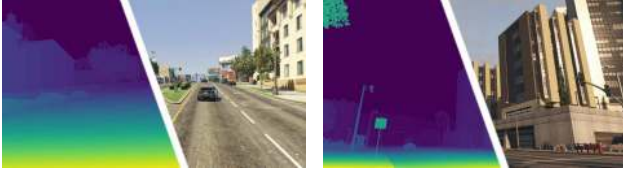
Figure 4. Samples from the proposed MVS-SYNTH dataset, which provides photorealistic images with ground truth disparities even for the sky, reflective surfaces, and thin structures.

output distribution $\mathbf{y} = (y_0, \ldots, y_{D-1})$ of each pixel, the predicted disparity can be chosen by

$$\hat{d}_{\text{raw}} = \operatorname*{argmax}_{d} y_d.$$

We refer to this $\hat{d}_{\text{raw}}$ as the *raw predictions*.

3.4. Refinement

To further improve the quality of the results, we apply the Fully-Connected Conditional Random Field (Dense-CRF) [22] to our raw disparity predictions. The use of DenseCRF encourages the pixels which are spatially close and with similar colors to have closer disparity predictions.

4. Experimental Results

4.1. Datasets

DeMoN datasets. We train our network with the same datasets as used in DeMoN [37]. The dataset consists of short sequences ranging from two to tens of images including real-world datasets (SUN3D [40], RGB-D SLAM [36], CITYWALL and ACHTECK-TURM [6]) of outdoor and indoor scenes and a synthesized dataset (SCENES11 [37, 2]) with random objects flying in the air. As suggested in [37], mixing real-world and synthetic datasets is important since each has its own limitations. The ground truth for real-world datasets contains measurement errors, whereas synthesized datasets have unrealistic appearance, and may not be capable of reflecting some characteristics of real imagery, such as illumination, depth of field, and noise. The image resolution of this dataset is 640×480 pixels.

MVS-Synth dataset. To address the limitations of the DeMoN datasets, we introduce the MVS-SYNTH dataset, which consists of 120 sequences of urban scenes captured in the video game *Grand Theft Auto V*.[2] Each sequence is composed of 100 RGB frames of size 1920×1080, ground

---

[2]This academic article may contain images and/or data from sources that are not affiliated with the article submitter. Inclusion should not be construed as approval, endorsement or sponsorship of the submitter, article or its content by any such party.

truth disparity maps, and the extrinsic and intrinsic camera parameters. Figure 4 shows examples from the MVS-SYNTH dataset.

Compared to existing synthetic datasets, the MVS-SYNTH dataset is more realistic in terms of context and shading. Compared to real-world datasets, MVS-SYNTH provides complete ground truth disparities which cover regions such as the sky, reflective surfaces, and thin structures, whose ground truths are usually missing in real-world datasets. Therefore, training with MVS-SYNTH allows us to predict disparities for these challenging regions. We train the network using both image resolution 1280×720 and 960×540 pixels as data augmentation.

ETH3D datasets. For evaluation, we use the high-res multi-view dataset in the recently introduced ETH3D benchmark datasets [34]. It consists of 13 sequences of real-world outdoor and indoor scenes with ground truth point clouds captured by laser scanners. We project the point clouds back to each view to obtain a ground truth disparity map for each reference image. Note that ground truth data are not complete and contain holes in the sky, reflective surfaces, and thin objects. Nevertheless, we use it to validate the efficacy of our method for real-world scenes. We resize the images to 810×540 pixels for evaluation.

4.2. Implementation Details

Our training process consists of two stages. First, we train the network by replacing the intra-volume feature aggregation network with two simple 3×3 convolutional layers. Here, our goal in the first stage is to pre-train the network so it can be transferred to the second stage. Then, we add the intra-volume feature aggregation network back with weights initialized from the pre-trained network, and train the entire network using both DeMoN and the MVS-SYNTH datasets.

For both training stages, we use the Adam solver [20] with learning rates 10^{-5} and 10^{-6}, respectively, for 320k iterations per stage. We apply gradient clipping to prevent gradient explosion by constraining the L2-norm of the gradients at each layer to be no more than 1.0 at the first stage and 0.1 at the second stage. We implement the network in PyTorch. Training the network with an NVIDIA P100 GPU with 16GB memory takes two days for each stage.

We use 64px$\times$64px patches as our inputs so as to fit our network into the GPU memory at the training stages. We generate the semantic features by a feed-forward pass of a VGG-19 network using the entire image. We then take only the region of interest corresponding to the input patches from the intermediate features. At test time, we feed 128px$\times$128px patches into the network, and take only the center 64px$\times$64px of the output to reduce boundary artifacts. The 64px$\times$64px output patches are then tiled to

achieve full-resolution results.

4.3. Evaluation Metrics

Geometric errors. We compute geometric error by taking the L1 distance between the predicted disparity and the ground truth. Unavailable pixels are ignored.

Photometric errors. We also measure photometric *rephotography error* [38] — the L1 distance between the reference and the rephotography image. We generate the rephotography using the predicted disparity map, warping the pixels to all other neighbor images, sampling colors from the neighbor images, and finally selecting the median among all color candidates for each pixel.

Completeness. Another important factor for evaluation is completeness. We measure completeness using the percentage of pixels whose errors are below a certain threshold. Plotting the relationship between different error thresholds and their corresponding completeness helps visualize the distributions of the errors. The curves lying in the lower right represent more pixels having lower errors and thus have better performance.

4.4. Evaluation

COLMAP. Several conventional MVS algorithms have been proposed, including PMVS [10], MVE [6], and COLMAP [33]. We choose to compare with COLMAP as it is the top performer on the ETH3D dataset [34].

We follow the default settings of COLMAP unless otherwise mentioned. COLMAP provides an option to filter out the predictions that are not geometrically consistent. However, the filtered disparity maps may significantly reduce completeness. We show both unfiltered and filtered maps for comparison.

Note that we do not use DenseCRF to refine COLMAP's noisy unfiltered maps since COLMAP predicts a *deterministic* disparity for each pixel, whereas DenseCRF requires pixel-wise distributions as inputs.

DeMoN. We compare our approach with DeMoN [37] because it is the closest to ours among the existing learning-based stereopsis methods. However, as their network only works with image *pairs*, we propose two ways to extend their approach to multi-view stereo applications.

The first method is to choose the best result among all the disparity maps generated from the image pairs formed by the reference image and its neighbor images. This method is not practical in real applications since the ground truths are not available. Nevertheless, the method establishes the upper-bound performance of DeMoN. The second method is to compute the per-pixel median among all the generated disparity maps so as to aggregate information from all available image pairs.

Table 1. Quantitative comparisons between different algorithms on ETH3D dataset.

Algorithm	Completeness	Geo. error	Pho. error
DeMoN (best)	100%	0.045	0.288
DeMoN (median)	100%	0.201	0.367
COLMAP (filtered)	71%	0.007	0.178
COLMAP (unfiltered)	100%	0.046	**0.218**
Ours	100%	**0.036**	0.224

Since DeMoN is trained with images taken with fixed focal lengths and image resolutions, we crop and resize the images from ETH3D dataset before using them to evaluate DeMoN's performance. This leads to the incomplete reconstruction results in Figure 5 and Figure 6. The cropped regions are ignored when the error is computed. In addition, DeMoN assumes that the translation between the input image pair is a unit vector. Therefore, we multiply the depth maps produced by DeMoN by the actual translational distance between the two views before comparing them with the ground truths.

Qualitative comparisons. Figure 5 shows qualitative comparisons between DeMoN, COLMAP, and our approach. While DeMoN detects the overall structure of the scene, it fails to predict accurate scaling factors and thus results in inaccurate predictions. On the other hand, COLMAP and our approach give accurate predictions wherever the depth cues are sufficient. However, for textureless regions like the sky, the wall, and the surface of the white desks, the predictions made by COLMAP are very noisy, whereas our network is capable of assigning zero disparity to the sky, and interpolating or extrapolating disparities for poorly textured regions.

Figure 6 shows several rephotography results. The results from DeMoN are often blurry and distorted, indicating that the predictions are not accurate. COLMAP performs well in rephotography in the regions where the predictions are clean. However, for challenging regions, the results contain large holes. Our rephotography results generally recover the reference images with only small holes. However, edges appear to be jagged because of the disparity quantization in our approach.

Quantitative comparisons. Table 1 shows quantitative comparisons of the average errors over the entire ETH3D dataset between DeMoN, COLMAP, and our approach. First, DeMoN gives much larger errors than COLMAP and our approach with respect to both metrics. COLMAP's filtered predictions have significantly lower average errors, but it discards 29% of the pixels to achieve that. Finally, COLMAP's unfiltered maps and our results have similar errors. While COLMAP gives slightly lower photometric errors, our approach gives slightly lower geometric errors.
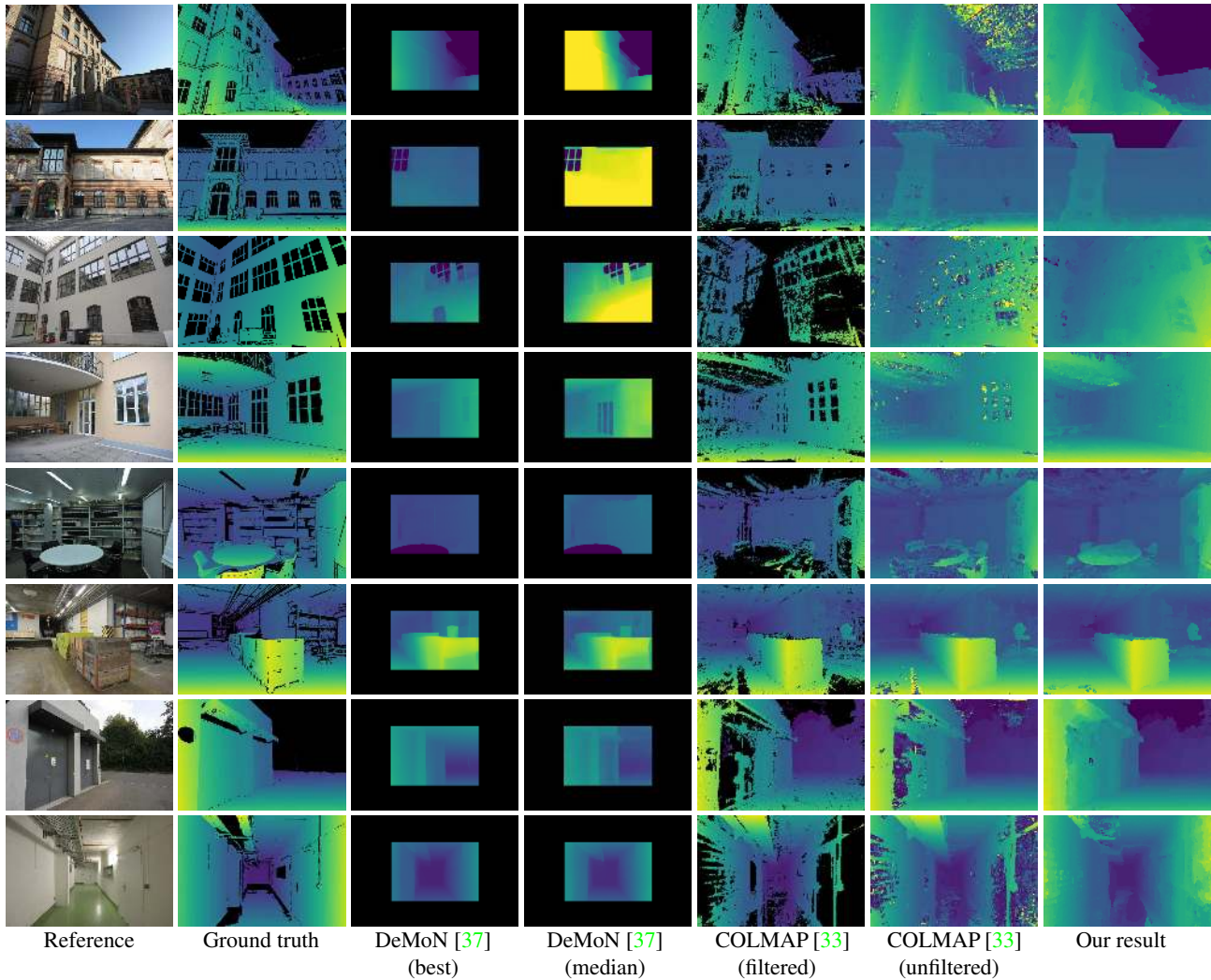
Figure 5. Qualitative comparisons between different algorithms on ETH3D dataset.

| | Reference | Ground truth | DeMoN [37] (best) | DeMoN [37] (median) | COLMAP [33] (filtered) | COLMAP [33] (unfiltered) | Our result |

Figure 7 shows the distributions of the errors. We observe that COLMAP predicts 85% of the pixels with smaller geometric errors than our approach, whereas our approach gives more accurate results for the other 15% pixels. A possible reason is that for regions with sufficient depths cues, COLMAP produces accurate predictions. Our approach, on the other hand, suffers from the quantized disparity effects. However, for the challenging regions, COLMAP gives noisy predictions which lead to large errors, whereas our approach produces plausible predictions. As for the distributions of the photometric errors, our approach produces almost the same curve as COLMAP does.

Progressive improvement. Figure 8 shows two examples of the progressive improvements by COLMAP and our approach for an increasing number of input images. When N is small, COLMAP tends to produce large geometric errors, whereas our network can still generate accurate predictions

Table 2. Contributions of different components in our algorithm.

Components	Geo. error	Pho. error
Pretraining	0.051	0.242
+ U-net	0.043	0.230
+ U-net + VGG	0.040	0.226
+ U-net + VGG + DenseCRF	**0.036**	**0.224**
+ U-net + VGG + DenseCRF − MVS-SYNTH	0.037	0.225

and hallucinate disparities for the regions lacking of good depth cues.

4.5. Ablation Studies

DenseCRF. As shown in Figure 9, applying DenseCRF removes a large portion of the noisy patches in low-confidence regions such as the reflective wall, and encourages the disparity predictions to follow the color edges. As shown in Table 2, DenseCRF improves the results with respect to both error metrics.
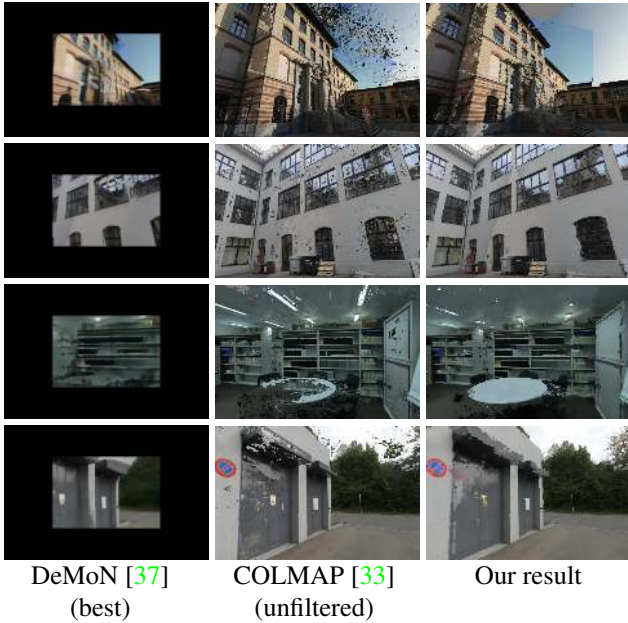
DeMoN [37]   COLMAP [33]   Our result
(best)   (unfiltered)

Figure 6. Comparisons of rephotography results. See Figure 5 for the ground truth reference images.

Geometric errors   Photometric errors
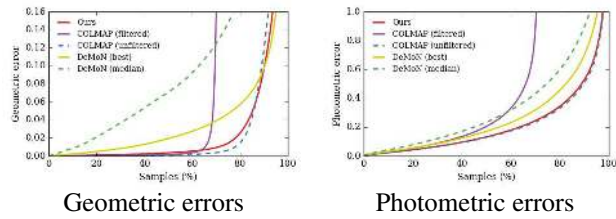
Figure 7. The distributions of the errors of different approaches on ETH3D dataset.

Image /   Our result   COLMAP [33]   Geometric Errors
ground truth   (unfiltered)
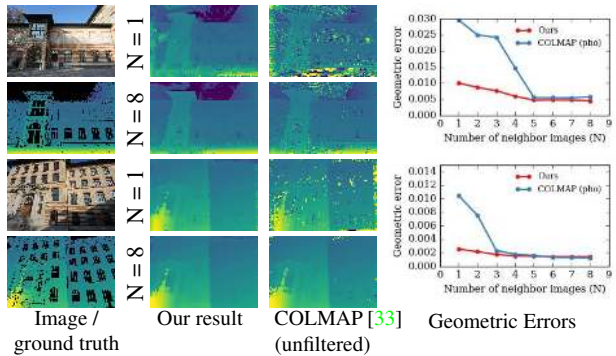
Figure 8. Examples of progressive improvements for increasing number of input images.

MVS-Synth dataset. Table 2 shows that removing MVS-SYNTH dataset from the training set results in slightly larger errors for both metrics. Qualitatively, we observe that the network trained without MVS-SYNTH dataset works very poorly for the sky and reflective surfaces, as Figure 10 shows. These regions usually lack ground truth data, so

Image   Ours   Ours w/o
                      DenseCRF
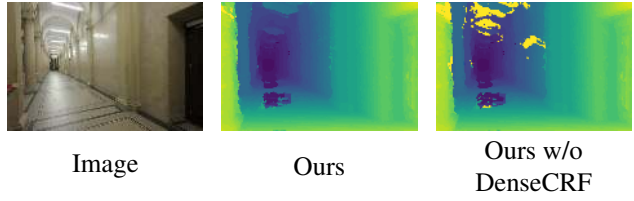
Figure 9. Example of the improvements from the DenseCRF refinement. Applying DenseCRF removes the noisy predictions.

Image   Ours   Ours w/o
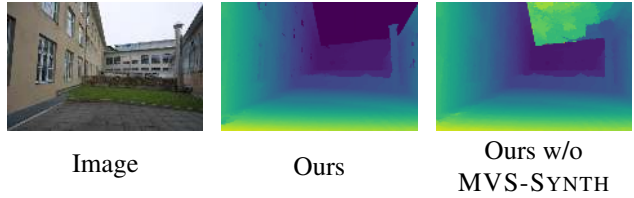   MVS-SYNTH

Figure 10. Comparisons between networks trained with and without the MVS-SYNTH dataset. Without MVS-SYNTH dataset, the network has difficulty in handling regions such as the sky because real-world datasets do not cover these regions.

the errors do not reflect much on the quantitative errors. We suggest that the poor predictions result from the fact that the ground truths in DeMoN dataset does not cover such regions.

U-Net and VGG features. As Table 2 shows, adding the U-net and VGG features each provides improvements in both error metrics. This shows that allowing non-local information and providing semantic features both help the network in better disparity predictions.

4.6. Limitations

Following are some limitations of our network. First, the quantization of disparity results in undesired geometric and photometric errors. Second, our network often fails to predict correct disparities for vegetation areas containing trees or grass. Finally, the computation speed of our algorithm is constrained by the time-consuming generation of the plane-sweep volumes and the deep and large network structures.

5. Conclusions

With DeepMVS, we demonstrate the feasibility of learning Mulit-View Stereopsis with a convolutinoal neural network, and show that learning-based approaches can overcome the weaknesses of conventional algorithms.

Acknowledgement

References

[1] M. Chandraker, S. Agarwal, and D. Kriegman. Shadowcuts: Photometric stereo with shadows. In *CVPR*, 2007. 1

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016. 2, 3

[4] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, Dec 2015. 1

[5] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016. 2

[6] S. Fuhrmann, F. Langguth, and M. Goesele. Mve-a multiview reconstruction environment. In *Eurographics Workshop on Graphics and Cultural Heritage*, 2014. 5, 6

[7] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, 2009. 1

[8] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010. 2

[9] Y. Furukawa, C. Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 1, 2

[10] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010. 2, 6

[11] D. Gallup, J. M. Frahm, and M. Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *CVPR*, 2010. 1

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[13] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *ICCV*, 2007. 2

[14] C. Häne, C. Zach, A. Cohen, and M. Pollefeys. Dense semantic 3d reconstruction. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1730–1743, 2017. 1

[15] W. Hartmann, S. Galliani, M. Havlena, K. Schindler, and L. Van Gool. Learned multi-patch similarity. In *ICCV*, 2017. 2, 4

[16] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 1

[17] N. K. Kalantari, T.-C. Wang, and R. Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics*, 35(6), 2016. 2

[18] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. In *NIPS*, 2017. 2, 3

[19] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *ICCV*, 2017. 2

[20] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[21] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-normalizing neural networks. *arXiv preprint arXiv:1706.02515*, 2017. 3

[22] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *NIPS*, pages 109–117. 2011. 5

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[24] F. Langguth, K. Sunkavalli, S. Hadap, and M. Goesele. Shading-aware multi-view stereo. In *ECCV*, 2016. 1, 2

[25] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[26] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016. 2

[27] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2, 3

[28] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2016. 4

[29] S. R. Richter, Z. Hayder, and V. Koltun. Playing for benchmarks. In *ICCV*, 2017. 3

[30] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 3

[31] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 3

[32] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 3

[33] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 2, 3, 6, 7, 8

[34] 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. In *CVPR*, 2017. 2, 5, 6

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2, 4

[36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 5

[37] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 2, 5, 6, 7, 8

[38] M. Waechter, M. Beljan, S. Fuhrmann, N. Moehrle, J. Kopf, and M. Goesele. Virtual rephotography: Novel view prediction error for 3d reconstruction. *ACM Transactions on Graphics (TOG)*, 36(1):8, 2017. 6

[39] C. Wu, B. Wilburn, Y. Matsushita, and C. Theobalt. High-quality shape from multi-view stereo and shading under general illumination. In *CVPR 2011*, 2011. 1

[40] J. Xiao, A. Owens, and A. Torralba. SUN3D: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, 2013. 5

[41] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 2

[42] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016. 2

[43] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. In *CVPR*, 2017. 3