Visibility-Aware Point-Based Multi-View Stereo Network

Rui Chen, Songfang Han, Jing Xu, and Hao Su

Abstract—We introduce VA-Point-MVSNet, a novel visibility-aware point-based deep framework for multi-view stereo (MVS). Distinct from existing cost volume approaches, our method directly processes the target scene as point clouds. More specifically, our method predicts the depth in a coarse-to-fine manner. We first generate a coarse depth map, convert it into a point cloud and refine the point cloud iteratively by estimating the residual between the depth of the current iteration and that of the ground truth. Our network leverages 3D geometry priors and 2D texture information jointly and effectively by fusing them into a feature-augmented point cloud, and processes the point cloud to estimate the 3D flow for each point. This point-based architecture allows higher accuracy, more computational efficiency and more flexibility than cost-volume-based counterparts. Furthermore, our visibility-aware multi-view feature aggregation allows the network to aggregate multi-view appearance cues while taking into account visibility. Experimental results show that our approach achieves a significant improvement in reconstruction quality compared with state-of-the-art methods on the DTU and the Tanks and Temples dataset. The code of VA-Point-MVSNet proposed in this work will be released at https://github.com/callmeray/PointMVSNet.

Index Terms—Multi-view stereo, 3D deep learning

1 Introduction

Multi-view stereo (MVS) aims to reconstruct the dense geometry of a 3D object from a sequence of images and corresponding camera poses and intrinsic parameters. MVS has been widely used in various applications, including autonomous driving, robot navigation, and remote sensing [1], [2]. Recent learning-based MVS methods [3], [4], [5] have shown great success compared with their traditional counterparts as learning-based approaches are able to learn to take advantage of scene global semantic information, including object materials, specularity, 3D geometry priors, and environmental illumination, to get more robust matching and more complete reconstruction. Most of these approaches apply dense multi-scale 3D CNNs to predict the depth map or voxel occupancy. However, 3D CNNs require memory cubic to the model resolution, which can be potentially prohibitive to achieving optimal performance. While Tatarchenko *et al.* [6] addressed this problem by progressively generating an Octree structure, the quantization artifacts brought by grid partitioning still remain, and errors may accumulate since the tree is generated layer by layer. Moreover, MVS fundamentally relies on finding photo-consistency across the input images. However, image appearance cues from invisible views, which includes being occluded and out of FOV (Field of View), are not consistent with those from visible views, which is misguiding for accurate depth prediction and therefore needs robust handling.

In this work, we propose a novel Visibility-Aware Point-based Multi-View Stereo Network (VA-Point-MVSNet), where the target scene is directly processed as a point cloud, a more efficient representation, particularly when the 3D resolution is high. Our framework is composed of two steps: first, in order to carve out the approximate object surface from the whole scene, an initial coarse depth map is generated by a relatively small 3D cost volume and then converted to a point cloud. Subsequently, our novel *PointFlow* module is applied to iteratively regress accurate and dense point clouds from the initial point cloud. Similar to ResNet [7], we explicitly formulate the *PointFlow* to predict the residual between the depth of the current iteration and that of the ground truth. The 3D flow is estimated based on geometry priors inferred from the predicted point cloud and the 2D image appearance cues dynamically fetched from multi-view input images (Fig. 1). Moreover, in order to take into account visibility, including occlusion and out of FOV, for accurate MVS reconstruction, we propose a number of network structure alternatives that infer the visibility of each view for the multi-view feature aggregation.

We find that our VA-Point-MVSNet framework enjoys advantages in accuracy, efficiency, and flexibility when it is compared with previous MVS methods that are built upon a predefined 3D cost volume with a fixed resolution to aggregate information from views. Our method adaptively samples potential surface points in the 3D space. It keeps the continuity of the surface structure naturally, which is necessary for high precision reconstruction. Furthermore, because our network only processes valid information near

- *R. Chen and J. Xu are with the State Key Laboratory of Tribology, the Beijing Key Laboratory of Precision/Ultra-Precision Manufacturing Equipment Control, Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China. E-mail: callmeray@163.com, jingxu@tsinghua.edu.cn.*
- *S. Han is with The Hong Kong University of Science and Technology, Hong Kong. E-mail: hansongfang@gmail.com.*
- *H. Su is with the Department of Computer Science and Engineering, University of California, San Diego, San Diego, CA 92093 USA. E-mail: haosu@eng.ucsd.edu.*

Manuscript received 13 Oct. 2019; revised 2 Apr. 2020; accepted 15 Apr. 2020. Date of publication 22 Apr. 2020; date of current version 2 Sept. 2021. (Corresponding author: Jing Xu.) Recommended for acceptance by T. Hassner. Digital Object Identifier no. 10.1109/TPAMI.2020.2988729

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

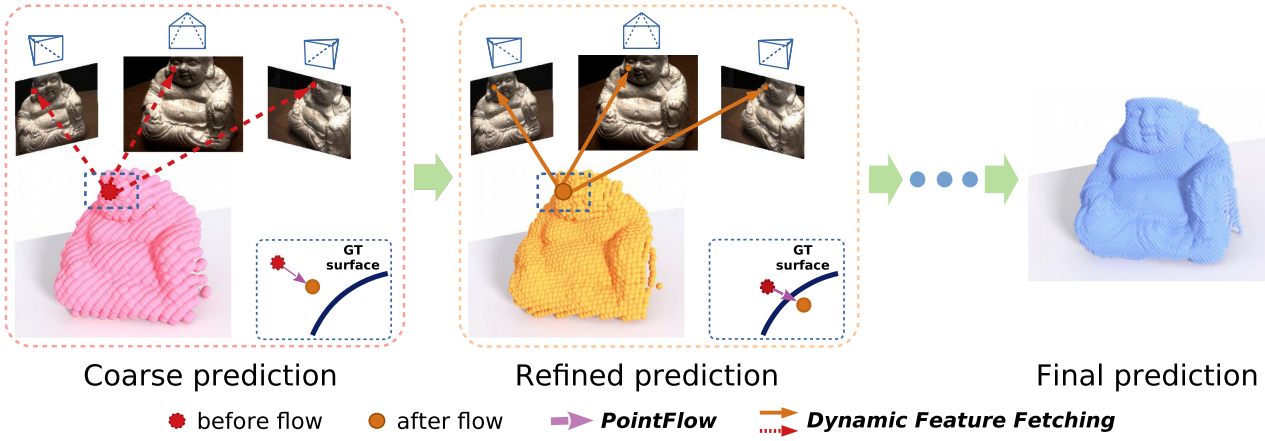


Fig. 1. VA-Point-MVSNet performs multi-view stereo reconstruction in a coarse-to-fine fashion, learning to predict the 3D flow of each point to the ground truth surface based on geometry priors and 2D image appearance cues dynamically fetched from multi-view images and regress accurate and dense point clouds iteratively.

the object surface instead of the whole 3D space as is the case in 3D CNNs, the computation is much more efficient. The adaptive refinement scheme allows us to first peek at the scene at a coarse resolution and then densify the reconstructed point cloud only in the region of interest (ROI). For scenarios such as interaction-oriented robot vision, this flexibility would result in saving of computational power. Lastly, the visibility-aware multi-view feature aggregation allows the network to aggregate multi-view appearance cues while taking into account visibility, which excludes misguiding image information from invisible views and leads to improved reconstruction quality.

Our method achieves state-of-the-art performance on standard multi-view stereo benchmarks among learning-based methods, including DTU [8] and Tanks and Temples [9]. Compared with previous state-of-the-arts, our method produces better results in terms of both completeness and overall quality.

This article is an extension of our previous ICCV work [10]. There are two main additional contributions in this work:

1) We design the novel visibility-aware multi-view feature aggregation module, which takes into account visibility when aggregating multi-view image features and thus improves the reconstruction quality.
2) We create a synthetic MVS dataset using path tracing renderer to generate accurate visibility masks, which are not available from incomplete ground truth depth maps captured by 3D sensors. We present an extensive and comprehensive evaluation of our work on both synthetic dataset and real dataset, and analyze the effectiveness of each component, in particular our novel visibility-aware multi-view feature aggregation module, in our network through comparison and ablation study.

2 Related Work

Multi-View Stereo Reconstruction. MVS is a classical problem that had been extensively studied before the rise of deep learning. A number of 3D representations are adopted, including volumes [11], [12], [13], deformation models [14], [15], [16], and patches [17], [18], [19], which are iteratively updated through multi-view photo-consistency and regularization optimization. Our iterative refinement procedure shares a similar idea with these classical solutions by updating the depth map iteratively. However, our learning-based algorithm achieves improved robustness to input image corruption and avoids the tedious manual hyper-parameters tuning.

Occlusion-Robust MVS. Since MVS counts on finding correspondences across input images, image appearance from occluded views will cause mismatches and reduce the reconstruction accuracy. Vogiatzis *et al.* [20] addressed this problem by designing a new metric of multi-view voting which considers only points of local maximum and eliminates the influence of occluded views on correspondence matching. Further, Liu *et al.* [21] improved the metric by using Gaussian filtering to counteract the effect of noise. COLMAP [22] and some following works [23], [24] handled this problem by dataset-wide pixel-wise view selection using patch color distribution. Our network learns to predict the pixel-wise visibility for all the given source views and use the prediction in multi-view feature aggregation, which can be trained end-to-end and improve the robustness to occlusions.

Learning-Based MVS. Inspired by the recent success of deep learning in image recognition tasks, researchers began to apply learning techniques to stereo reconstruction tasks for better patch representation and matching [25], [26], [27]. Although these methods in which only 2D networks are used have made a great improvement on stereo tasks, it is difficult to extend them to multi-view stereo tasks, and their performance is limited in challenging scenes due to the lack of contextual geometry knowledge. Concurrently, 3D cost volume regularization approaches have been proposed [3], [28], [29], where a 3D cost volume is built either in the camera frustum or the scene. Next, the multi-view 2D image features are warped in the cost volume, so that 3D CNNs can be applied to it. The key advantage of 3D cost volume is that the 3D geometry of the scene can be captured by the network explicitly, and the photo-metric matching can be performed in 3D space, alleviating the influence of image distortion caused by perspective transformation, which makes these

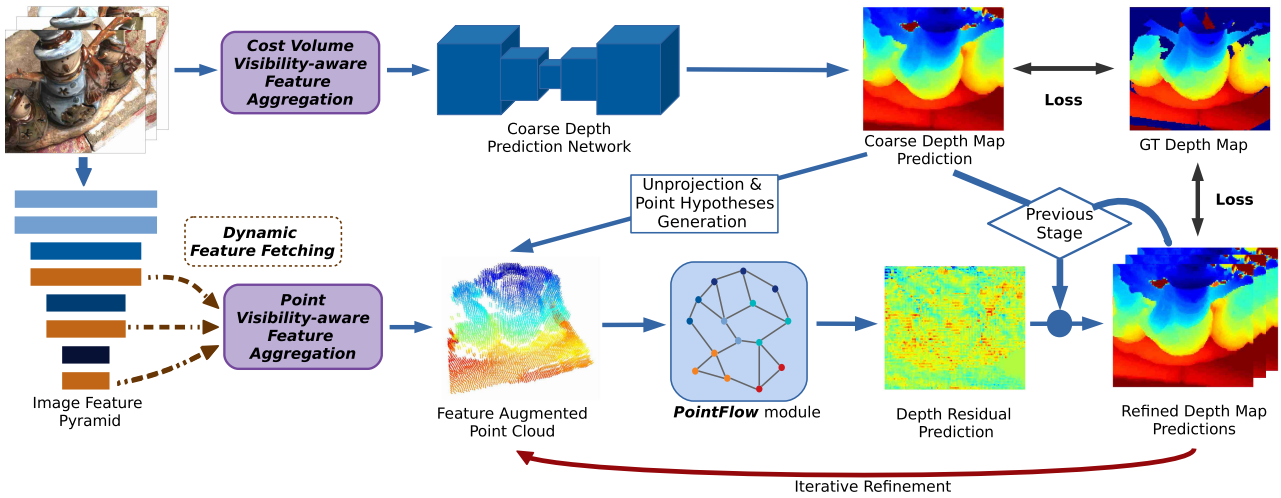


Fig. 2. *Overview of VA-Point-MVSNet architecture.* The visibility-aware feature aggregation module aggregates the multi-view image appearance cues to generate visibility-robust features for coarse depth prediction and depth refinement separately. A coarse depth map is first predicted with low GPU memory and computation cost and then unprojected to a point cloud along with hypothesized points. For each point, the feature is fetched from the multi-view image feature pyramid dynamically. The *PointFlow* module uses the feature-augmented point cloud for depth residual prediction, and the depth map is refined iteratively along with up-sampling.

methods achieve better results than 2D learning-based methods.

More recently, Luo *et al.* [30] proposed to use a learnable patchwise aggregation function and apply isotropic and anisotropic 3D convolutions on the 3D cost volume to improve the matching accuracy and robustness. Yao *et al.* [31] proposed to replace 3D CNNs with recurrent neural networks, which leads to improved memory efficiency. Xue *et al.* [32] proposed MVSCRF, where multi-scale conditional random fields (MSCRFs) are adopted to constraint the smoothness of depth prediction explicitly. Instead of using voxel grids, in this paper we propose to use a point-based network for MVS tasks to take advantage of 3D geometry learning without being burdened by the inefficiency found in 3D CNN computation.

Besides depth map prediction, deep learning can also be used to refine depth maps and fuse them into a single consistent reconstruction [33].

High-Resolution & Hierarchical MVS. High-resolution MVS is critical to real applications such as robot manipulation and augmented reality. Traditional methods [17], [34], [35] generate dense 3D patches by expanding from confident matching key-points repeatedly, which is potentially time-consuming. These methods are also sensitive to noise and change of viewpoint owing to the usage of hand-crafted features. Hierarchical MVS generates high-resolution depth maps in a coarse-to-fine manner, which reduces unnecessary computation and leads to improved efficiency. For classic methods, hierarchical MI (mutual information) computation is utilized to initialize and refine disparity maps [36], [37]. And learning-based methods are proposed to predict the residual of the depth map from warped images [38] or by constructing cascade narrow cost volume [39], [40]. In our work, we use point clouds as the representation of the scene, which explicitly encodes the spatial position and relationship as important cues for depth residual prediction and also is more flexible for potential applications (e.g., foveated depth inference).

Point-Based 3D Learning. Recently, a new type of deep network architecture has been proposed in [41], [42], which is able to process point clouds directly without converting them to volumetric grids. Compared with voxel-based methods, this kind of architecture concentrates on the point cloud data and saves unnecessary computation. Also, the continuity of space is preserved during the process. While PointNets have shown significant performance and efficiency improvement in various 3D understanding tasks, such as object classification and detection [42], it is under exploration how this architecture can be used for MVS task, where the 3D scene is unknown to the network. In this paper, we propose *PointFlow* module, which estimates the 3D flow based on joint 2D-3D features of *point hypotheses*.

3 Method

This section describes the detailed network architecture of VA-Point-MVSNet (Fig. 2). Our method can be divided into two steps, coarse depth prediction, and iterative depth refinement. First, we introduce the visibility-aware feature aggregation (Section 3.1), which reasons about the visibility of source images from image appearance cues and aggregates multi-view image information while considering visibility. The visibility-aware feature aggregation is utilized in both coarse depth prediction and iterative depth refinement. Second, we describe the coarse depth map prediction. Let $\mathbf{I_0}$ denote the reference image and $\{\mathbf{I_i}\}_{i=1}^{N}$ denote a set of its neighboring source images. Since the resolution is low, the existing volumetric MVS method has sufficient efficiency and can be used to predict a coarse depth map for $\mathbf{I_0}$ (Section 3.2). Then we describe the 2D-3D feature lifting (Section 3.3), which associates the 2D image information with 3D geometry priors. Finally we propose our novel *PointFlow* module (Section 3.4) to iteratively refine the input depth map to higher resolution with improved accuracy.

3.1 Visibility-Aware Feature Aggregation

The main intuition for depth estimation is multi-view photo-consistency, that image projections of the reconstructed shape should be consistent across visible images.

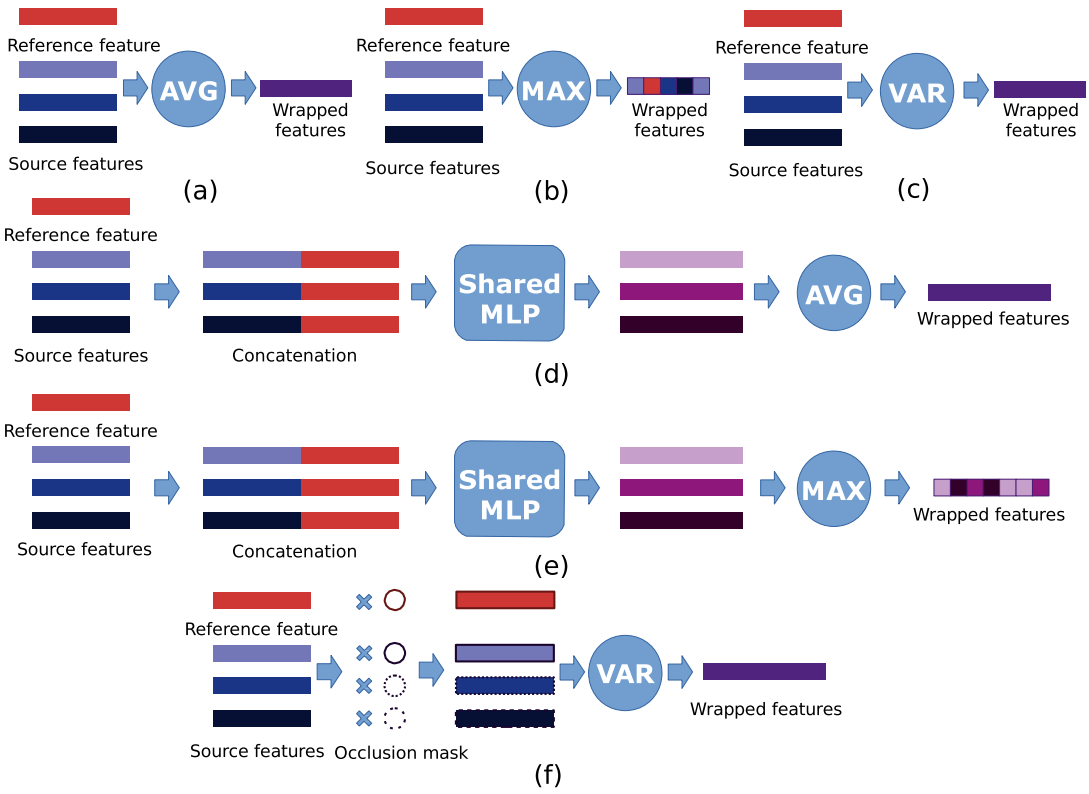


Fig. 3. *Different structures of multi-view feature aggregation.* (a) 'avg': average operation applied in [43]; (b) 'max': element-wise max-pooling operation; (c) 'var': variance operation applied in [4]; (d)'vis-avg': our visibility-aware average operation; (e) 'vis-max': our visibility-aware max-pooling operation; (f)'vis-var': our visibility-aware variance operation. "Shared MLP" stands for multi-layer perceptron with shared weights for all input features. Batchnorm and ReLU are used for all layers except the last layer.

Thus, features of multiple input views are aggregated together for later depth prediction. Various methods for feature aggregation have been proposed, such as average [43] and variance [4]. However, previous methods are not able to handle invisible views and can lead to incorrect feature matching. So we design novel feature aggregation structures that take visibility into consideration.

Let $\mathbf{F_0}$ denote the reference view feature, $\mathbf{F_i}(\mathbf{i} = \mathbf{1}, \ldots, \mathbf{N})$ denote the source view feature, and $\mathbf{C}$ denote the generated feature. The multi-view feature aggregation processes $\{\mathbf{F_i}\}$ to generate $\mathbf{C}$

$$\mathbf{C} = \mathbf{f}(\{\mathbf{F_i}\}), \mathbf{i} = \mathbf{0}, \ldots, \mathbf{N}. \tag{1}$$

We observe that effective multi-view feature aggregation structures should meet the following properties:

- Arbitrary number of views. As the number of input views can be different for training and evaluation, the feature aggregation should be generalizable to an arbitrary number of views.
- Unordered. 3D reconstruction is irrelevant to the order of input views. Thus our feature aggregation module should be invariant to the permutation of source view feeding order.
- Differentiable. As shown in Fig. 2, the feature aggregation module serves as an interface between image feature extraction and 3D geometry inference. It has to be differentiable such that the whole framework can be trained end-to-end.
- Robust to invisibility. The invisible views will obstruct the photo-consistency across views for ground truth surface. Under the multi-view photo-consistency assumption, depth estimation can be wrong if we try to match the feature of invisible views, including occlusion and out of FOV. In other words, the feature aggregation process needs to take each input view visibility into consideration.

Following the above philosophy, we compare feature aggregation operations applied in recent works [4], [43] and propose novel aggregation structures to address the visibility issue. See Fig. 3 for visualization. The average operation (Fig. 3a) applied in [43] meets some properties, but it expresses no information about photo-consistency across views and cannot handle the visibility problem. The element-wise max-pooling operation (Fig. 3b) also does not take into account the relationship of appearances between the reference view and source views. The variance operation (Fig. 3c) used in [4] measures the appearance difference across all views but does not count for the visibility issue.

Vis-Avg. We first propose the visibility-aware average operation (Fig. 3d) by including a network to infer the visibility for each source view. Note that pixels are all visible for the reference view $\mathbf{F_0}$. Each source view $\mathbf{F_i}$ is concatenated with the reference view $\mathbf{F_0}$ to predict the relationship separately. The subsequent average operation combines the learned relationship across all source views and generates $\mathbf{C}$.

Vis-Max. Similarly, a visibility-aware max-pooling operation (Fig. 3e) is designed by replacing the average operation with max-pooling. From the relationship between

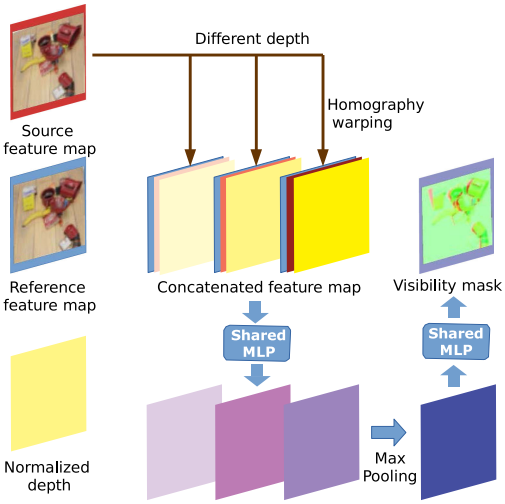


Fig. 4. Network architecture of the visibility prediction module. The module takes as input the feature map of reference view $\mathbf{F}_0$ and that of source view $\mathbf{F}_i$, and outputs the visibility mask for view i. "Shared MLP" stands for multi-layer perceptron with shared weights across all pixels, such that the visibility of each pixel is predicted independently. The element-wise sigmoid function is applied on the last layer to constrain the output in the range of $[0, 1]$. Normalized depth is computed by $\tilde{d}^k = (d^k - d_{min})/(d_{max} - d_{min})$.

the reference view and each source view, the max-pooling operation is able to preserve the most significant one for further depth prediction while excluding others, which improves the robustness to potential causes of mismatches like occlusion and highlights.

Vis-Var. Furthermore, we propose a visibility-aware variance operation (Fig. 3f)

$$\mathbf{C} = \frac{\sum_{\mathbf{i}=0}^{\mathbf{N}}(\omega_\mathbf{i} \cdot \mathbf{F_i}^2)}{\sum_{\mathbf{i}=0}^{\mathbf{N}} \omega_\mathbf{i}} - \left(\frac{\sum_{\mathbf{i}=0}^{\mathbf{N}}(\omega_\mathbf{i} \cdot \mathbf{F_i})}{\sum_{\mathbf{i}=0}^{\mathbf{N}} \omega_\mathbf{i}}\right)^2, \tag{2}$$

where $\omega_i \in [0, 1]$ is the visibility mask of view i. Note that $\omega_0 = 1$ as we predict the depth map in the reference view. For source views, a neural network module is applied for visibility prediction. The visibility module architecture is explained in the paragraph below. Our 'vis-var' operation measures the appearance variance across all visible views explicitly.

As demonstrated in [20], for visible views, the correlation between the reference view and source view with regard to depth shows a significant local maximum at the correct depth. To utilize this observation, our visibility prediction module (Fig. 4) warps the source feature map into different depth planes at the reference camera. The projected source view feature is concatenated with the reference view feature and the normalized depth $\tilde{d}^k$. The visibility module takes the compromised feature and aggregates processed depth plane features using max-pooling operation. Note that max-pooling operation can handle an arbitrary number of depth planes during training and evaluation. The visibility prediction module can be trained either in a supervised fashion or an unsupervised fashion as in Section 4.1.3.

3.2 Coarse Depth Prediction

Recently, learning-based MVS [3], [4], [43] achieves state-of-the-art performance using multi-scale 3D CNNs on cost volume regularization. However, this step could be extremely memory expensive as the memory requirement is increasing cubically as the cost volume resolution grows. Taking memory and time into consideration, we use the recently proposed MVSNet [4] to predict a relatively low-resolution depth map.

Given the images and corresponding camera parameters, MVSNet [4] builds a 3D cost volume upon the reference camera frustum. Then the initial depth map for the reference view is regressed through multi-scale 3D CNNs and the soft argmin [9] operation. In MVSNet, feature maps are downsampled to $1/4$ of the original input image in each dimension and the number of virtual depth planes is 256 for both training and evaluation. On the other hand, in our coarse depth estimation network, the cost volume is constructed with feature maps of $1/8$ the size of the reference image, containing 48 and 96 virtual depth planes for training and evaluation, respectively. Therefore, our memory usage of this 3D feature volume is about $1/20$ of that in MVSNet. Moreover, the visibility-aware feature aggregation (Section 3.1) is adopted for the cost volume construction, where MVSNet utilizes 'var' which does not count for the visibility issue.

3.3 2D-3D Feature Lifting

Image Feature Pyramid. Learning-based image features have been demonstrated to be critical to boosting up dense pixel correspondence quality [4], [44]. In order to endow points with a larger receptive field of contextual information at multiple scales, we construct a 3-scale feature pyramid. 2D convolutional networks with stride 2 are applied to downsample the feature map, and each last layer before the downsampling is extracted to construct the final feature pyramid $\{\mathbf{F_i^1}, \mathbf{F_i^2}, \mathbf{F_i^3}\}$ for image $\mathbf{I_i}$. Similar to common MVS methods [4], [43], feature pyramids are shared among all input images.

Dynamic Feature Fetching. The point feature used in our network is compromised of the aggregated multi-view image feature $\mathbf{C_p}$ with the normalized 3D coordinates in the world space $\mathbf{X_p}$. We will introduce them separately.

Image appearance features for each 3D point can be fetched from the multi-view feature maps using a differentiable unprojection given corresponding camera parameters. Note that features $\mathbf{F_i^1}, \mathbf{F_i^2}, \mathbf{F_i^3}$ are at different image resolutions, thus the camera intrinsic matrix should be scaled at each level of the feature maps for correct feature warping. The feature of view i is generated using concatenation

$$\mathbf{F_i} = \text{concat}[\mathbf{F_i^1}, \mathbf{F_i^2}, \mathbf{F_i^3}]. \tag{3}$$

To form the features residing at each 3D point, we aggregate the multi-view features using the visibility-aware feature aggregation module (Section 3.1) and do a concatenation of the aggregated feature and the normalized point coordinates

$$\mathbf{G_p} = \text{concat}[\mathbf{C_p}, \mathbf{X_p}]. \tag{4}$$

This feature-augmented point $\mathbf{G_p}$ is the input to our *PointFlow* module.

As shall be seen in the next section, since we are predicting the depth residual iteratively, we update the point

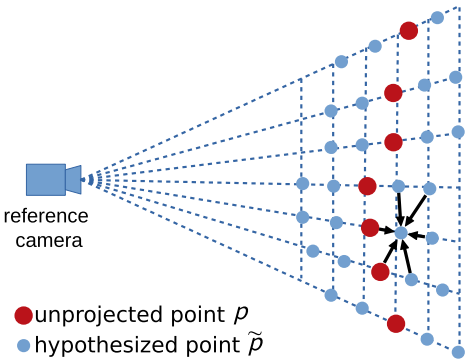


Fig. 5. Illustraion of point hypotheses generation and edge construction: For each unprojected point $\mathbf{p}$, the $2m$ point hypotheses $\{\tilde{\mathbf{p}}_\mathbf{k}\}$ are generated along the reference camera direction. Directed edges are constructed between each hypothesized point and its *kNN* points for edge convolution.

position $\mathbf{X_p}$ after each iteration and fetch the point feature from the multi-view images, an operation we name as *dynamic feature fetching*. Note that this step is distinct from cost-volume-based methods, by which the fetched features at each voxel are determined by the fixed space partition of the scene. In contrast, our method can fetch features from different areas of images dynamically according to the updated point position. Therefore, we can concentrate on the regions of interest in the feature maps, instead of treating them uniformly.

3.4 *PointFlow*

Depth maps generated from Section 3.2 have limited accuracy due to the low spatial resolution of 3D cost volume. We propose *PointFlow*, our novel approach to iteratively refine the depth map.

With known camera parameters, we first unproject the depth map to be a 3D point cloud. For each point, we aim to estimate its displacement to the ground truth surface along the reference camera direction by observing its neighboring points from all views, so as to push the points to *flow* to the target surface. Next, we discuss the components of our module in detail.

Point Hypotheses Generation. It is non-trivial to regress the depth displacement of each point from the extracted image feature maps. Due to perspective transformation, the spatial context embedded in 2D feature maps cannot reflect the proximity in 3D euclidean space.

In order to facilitate the modeling of network, we propose to generate a sequence of *point hypotheses* $\{\tilde{\mathbf{p}}_\mathbf{k}\}$ with different displacements along the reference camera direction as shown in Fig. 5. Let $\mathbf{t}$ denote the normalized reference camera direction, and s denote the displacement step size. For an unprojected point $\mathbf{p}$, its hypothesized point set $\{\tilde{\mathbf{p}}_\mathbf{k}\}$ is generated by

$$\tilde{\mathbf{p}}_\mathbf{k} = \mathbf{p} + \mathbf{kst}, \quad \mathbf{k} = -\mathbf{m}, \ldots, \mathbf{m}. \tag{5}$$

These *point hypotheses* gather the necessary neighborhood image feature information at different depths, and combine the image information with spatial geometry relationship, which helps the network to infer the displacement.

Edge Convolution. Classical MVS methods have demonstrated that local neighborhood is important for robust depth prediction. Similarly, we take the strategy of recent work DGCNN [45] to enrich feature propagation between neighboring points. As shown in Fig. 5, a directed graph is constructed on the point set using k nearest neighbors (*kNN*), such that local geometric structure information could be used for the feature propagation of points.

The edge convolution is defined as

$$\mathbf{G}'_\mathbf{p} = \underset{\mathbf{q}\in\mathbf{kNN(p)}}{\square} \mathbf{h}_\Theta\big(\mathbf{G}_\mathbf{p},\ \mathbf{G}_\mathbf{p} - \mathbf{G}_\mathbf{q}\big), \tag{6}$$

where $\mathbf{G}_\mathbf{p}, \mathbf{G}_\mathbf{q}$ are the input features of center point $\mathbf{p}$ and neighbor point $\mathbf{q}$, $\mathbf{G}'_\mathbf{p}$ is the output feature of point $\mathbf{p}$, h_Θ is a learnable non-linear function parameterized by Θ, and $\square$ is a channel-wise symmetric aggregation operation. There are multiple options for the symmetry operation, including max-pooling, average pooling, and weighted sum. We compared max-pooling and average pooling and observed similar performance after tuning hyper-parameters carefully(see Fig. 13).

Flow Prediction. The network architecture for flow prediction is shown in Fig. 6. The input is a feature-augmented point cloud, and the output is a depth residual map. We use three EdgeConv layers to aggregate point features at different scales of the neighborhood. Shortcut connections are used to include all the EdgeConv outputs as local point features. Finally, a shared MLP is used to transform the point-wise features, which outputs a probability scalar with softmax among hypothesized points of each unprojected point. The displacement of the unprojected points are predicted as the probabilistic weighted sum of the displacement among all point hypotheses

$$\Delta d_p = \mathbf{E}(\mathbf{ks}) = \sum_{\mathbf{k}=-\mathbf{m}}^{\mathbf{m}} \mathbf{ks} \times \mathbf{Prob}(\tilde{\mathbf{p}}_\mathbf{k}). \tag{7}$$

Note that this operation is differentiable. The output depth residual map is obtained by projecting the displacement back,

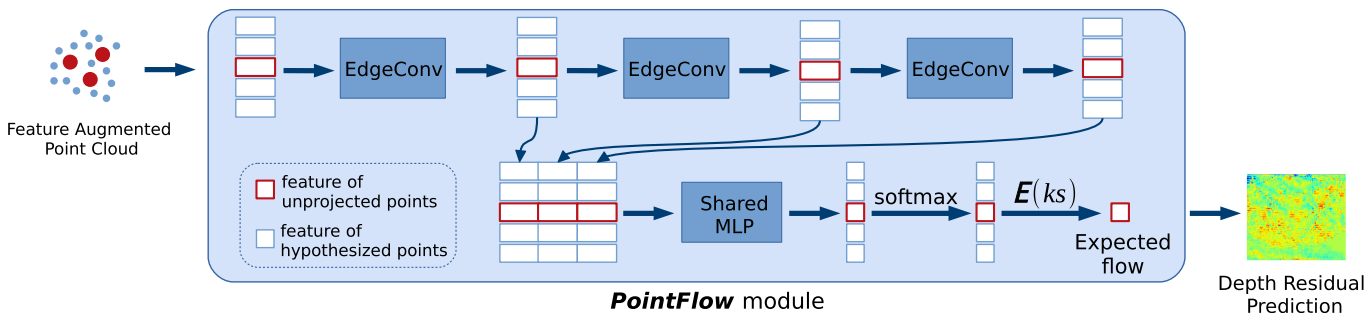


Fig. 6. Network architecture of the proposed *PointFlow* module.

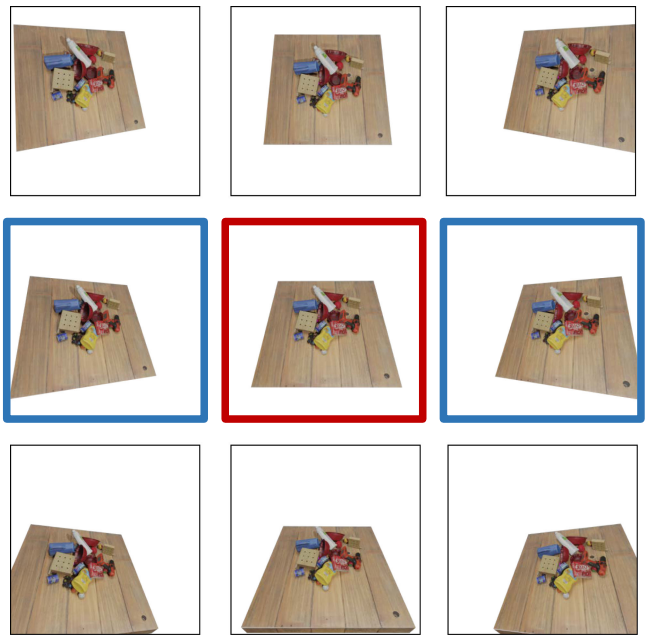


Fig. 7. Illustration of our synthetic dataset: rendered images for 9 known camera configurations, image with red border is the reference view, images with blue border are the source views used for training.

TABLE 1
Comparison Results of Predicted Depth Accuracy for Different Multi-View Feature Aggregation Structures

Coarse depth map						
structure	$< \delta_c$	$< 3\delta_c$	Invis $< \delta_c$	Invis $< 3\delta_c$	Fully-vis $< \delta_c$	Fully-vis $< 3\delta_c$
avg	0.8931	0.9842	0.8226	0.9652	0.9170	0.9907
max	0.9347	0.9887	0.8744	0.9725	0.9552	0.9943
var	0.9187	0.9862	0.8099	0.9507	0.9499	0.9964
vis-avg	**0.9478**	**0.9917**	**0.9032**	**0.9817**	**0.9630**	0.9951
vis-max	0.9462	0.9908	0.9016	0.9806	0.9615	0.9943
vis-var	0.9242	0.9878	0.8315	0.9577	0.9510	**0.9965**
Refined depth map						
structure	$< \delta_f$	$< 3\delta_f$	Invis $< \delta_f$	Invis $< 3\delta_f$	Fully-vis $< \delta_f$	Fully-vis $< 3\delta_f$
avg	0.7847	0.9485	0.8107	0.9353	0.9130	0.9866
max	0.8863	0.9652	0.8396	0.9443	0.9553	0.9913
var	0.8679	0.9636	0.8054	0.9384	0.9524	**0.9926**
vis-avg	**0.9035**	0.9706	0.8636	**0.9518**	0.9581	0.9902
vis-max	0.8982	**0.9721**	**0.8643**	0.9493	**0.9602**	0.9902
vis-var	0.8899	0.9665	0.8407	0.9391	0.9529	0.9908

Inlier ratios are used as metrics. "Invis" stands for areas in the reference view that are invisible in at least one source view, and "Fully-vis" stands for areas that are visible in all source views. $\delta_c = 19.15\,mm, \delta_f = 7.18\,mm$.

which will be added to the initial input depth map for depth refinement.

Iterative Refinement With Upsampling. Because of the flexibility of our point-based network architecture, the flow prediction can be performed iteratively, which is much harder for 3D cost-volume-based methods, because the space partitioning is fixed after the construction of cost volume. For depth map $\mathbf{D}^{(\mathbf{i})}$ from coarse prediction or former residual prediction, we can first upsample it using nearest neighbor to higher spatial resolution and then perform the flow prediction to obtain $\mathbf{D}^{(\mathbf{i+1})}$. Moreover, we decrease the depth interval s between the unprojected points and hypothesized points at each iteration, so that more accurate displacement can be predicted by capturing more detailed features from closer point hypotheses.

3.5 Training Loss

Similar to most deep MVS networks, we treat this problem as a regression task and train the network with the L_1 loss, which measures the absolute difference between the predicted depth map and the ground truth depth map. Losses for the initial depth map and iteratively refined ones are all considered

$$Loss = \sum_{i=0}^{l} \left(\frac{\lambda^{(i)}}{s^{(i)}} \sum_{p \in \mathbf{P}_{\text{valid}}} \left\| \mathbf{D}_{\text{GT}}(\mathbf{p}) - \mathbf{D}^{(\mathbf{i})}(\mathbf{p}) \right\|_1 \right), \tag{8}$$

where $\mathbf{P}_{\text{valid}}$ represents the valid ground truth pixel set and l is the iteration number. The weight $\lambda^{(i)}$ is set to 1.0 in training for all the iterations.

4 Experiments

We now present an extensive and comprehensive evaluation on both synthetic and real data. Due to the fact that the depth captured for real data is usually not complete and accurate, which is indispensable for visibility mask computation, we first generate a small synthetic dataset to justify the effectiveness of our visibility-aware multi-view feature aggregation structures (Section 4.1). Second, we evaluate our VA-PointMVSNet on the DTU benchmark [8] (Section 4.2). Then we provide detailed experiments to analyze our network design (Section 4.3). At last we show generalizability and potential applications of our network (Sections 4.4 and 4.5).

4.1 Experiments on Synthetic Data

4.1.1 Dataset Generation

We generate a synthetic dataset as shown in Fig. 7 using the YCB object dataset [46]. We first generate physically-plausible densely cluttered layouts for multiple objects using the physics simulator MuJoCo [47]. Then each scene is rendered with 9 given camera configurations. We use path tracing renderer for realistic looking. The resolution of rendered images is 512×512. The intensities and positions of lights are set randomly for each scene. We create 400 training scenes and 100 testing scenes. The networks are trained with the 3 views with red or blue border in Fig. 7 and tested with source views chosen from the 8 views that can be novel to the networks. Since the visibility only depends on the local photo-consistency, we assume the networks are able to learn to infer the visibility from the local image information even for views that are unseen in the training set.

4.1.2 Comparison of Multi-View Aggregation Structures

All the 6 structures in Section 3.1 are trained and evaluated on the synthetic dataset. The coarse prediction network is trained alone for 40 epochs, and then, the model is trained end-to-end for another 176 epochs. The batch size is set as 4. The number of views is set as 3 and 7 for training and evaluation, respectively. Percentage of inliers of depth map ($\Delta d < \delta, \Delta d < 3\delta$) are used as evaluation metrics and comparison result is shown in Table 1. $\delta_c = 19.15\,mm, \delta_f = 0.375 * \delta_c =$

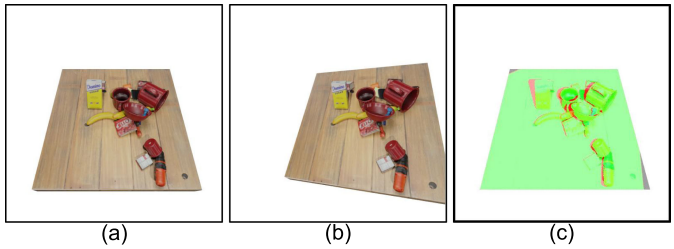


Fig. 8. Ground truth visibility mask: (a) reference view; (b) source view; (c) ground truth visibility mask computed by checking depth consistency between (a) and (b), green represents visible and red represents occluded.

$7.18\,mm$ for coarse depth map and refined depth map respectively, which are the depth intervals between virtual planes or point hypotheses. The hyper-parameter 0.375 is chosen empirically based on the experiments on the synthetic dataset and DTU dataset. Our visibility-aware multi-view feature aggregation module can improve the depth prediction accuracy significantly, especially in areas that are invisible in source views, because by taking into account the appearance difference between the reference view and each source view, our module can aggregate visible multi-view appearance effectively while excluding misguiding invisible views.

4.1.3 Supervised Learning Versus Unsupervised Learning

In our 'vis-var' structure, we predict the visibility mask of each source view explicitly. Given the complete and accurate depth maps generated from the synthetic data, we obtain the ground truth visibility masks through cross-check (Fig. 8). Our visibility prediction is supervised by the ground truth mask. An L_1 loss term for visibility prediction is added to our depth prediction loss (8) as

$$L_{vis} = \lambda_{vis} \sum_{p \in \mathbf{P_{valid}}} \left\| \mathbf{V_{GT}}(\mathbf{p}) - \mathbf{V_{pred}}(\mathbf{p}) \right\|_1, \tag{9}$$

where $\mathbf{V_{GT}}$ is the ground truth visibility mask which equals 1 at visible pixels and 0 at invisible ones. The weight λ_{vis} is set to 0.01 in training.

Table 2 compares the predicted depth accuracy of supervised learning with unsupervised learning. Furthermore, we add the result using ground truth visibility masks for training and evaluation to validate the effectiveness of visibility masks. Although unsupervised learning achieves better overall prediction accuracy, supervision on visibility mask improves the prediction accuracy in invisible areas. Moreover, the result of ground truth masks achieves the best accuracy for invisible regions, which demonstrates accurate visibility masks are crucial for MVS reconstruction.

4.1.4 Ablation Study on Visibility Prediction Module

In order to help the network distinguish the spatial order of virtual planes before max-pooling, we concatenate the normalized depth of each plane with warped features. Table 3 shows the ablation study result, which demonstrates the effectiveness of introducing depth information for visibility prediction.

TABLE 2
Comparison Results of Supervised Learning and Unsupervised Learning of Our 'vis-var' Structure

	$< \delta$	$< 3\delta$	Invis $< \delta$	Invis $< 3\delta$	Fully-vis $< \delta$	Fully-vis $< 3\delta$
supervised	0.8899	**0.9665**	**0.8407**	**0.9391**	0.9529	**0.9908**
unsupervised	**0.8930**	0.9661	0.8397	0.9378	**0.9534**	0.9901
ground truth	0.9037	0.9738	0.8560	0.9471	0.9604	0.9924

$\delta = 7.18\,mm$.

TABLE 3
Ablation Study on Introduction of Normalized Depth in Our Visibility Prediction Module

	$< \delta$	$< 3\delta$	Invis $< \delta$	Invis $< 3\delta$	Fully-vis $< \delta$	Fully-vis $< 3\delta$
w/ depth	**0.8899**	**0.9665**	**0.8407**	**0.9391**	**0.9529**	**0.9908**
w/o depth	0.8856	0.9628	0.8307	0.9333	0.9488	0.9901

$\delta = 7.18\,mm$.

4.2 Experiments on DTU Benchmark

4.2.1 Dataset Introduction

The DTU dataset [8] is a large-scale MVS dataset, which consists of 124 different scenes scanned in 7 different lighting conditions at 49 or 64 positions. The data for each scan is composed of an RGB image and corresponding camera parameters. The dataset is split into training, validation, and evaluation sets following MVSNet [4].[1]

4.2.2 Implementation Details

Training. We train VA-Point-MVSNet on the DTU training dataset. For data pre-processing, we follow MVSNet [4] to generate depth maps from the given ground truth point clouds. During training, we set input image resolution to 640×512, and number of views to $N = 3$. The input view set is chosen with the same view selection strategy as in MVSNet. For coarse prediction, we construct a 3D cost volume with $D = 48$ virtual depth planes, which are uniformly sampled from 425 mm to 921 mm. For the depth refinement step, we set flow iteration number $l = 2$, with depth intervals being 8 mm and 4 mm, respectively. The number of nearest neighbor points is 16. We use RMSProp [48] of initial learning rate 0.0005 which is decreased by 0.9 for every 2 epochs. The coarse prediction network is trained alone for 4 epochs, and then, the model is trained end-to-end for another 20 epochs. Batch size is set to 4 on 4 NVIDIA GTX 1080Ti graphics cards.

Evaluation. We use $D = 96$ depth layers for initial depth prediction and set flow iterations $l = 3$ for depth refinement. We predict the reference view depth map for each $N = 5$ view set. Then we fuse all depth maps to point clouds using the same post-processing provided by [4]. The image resolution for evaluation is 1280×960.

Implementation of kNN. Naïve *kNN* of point cloud of N points can be memory-consuming with $O(N^2)$ complexity. However, we notice the *kNN* of a point tend to come from its nearby 2D pixels in the depth map. By leveraging this fact and taking the hypothesized points into consideration,

1. Validation set: 18 scans {3, 5, 17, 21, 28, 35, 37, 38, 40, 43, 56, 59, 66, 67, 82, 86, 106, 117}. Evaluation set: 22 scans {1, 4, 9, 10, 11, 12, 13, 15, 23, 24, 29, 32, 33, 34, 48, 49, 62, 75, 77, 110, 114, 118}. Training set: the other 79 scans.

TABLE 4
Quantitative Results of Reconstruction Quality on the DTU Evaluation Dataset (Lower is Better)

	Acc. (mm)	Comp. (mm)	Overall (mm)
Camp [50]	0.835	0.554	0.695
Furu [17]	0.613	0.941	0.777
Tola [51]	0.342	1.190	0.766
Gipuma [49]	**0.283**	0.873	0.578
SurfaceNet [3]	0.450	1.040	0.745
MVSNet [4]	0.396	0.527	0.462
R-MVSNet [31]	0.383	0.452	0.417
Ours-'var'	0.361	0.421	0.391
Ours-'vis-var'	0.379	0.374	0.377
Ours-'vis-avg'	0.475	0.467	0.471
Ours-'vis-max'	0.359	**0.358**	**0.359**

we restrict the *kNN* search of each point from the whole point cloud to its $k \times k \times (2m+1)$ neighborhood. Furthermore, we parallel the distance computation by using a fixed weight 3D kernel.

Post-Processing. Similar to MVSNet [4], our post-processing is composed of three steps: photo-metric filtering, geometric consistency filtering, and depth fusion. For photometric filtering, we use the predicted probability of the most likely depth layer as the confidence metric and filter out points whose confidence is below a threshold. The filtering threshold is set to 0.2 and 0.1 for coarse and our *PointFlow* stage, respectively. For geometric consistency, we calculate the discrepancy of predicted depths among multi-view predictions through reverse-projection. Points with discrepancy larger than 0.12mm are discarded. For depth fusion, we take the average value of all reprojected depths of each point in visible views as the final depth prediction and produce the 3D point cloud.

4.2.3 *Benchmarking*

We evaluate the proposed method on the DTU evaluation dataset. Quantitative results are shown in Table 4 and Fig. 9,

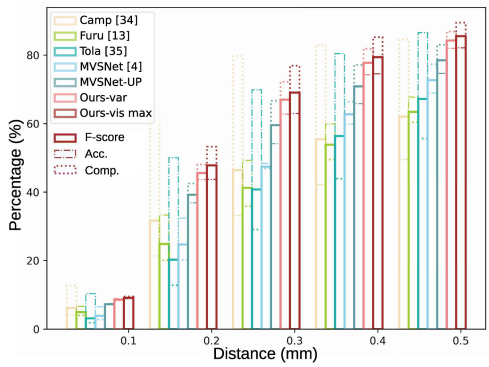


Fig. 9. F-score, accuracy, and completeness of different distance thresholds on the DTU evaluation dataset (higher is better). For a fair comparison, we upsample the depth map predicted by MVSNet to the same resolution as our method before depth fusion (288×216 to 640×480). The reconstruction results of Gipuma [49] and SurfaceNet [3] are not publicly available.

where the accuracy and completeness are computed using the official code from the DTU dataset, and the *f-score* is calculated as mentioned in [9] as the measure of the overall performance of accuracy and completeness. While Gipuma [49] performs the best in terms of accuracy, our VA-Point-MVSNet outperforms start-of-the-art in both completeness and overall quality, and our visibility-aware feature aggregation can improve the reconstruction completeness significantly. Qualitative results are shown in Fig. 10. VA-Point-MVSNet generates a more detailed point cloud compared with MVSNet. Especially in those edgy areas, our method can capture high-frequency geometric features.

4.3 Network Design Analysis

4.3.1 PointFlow *Iteration*

Because of the continuity and flexibility of point representation, the refinement and densification can be performed iteratively on former predictions to give denser and more accurate predictions. While the model is trained using $l = 2$

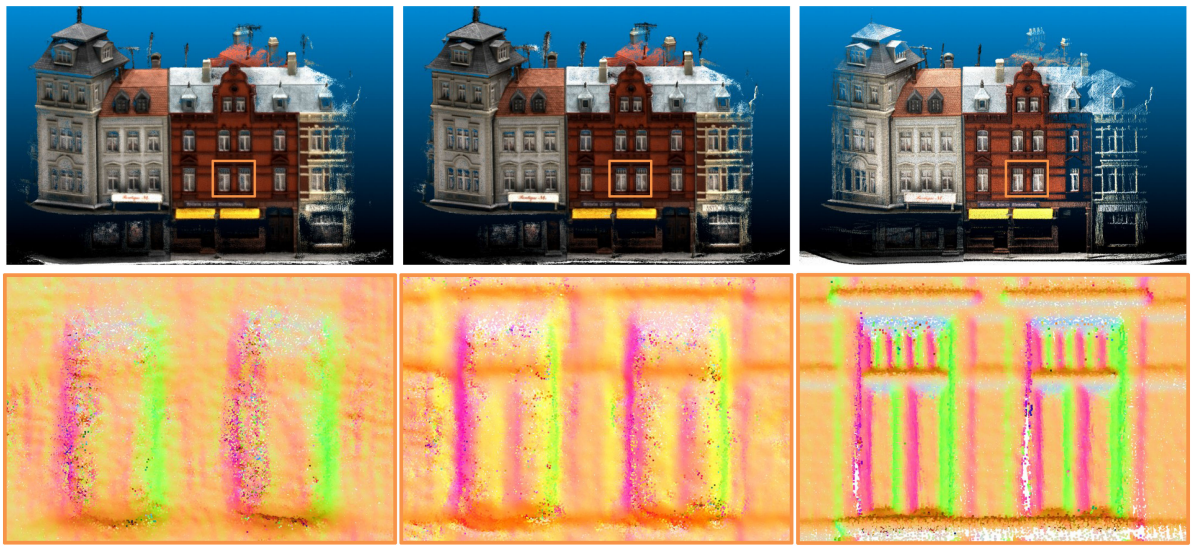


Fig. 10. Qualitative results of *Scan* 9 of DTU dataset. Top: Whole point cloud. Bottom: Visualization of normals in zoomed local area. Our VA-Point-MVSNet generates detailed point clouds with more high-frequency components than MVSNet. For a fair comparison, the depth maps predicted by MVSNet are interpolated to the same resolution as our method.

TABLE 5
Comparison Result at Different Flow Iterations Measured by Reconstruction Quality and Depth Map Resolution on the DTU Evaluation Set

Iter.	Acc. (mm)	Comp. (mm)	Overall (mm)	0.5mm *f-score*	Depth Map Res.	Depth Interval (mm)	GPU Mem. (MB)	Runtime (s)
-	0.693	0.758	0.726	47.95	160×120	5.30	**7219**	**0.34**
1	0.674	0.750	0.712	48.63	160×120	5.30	7221	0.61
2	0.448	0.487	0.468	76.08	320×240	4.00	7235	1.14
3	**0.361**	**0.421**	**0.391**	**84.27**	**640×480**	**0.80**	8731	3.35
MVSNet[4]	0.456	0.646	0.551	71.60	288×216	2.65	10805	1.05

Due to the GPU memory limitation, we decrease the resolution of MVSNet [4] to $1152 \times 864 \times 192$.

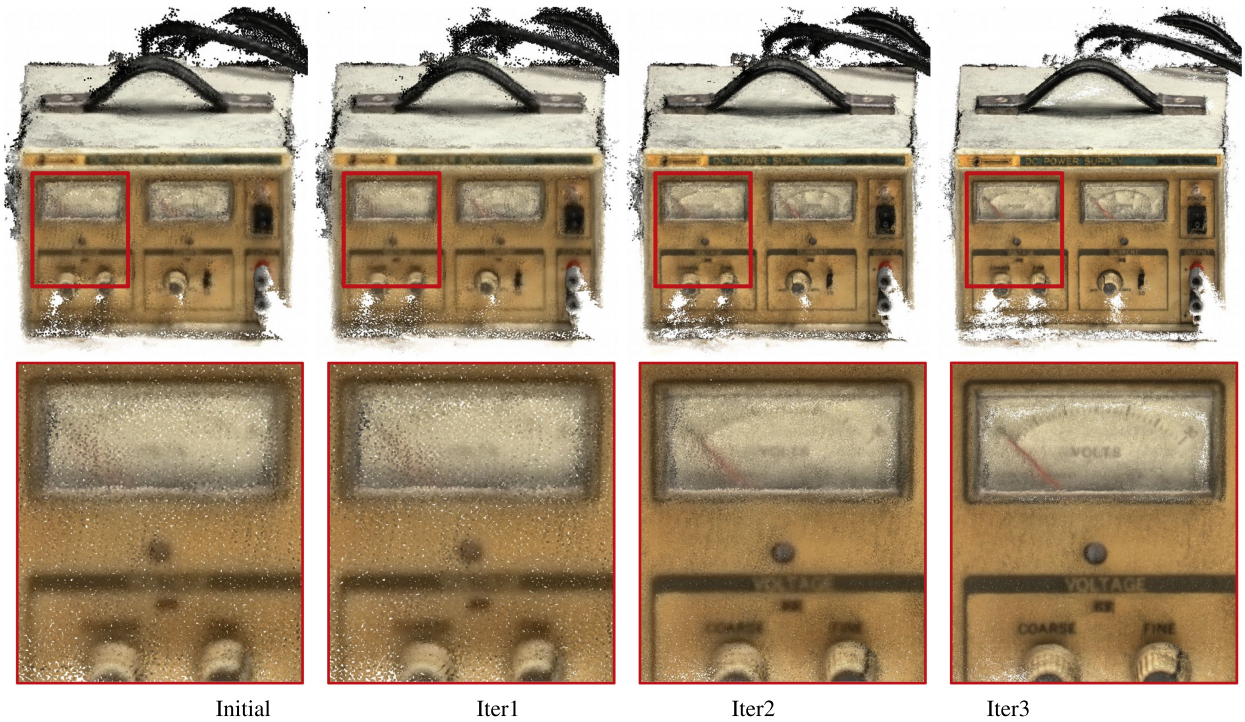


Fig. 11. Qualitative results at different flow iterations. Top: Whole point cloud. Bottom: Zoomed local area. The generated point cloud becomes denser after each iteration, and more geometry details can be captured.

iterations, we test the model using iteration ranging from 0 to 3. For each iteration, we upsample the point cloud and decrease the depth interval of point hypotheses simultaneously, enabling the network to capture more detailed features. We compare the reconstruction quality, depth map resolution, GPU memory consumption and runtime at different iterations, along with performance reported by state-of-the-art methods in Table 5. We use 'var' feature aggregation structure for the comparison. The reconstruction quality improves significantly with multiple iterations, which verifies the effectiveness of our method. Note that our method already outperforms the state-of-the-art after the second iteration. Qualitative results are shown in Fig. 11.

4.3.2 Feature Aggregation Structure

In this section, we validate the effectiveness of different multi-view feature aggregation structures with regard to the number of views. All the structures can process an arbitrary number of input views. Although the model is trained using $N = 3$, we can evaluate the model using either $N = 2, 3, 5, 7$ on the DTU evaluation set as shown in Fig. 12.

When the number of input views increases from 2 to 5, the reconstruction qualities of all the structures improve, which is consistent with common knowledge of MVS reconstruction. However, when the number of input views increases from 5 to 7, the qualities of 'var' and 'vis-var' drop, which demonstrates that image information from unideal views may lead to corrupted reconstruction if the visibility issue is not considered. The qualities of 'vis-avg' and 'vis-max' keep improving for $N = 7$, validating the effectiveness of our visibility-aware multi-view feature aggregation structures.

4.3.3 Ablation Study

In this section, we provide ablation experiments and quantitative analysis to evaluate the strengths and limitations of the key components in our framework. For all the following studies, experiments are performed and evaluated on the DTU dataset, and both *accuracy* and *completeness* are used to measure the reconstruction quality. We set the iteration number to $l = 2$, use 'var' feature aggregation structure, and all other experiment settings are the same as Section 4.2.3.

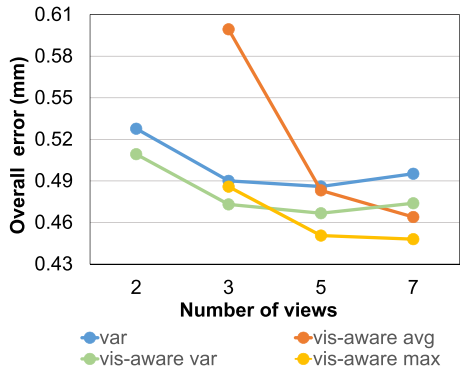


Fig. 12. Influence of number of input views on reconstruction quality on the DTU evaluation set for different multi-view aggregation structures. 'vis-avg' and 'vis-max' fail to predict reasonable depth when $N = 2$.

TABLE 6
Ablation Study on Network Architectures on the DTU Evaluation Dataset, Which Demonstrates the Effectiveness of Different Components

EDGE	EUCNN	PYR	Acc. (mm)	Comp. (mm)
✓	✓	✓	**0.448**	**0.487**
✓	✓	×	0.455	0.489
✓	×	✓	0.455	0.492
×	✓	✓	0.501	0.518
✓	×	×	0.475	0.504
×	✓	×	0.574	0.565
×	×	✓	0.529	0.532

EDGE denotes edge convolution, EUCNN denotes grouping by nearest neighbor points in euclidean distance, and PYR denotes the usage of image feature pyramid.

Edge Convolution. By replacing the edge convolution in Equation (6) with geometry-unaware feature aggregation

$$\mathbf{G}'_{\mathbf{p}} = \square_{\mathbf{q}\in \mathbf{kNN}(\mathbf{p})} \mathbf{h}_{\Theta}(\mathbf{G}_{\mathbf{q}}), \tag{10}$$

where the features of neighbor points are treated equally with no regard to their geometric relationship to the centroid point, the reconstruction quality drops significantly as shown in Table 6, which illustrates the importance of local neighborhood relationship information (captured by $\mathbf{G_p} - \mathbf{G_q}$) for feature aggregation.

euclidean Nearest Neighbor. In this part, we construct the directed graph using points belonging to adjacent pixels in the reference image, instead of searching the *kNN* points, which leads to decreased reconstruction quality. The reason is that, for images of 3D scenes, near-by pixels may correspond to distant objects due to occlusion. Therefore, using neighboring points in the image space may aggregate irrelevant features for depth residual prediction, leading to descending performance.

Feature Pyramid. In this part, the point cloud only fetches features from the last layer of the feature map, instead of from the whole feature pyramid. As shown in Table 6, in contrast to the relatively stable performance for changing edge convolution strategies as discussed above, the drop will be significant in the absence of the other two components, which demonstrates the effectiveness of the leveraging context information at different scales for feature fetching.

TABLE 7
Ablation Study of Different Number of Point Hypotheses m on the DTU Evaluation Set [8]

Point Hypotheses	Acc.(mm)	Comp.(mm)	Overall(mm)
1	**0.442**	0.515	0.479
2	0.448	**0.487**	**0.468**
3	0.468	0.499	0.484
3 (m=3)	0.453	0.497	0.475

(The model is trained with $m = 2$ except the last one.)

TABLE 8
Comparison Result of Direct Regression and Point Hypotheses on the DTU Evaluation Set [8]

Overall error (mm)	Direct regression	Ours
Iter0	0.751	**0.726**
Iter1	0.756	**0.712**
Iter2	0.501	**0.468**
Iter3	0.464	**0.391**

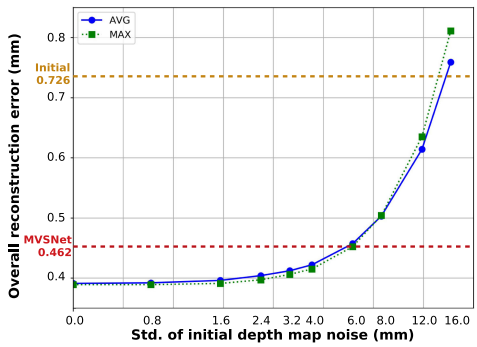


Fig. 13. Reconstruction error w.r.t. initial depth map noise. AVG denotes average pooling in EdgeConv, MAX denotes max-pooling in EdgeConv.

Point Hypotheses. We choose $m = 2, 3$ for the training and $m = 1, 2, 3$ for the test, and conduct the evaluation on the DTU evaluation set [8]. Table 7 shows the comparison result. Our proposed algorithm achieves the best reconstruction quality in terms of completeness and overall quality when the number of point hypotheses is the same as training.

We also directly regress the depth residual without the facilitation of point hypotheses and the comparison result is shown in Table 8. Although the direct regression is also able to improve the reconstruction quality after each iteration, it is still worse than the flow prediction facilitated by point hypotheses, which shows the necessity of neighborhood image feature information to accurate flow prediction.

4.3.4 Reliance on Initial Depth Maps

Our method uses state-of-the-art approaches to get a coarse depth map prediction, which is then iteratively refined by predicting depth residuals. We find that our approach is robust to noisy initial depth estimation in a certain range through the following experiments. We add Gaussian noise of different scales to the initial depth map and evaluate the reconstruction error. Fig. 13 shows that the error increases slowly and is smaller than MVSNet within 6 mm noise.

TABLE 9
Comparison of Reconstruction Quality on the DTU Evaluation Dataset With PU-Net [52]

	Acc. (mm)	Comp. (mm)	Overall (mm)
PU-Net [52]	1.220	0.667	0.943
Ours	**0.361**	**0.421**	**0.391**

4.3.5 Comparison to Point Cloud Upsampling

Our work can also be considered as a data-driven point cloud upsampling method with assisting information from multi-view images. Therefore, we compare our method with PU-Net [52], where multi-level features are extracted from the coarse point cloud to reconstruct an upsampled point cloud.

We use the same coarse depth prediction network as in our model, and train PU-Net to upsample the coarse point cloud. We use the same joint loss as mentioned in their paper, which consists of two losses—the Earth Mover's Distance (EMD) [53] loss between the predicted point cloud and the reference ground truth point cloud and a repulsion loss. For evaluation, the PU-Net is applied on the coarse predicted point cloud twice to generate a denser point cloud with 16 times more points. Quantitative result is shown in Table 9. Our VA-Point-MVSNet can generate a more accurate point cloud from the coarse one by inducing *flow* for each point from observation of context information in multi-view images.

4.4 Generalizability of the *PointFlow* Module

In order to evaluate the generalizability of our *PointFlow* module, we test it on the Tanks and Temples intermediate dataset [9], which is a large outdoor dataset captured in complex environments. We first generate coarse depth maps using MVSNet [4], and then apply our *PointFlow* module to refine them. The *f-score* increases from 43.48 to 48.70 (larger is better) and the rank rises from 45.75 to 32.25 (lower is better, date: Jan. 30, 2020). Table 10 shows the results of published learning-based approaches. Reconstructed point clouds are shown in Fig. 14.



Fig. 15. Illustration of foveated depth inference with our proposed method. Different point density levels are denoted by different colors: Gray for sparsest, Brown for intermediate, Green for densest.

4.5 Foveated Depth Inference

The point-based network architecture enables us to process an arbitrary number of points. Therefore, instead of upsampling and refining the whole depth map, we can choose to only infer the depth in the ROI based on the input image or the predicted coarse depth map. As shown

TABLE 10
Quantitative Results of Published Learning-Based Methods on Tanks and Temples Benchmark [9]

Method	Rank	Mean	Family	Francis	Horse	Lighthouse	M60	Panther	Playground	Train
P-MVSNet [30]	12.38	55.62	70.04	44.64	40.22	65.20	55.08	55.17	60.37	54.29
VA-Point-MVSNet(Ours)	32.25	48.70	61.95	43.73	34.45	50.01	52.67	49.71	52.29	44.75
R-MVSNet [31]	36.00	48.40	69.96	46.65	32.59	42.95	51.88	48.80	52.00	42.38
MVSCRF [32]	37.12	45.73	59.83	30.60	29.93	51.15	50.61	51.45	52.60	39.68
MVSNet [4]	45.75	43.48	55.99	28.55	25.07	50.79	53.96	50.86	47.90	34.69



Fig. 14. Reconstruction results on the intermediate set of Tanks and Temples [9].

in Fig. 15, we generate a point cloud of three different density levels by only upsampling and refining the ROI in the previous stage.

5 Conclusion

We present a novel visibility-aware point-based architecture for high-resolution multi-view stereo reconstruction. Instead of building a high-resolution cost volume, our proposed VA-Point-MVSNet processes the scene as a point cloud directly. The *PointFlow* module concentrates on the neighborhood of the target surface and refines the predicted depth map iteratively, which therefore achieves improved time and memory efficiency. Experiments on synthetic data and real data demonstrate that the visibility-aware multi-view feature aggregation module can improve the reconstruction quality significantly by considering visibility, and VA-Point-MVSNet is able to produce high-quality reconstruction point clouds on benchmarks. Additionally, VA-Point-MVSNet is applicable to foveated depth inference to greatly reducing unnecessary computation, which cannot be easily implemented for cost-volume-based methods.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFE0206200, and in part by the National Natural Science Foundation of China (NSFC) under Grant U1613205 and Grant 51675291, NSF Grant IIS-1764078, gifts from Qualcomm, Adobe and support from DMAI.

References

[1] C. Häne, C. Zach, J. Lim, A. Ranganathan, and M. Pollefeys, "Stereo depth map fusion for robot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 1618–1625.
[2] S. Agarwal *et al.*, "Building rome in a day," *Commun. ACM*, vol. 54, no. 10, pp. 105–112, 2011.
[3] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang, "SurfaceNet: An end-to-end 3D neural network for multiview stereopsis," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2307–2315.
[4] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "MVSNet: Depth inference for unstructured multi-view stereo," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 767–783.
[5] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang, "DeepMVS: Learning multi-view stereopsis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2821–2830.
[6] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2088–2096.
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[8] H. Aanæs, R. R. Jensen, G. Vogiatzis, E. Tola, and A. B. Dahl, "Large-scale data for multiple-view stereopsis," *Int. J. Comput. Vis.*, vol. 120, no. 2, pp. 153–168, Nov. 2016.
[9] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 78:1–78:13, Jul. 2017.
[10] R. Chen, S. Han, J. Xu, and H. Su, "Point-based multi-view stereo network," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1538–1547.
[11] C. R. Dyer, "Volumetric scene reconstruction from multiple views," in *Foundations of Image Understanding*. Berlin, Germany: Springer, 2001, pp. 469–489.
[12] A. Hornung and L. Kobbelt, "Robust and efficient photo-consistency estimation for volumetric 3D reconstruction," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 179–190.
[13] A. Hornung and L. Kobbelt, "Hierarchical volumetric multi-view stereo reconstruction of manifold surfaces based on dual graph embedding," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, vol. 1, pp. 503–510.
[14] C. H. Esteban and F. Schmitt, "Silhouette and stereo fusion for 3D object modeling," *Comput. Vis. Image Understanding*, vol. 96, no. 3, pp. 367–392, 2004.
[15] A. Zaharescu, E. Boyer, and R. Horaud, "TransforMesh: A topology-adaptive mesh-based approach to surface evolution," in *Proc. Asian Conf. Comput. Vis.*, 2007, pp. 166–175.
[16] S. N. Sinha, P. Mordohai, and M. Pollefeys, "Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, 2007, pp. 1–8.
[17] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1362–1376, Aug. 2010.
[18] M. Bleyer, C. Rhemann, and C. Rother, "PatchMatch stereo-stereo matching with slanted support windows," in *Proc. Brit. Mach. Vis. Conf.*, 2011, vol. 11, pp. 1–11.
[19] P. Heise, S. Klose, B. Jensen, and A. Knoll, "PM-Huber: PatchMatch with huber regularization for stereo matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2360–2367.
[20] G. Vogiatzis, C. H. Esteban, P. H. Torr, and R. Cipolla, "Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 12, pp. 2241–2246, Dec. 2007.
[21] Y. Liu, Q. Dai, and W. Xu, "A point-cloud-based multiview stereo algorithm for free-viewpoint video," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 3, pp. 407–418, May/Jun. 2010.
[22] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys, "Pixelwise view selection for unstructured multi-view stereo," in *Proc. Eur. Conf. Comput. Vis.*, 2016, vol. 9907, pp. 501–518.
[23] Q. Xu and W. Tao, "Multi-scale geometric consistency guided multi-view stereo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5483–5492.
[24] A. Romanoni and M. Matteucci, "TAPA-MVS: Textureless-aware patchmatch multi-view stereo," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 10412–10421.
[25] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "MatchNet: Unifying feature and metric learning for patch-based matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3279–3286.
[26] A. Seki and M. Pollefeys, "SGM-Nets: Semi-global matching with neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 231–240.
[27] P. Knobelreiter, C. Reinbacher, A. Shekhovtsov, and T. Pock, "End-to-end training of hybrid CNN-CRF models for stereo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2339–2348.
[28] A. Kendall *et al.*, "End-to-end learning of geometry and context for deep stereo regression," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 66–75.
[29] A. Kar, C. Häne, and J. Malik, "Learning a multi-view stereo machine," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 365–376.
[30] K. Luo, T. Guan, L. Ju, H. Huang, and Y. Luo, "P-MVSNet: Learning patch-wise matching confidence aggregation for multi-view stereo," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 10451–10460.
[31] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan, "Recurrent MVSNet for high-resolution multi-view stereo depth inference," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5520–5529.
[32] Y. Xue *et al.*, "MVSCRF: Learning multi-view stereo with conditional random fields," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4311–4320.
[33] S. Donne and A. Geiger, "Learning non-volumetric depth fusion using successive reprojections," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7634–7643.
[34] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 418–433, Mar. 2005.
[35] A. Owens, J. Xiao, A. Torralba, and W. Freeman, "Shape anchors for data-driven multi-view reconstruction," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 33–40.
[36] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008.
[37] M. Rothermel, K. Wenzel, D. Fritsch, and N. Haala, "SURE: Photogrammetric surface reconstruction from imagery," in *Proc. LC3D Workshop*, 2012, vol. 8, Art. no. 2.

[38] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan, "Cascade residual learning: A two-stage convolutional neural network for stereo matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 887–895.
[39] H. Zhou, B. Ummenhofer, and T. Brox, "DeepTAM: Deep tracking and mapping," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 822–838.
[40] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, "Cascade cost volume for high-resolution multi-view stereo and stereo matching," 2019, *arXiv: 1912.06378*.
[41] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.
[42] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.
[43] S. Im, H.-G. Jeon, S. Lin, and I. S. Kweon, "DPSNet: End-to-end deep plane sweep stereo," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=ryeYHi0ctQ
[44] C. Tang and P. Tan, "BA-Net: Dense bundle adjustment networks," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=B1gabhRcYX
[45] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, 2019, Art. no. 146.
[46] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proc. Int. Conf. Adv. Robot.*, 2015, pp. 510–517.
[47] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.
[48] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6A overview of mini-batch gradient descent," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2012.
[49] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 873–881.
[50] N. D. F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla, "Using multiple hypotheses to improve depth-maps for multi-view stereo," in *Proc. Eur. Conf. Comput. Vis.*, 2008, vol. 5302, pp. 766–779.
[51] E. Tola, C. Strecha, and P. Fua, "Efficient large-scale multi-view stereo for ultra high-resolution image sets," *Mach. Vis. Appl.*, vol. 23, no. 5, pp. 903–920, Sep. 2012.
[52] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-Net: Point cloud upsampling network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2790–2799.
[53] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 605–613.



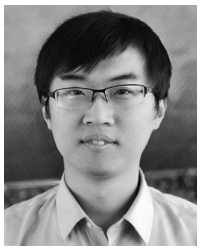
Rui Chen received the BE degree in mechanical engineering from Tsinghua University, Beijing, China, in 2014, where he is currently working toward the PhD degree in the Department of Mechanical Engineering. His research interests include 3-D measurement and cyber physical systems.



Songfang Han received the PhD degree from the Hong Kong University of Science and Technology, Hong Kong under the supervision of Prof. Pedro Sander. She visited University of California, San Diego under the supervision of Prof. Hao Su for 2018 December to 2019 June. Her research interests center around 3D reconstruction, geometry processing, and efficient rendering.



Jing Xu received the BE degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2003, and the PhD degree in mechanical engineering from Tsinghua University, Beijing, China, in 2008. He was a postdoctoral researcher with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, Michigan. He is currently an associate professor with the Department of Mechanical Engineering, Tsinghua University, Beijing, China. His research interests include vision-guided manufacturing, image processing, and intelligent robotics.



Hao Su has been an assistant professor of computer science and engineering, UC San Diego, since July 2017. He is affiliated with the Contextual Robotics Institute and Center for Visual Computing. He served on the program committee of multiple conferences and workshops on computer vision, computer graphics, and machine learning. He is the area chair of ICCV'19, CVPR'19, senior program chair of AAAI'19, IPC of Pacific Graphics'18, program chair of 3DV'17, publication chair of 3DV'16, and chair of various workshops at CVPR, ECCV, and ICCV. He is also invited as keynote speakers at workshops and tutorials in NIPS, 3DV, CVPR, RSS, ICRA, S3PM, etc.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.