360MVSNet: Deep Multi-view Stereo Network with 360° Images for Indoor Scene Reconstruction

Ching-Ya Chiu¹  Yu-Ting Wu²  I-Chao Shen³  Yung-Yu Chuang¹
¹National Taiwan University  ²National Taipei University  ³The University of Tokyo

Abstract

Recent multi-view stereo methods have achieved promising results with the advancement of deep learning techniques. Despite the progress, due to the limited fields of view of regular images, reconstructing large indoor environments still requires collecting many images with sufficient visual overlap, which is quite labor-intensive. 360° images cover a much larger field of view than regular images and would facilitate the capture process. In this paper, we present 360MVSNet, the first deep learning network for multi-view stereo with 360° images. Our method combines uncertainty estimation with a spherical sweeping module for 360° images captured from multiple viewpoints in order to construct multi-scale cost volumes. By regressing volumes in a coarse-to-fine manner, high-resolution depth maps can be obtained. Furthermore, we have constructed EQMVS, a large-scale synthetic dataset that consists of over 50K pairs of RGB and depth maps in equirectangular projection. Experimental results demonstrate that our method can reconstruct large synthetic and real-world indoor scenes with significantly better completeness than previous traditional and learning-based methods while saving both time and effort in the data acquisition process.

1. Introduction

Multi-view stereo (MVS) is a classic computer vision problem that aims to estimate the dense representation of a scene given multi-view images and calibrated cameras. After decades of research, traditional methods have achieved good results on surfaces with rich textures and Lambertian materials by combining hand-crafted similarity metrics with engineered regularization [34, 15, 14]. Recently learning-based methods [44, 19, 29, 45, 20, 41] further improved the overall reconstruction quality with better completeness by relaxing the restrictions of surface appearance.

While these works generate promising results in 3D reconstruction, most of them use normal field-of-view (FoV) images as input, thus requiring densely distributed capturing with enough overlaps. For large indoor scenes, collecting the input images become very labor-intensive. People have to capture several hundreds of photos to cover the spatial and angular information of a single room. As demonstrated in Figure 1 even with three hundred input images, the results reconstructed by neither a traditional method COLMAP [34] nor a learning-based method MVSNet [44] achieve satisfactory completeness.

At the same time, the fast improvement of consumer-level 360° cameras has made 360° images a popular data source for autonomous driving systems, virtual reality, and robotics. 360° images can provide broader coverage of a scene in a single shot than the normal FoV ones, thus requir-
ing fewer shots and less effort on matching image pairs with enough overlaps. Unfortunately, to the best of our knowledge, no existing learning-based MVS methods use 360° images as input. Although it is possible to warp the 360° images to multiple perspective view images and then apply classic MVS methods, it is suboptimal as it ignores the continuous geometry information within the 360° images.

In this paper, we present the first multi-view stereo deep network for reconstructing 3D scene structure using 360° images. Similar to other depth-map-based MVS methods [38, 3, 15, 34, 44], our network takes one reference image and several source images as input at a time and infers the depth map for the reference image. All the estimated depth maps are then merged together to produce the final point cloud. The core of our method is a spherical sweeping module for constructing a cost volume with 360° images captured from multiple viewpoints. The depth map for the reference image is then estimated by regressing the cost volume. To better preserve scene details, we predict the depth map in a coarse-to-fine manner by combining uncertainty estimation. Figure 1(b) shows that with only a tiny fraction of input images, our method significantly outperforms previous methods (Figure 1(c, d)) in terms of completeness and overall quality.

Our second contribution is to provide EOMVS, a large-scale synthetic MVS dataset of 360° images. By utilizing recent large-scale 3D scanned datasets of indoor environment [2, 5], we generate more than 50K RGB images with paired ground truth depth maps in equirectangular projection using a path tracing engine [10]. We believe this dataset can benefit future research on MVS using 360° images.

We demonstrate the effectiveness of the proposed model with both synthetic and real-world data. The experimental results show that our method significantly outperforms previous traditional and learning-based methods, especially in aspect of completeness on the synthetic testing datasets. More importantly, our method only needs a tiny fraction of input images than previous methods, thus saving lots of efforts on data acquisition. We also demonstrate that our model trained on the synthetic dataset could generalize well to real-world scenes without any finetuning.

2. Related works

Traditional multi-view stereo. Traditional multi-view stereo methods use hand-crafted similarity metrics and engineered regularizations to analyze the photo consistency over image patches. They can be roughly categorized by their output representation of the scene, including voxels, point clouds, and depth maps. Voxel-based methods [11, 26, 35] divide the space into voxels and determine whether a voxel is on the surface or not. Point cloud based methods [27, 14] directly reconstruct a sparse point cloud of a scene with spatial consistency assumption, then gradually densify the results. Approaches with depth map representation [22, 25, 8, 15, 38] estimate the depth map of each view given a few source images and merge multiple depth maps into the final point cloud with a specific fusion scheme. Depth map based methods are popular due to its flexibility and scalability for processing large-scale scene data and reconstructing fine-grained surface details [13]. Kang and Szeliski [21] proposed a multi-view stereo method using 360° images as its inputs. This method, however, does not exploit the advantages of deep learning.

Deep learning multi-view stereo. Traditional methods usually fail to match image pairs for textureless regions and shiny surfaces, leading to incomplete reconstructions. Recently, deep neural networks have improved the performance of these cases by building up learnable image features. Ji et al. [20] presents the first learning-based method in MVS using voxel color cube as scene representation and learn to predict the probabilities of whether a voxel belongs to a surface. Learnt Stereo Machine [23] uses differentiable unprojection operation to form the cost volume and regularize by 3D CNN to generate the result voxel grids.

MVSNet [44] adopts depth map representation and propose an end-to-end deep network for dealing with large-scale scene reconstruction. They first extract learnable features and apply a plane sweeping process to form a single cost volume. After that, per-view depth map is estimated by applying 3D CNNs to regularize the cost volume. Finally, all depth maps are fused together to infer the 3D geometry of the scene. Later methods [16, 43, 7, 6] employ coarse-to-fine architecture to solve the problem of large memory consumption in cost volume regularization and produce higher-resolution depth maps. Unlike these methods that take normal FoV images as input, we introduce 360° images into MVS, significantly boosting the reconstruction quality while requiring fewer input images.

Omnidirectional depth estimation and stereo matching. Several works have been proposed for solving the stereo problem with fisheye or 360° images. Previous works focus on stereo matching algorithm for fisheye image [28, 17] and video [18] by supporting spherical projection. More recently, deep learning based methods are proposed to estimate depth for a single 360° image [40, 42, 47, 39, 41]. Even though the estimated depth from a 360° image pair has been greatly improved, these works often use fixed camera setting [40, 41] or limited number of input images [47, 39]. Compared to these methods, we propose a spherical sweeping algorithm for 360° images captured from several different viewpoints in order to reconstruct the scene geometry, and our method has no assumption on camera setting.

Several spherical convolution methods have been proposed to tackle the problem of distortion in omnidirectional images [36, 37, 8, 12], while they focus on object detection instead of depth estimation.
3. Method

Our goal is to reconstruct a 3D scene structure from a set of 360° images. Similar to previous approaches [44], our method runs over the images and predicts their depth maps one at a time. Each time, our method picks up a reference image and estimates its depth map using \( N \) nearby source images. After all depth maps are estimated, they are merged together to produce the final scene point cloud.

Figure 2 depicts our network architecture for predicting the depth map for a reference image with a set of source images. To generate a high-resolution depth map, our model works in a coarse-to-fine fashion. It consists of three stages at different scales and starts from the coarsest stage. In each stage, our method first extracts features for all input 360° images (Section 3.1). It then warps the source feature maps onto multiple virtual spheres centered at the reference view using the proposed 360° spherical sweeping algorithm (Section 3.2) and constructs a cost volume (Section 3.3). Finally, the cost volume is regressed into a predicted depth map (Section 3.4).

3.1. Feature Extraction

Our first step is to extract the feature maps \( \{ F_i \}_{i=0}^N \) of the \( N+1 \) input images \( \{ I_i \}_{i=0}^N \), where \( I_0 \) is the reference image and the rest \( \{ I_i \}_{i=1}^N \) are source images. To capture information at different resolutions, we employ a U-Net structure [33] with skip connections to form a multi-scale feature extractor. To be specific, the encoder comprises a set of convolutional layers, each of which is followed by a batch normalization and a ReLU activation layer. Convolution with stride 2 is used to progressively downsample the spatial dimension of the feature maps. The decoder layers upsample the features and concatenate with features from skip connection. The transposed convolution is then applied to gradually recover the image information.

For the \( i \)-th source image \( I_i \), the feature extractor extracts its feature maps \( F_i = (F_i^1, F_i^2, F_i^3) \) at three different scales. The resolution of the three feature maps are \( W \times H, \frac{W}{2} \times \frac{H}{2}, \frac{W}{4} \times \frac{H}{4} \), where \( (W, H) \) is the width and height of the input 360° image.

3.2. 360 Spherical Sweeping

We propose a novel 360 spherical sweeping module that considers the geometry information from 360° images. Our module is inspired by the plane sweeping [9, 44, 19] and fish-eye spherical sweeping algorithms [18, 17, 42, 41]. It warps feature maps of equirectangular images onto virtual spheres of the reference view with different radii to form the cost volumes. The module is fully differentiable. Thus we can seamlessly integrate it into our training process.

The major differences between our method and previous spherical sweeping algorithms are twofold. First, our network uses 360° images as input. Second, previous methods aim to estimate the omnidirectional depth map from a single viewpoint, while we adhere to the convention of most MVS methods and reconstruct the scene geometry with images captured from several different viewpoints. As a consequence, when estimating the depth map of a reference view, our 360° spherical sweeping algorithm needs to derive the relationship between two viewpoints and then build the cost volume based on the two 360° images.
3.2.1 Preliminary: Spherical coordinates

Mapping between spherical and camera coordinates. Figure 3(a) defines the spherical camera model used in this paper. In the 3D camera coordinate system of a 360° camera, a point \( P(X, Y, Z) \) can be represented by the normalized spherical coordinate \((R, \Theta, \Phi)\), where \(R, \Theta\) and \(\Phi\) are the distance to the origin, elevation angle, and azimuthal angle, respectively. We can transform the point from the 3D coordinate to a 2D pixel projection by calculating

\[
R = \sqrt{X^2 + Y^2 + Z^2} ,
\]

\[
\Phi = \begin{cases} 
\pi - \tan^{-1}(X/Z), & \text{if } X > 0 \& Z < 0 \\
-\pi - \tan^{-1}(X/Z), & \text{if } X < 0 \& Z < 0 \\
\tan^{-1}(X/Z), & \text{otherwise}
\end{cases} ,
\]

\[
\Theta = \sin^{-1}(Y/R) .
\]

Specifically, \(\Theta \in (-\pi, +\pi)\) and \(\Phi \in (-\pi, +\pi)\). We can also map the normalized spherical coordinate to the 3D camera coordinate by:

\[
P = (R \sin \Phi \cos \Theta, R \sin \Theta, R \cos \Phi \cos \Theta)^T .
\]

Mapping between image and spherical coordinates. We use an equirectangular image with a latitude-longitude projection to store the scene information captured by a 360° camera. The mapping from a pixel \((x, y)\) in the equirectangular image with resolution \(W \times H\) to its corresponding unit vector in spherical coordinate can be written as:

\[
\Phi = \frac{(x + 0.5)}{W} \times 2\pi - \pi .
\]

\[
\Theta = \frac{(y + 0.5)}{H} \times \pi - \frac{\pi}{2} .
\]

Mapping between 3D and image coordinates. The projection \(f(P)\) of a 3D point \(P(X, Y, Z)\) in the camera coordinate to a 2D pixel \(p(x, y)\) in equirectangular image coordinate can be obtained by introducing the elevation and azimuthal angles:

\[
\begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = f(P) = \begin{bmatrix} \Phi \cdot W/2\pi - 0.5 \\ \Theta \cdot H/\pi - 0.5 \\ 1 \end{bmatrix} ,
\]

where the elevation and azimuthal angles can be calculated by Eq. 2 and Eq. 3.

Given the depth value \(d\) of a 2D pixel \((x, y)\), we can also back-project \(p\) from image coordinate to its corresponding 3D coordinate \(P(X, Y, Z)\) with the inverse function of \(f(P)\):

\[
P = f^{-1}(\begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}, d) = (d \sin \Phi \cos \Theta, d \sin \Theta, d \cos \Phi \cos \Theta, 1)^T ,
\]

where \(\Phi\) and \(\Theta\) can be calculated with the image resolution by Eq. 5 and Eq. 6. Note that the depth value we estimate in spherical coordinates is the distance between the point and the origin of the sphere (i.e. \(R\)) rather than that in the conventional pinhole camera.

3.2.2 Feature map warping

To construct the cost volume, our 360 spherical sweeping warps the feature map of a source image onto a series of virtual spheres centered at the reference view with different radii based on the depth hypotheses (Figure 3(b)). We use the extrinsic parameters to connect the local camera coordinates of the reference view and source views. The projection \(M_i\) for projecting a point from the reference view to the \(i\)th source view can be obtained by concatenating the matrix \(P_0^{-1}\) for transforming a point from the local camera coordinate of the reference view to world coordinate and the matrix \(P_i\) for transforming a point from world coordinate to the local camera coordinate of the source \(i\)th view:

\[
M_i = P_i P_0^{-1} ,
\]

where \(P_i\) is the full rank \(4 \times 4\) camera matrix of the \(i\)th view constructed by \(P_i = \begin{bmatrix} R_i & t_i \\ 0^T & 1 \end{bmatrix}\), where \(R_i\) and \(T_i\) are the rotation matrix and translation vector of the \(i\)th view, respectively.

We can warp the \(i\)th source features onto the \(k\)th virtual sphere with radius \(r_k\) centered at the reference view using the following inverse warping process: for each pixel \(p(x, y)\) in the warped feature map, we first transform it to spherical coordinate and project to the 3D camera space of the reference view with the depth hypothesis \(r_k\). The 3D point is then transformed to the camera coordinate of the \(i\)th source camera using \(\Phi_{ij}\). Finally, we project the 3D point onto the feature map of the \(i\)th source view and get the source location \(p'_k(x'_k, y'_k)\), then resample the feature map with bilinear interpolation. The calculation of the source location for the warped feature map can be written as:

\[
\begin{bmatrix} p'_{kx} \\ p'_{ky} \\ 1 \end{bmatrix} = f(M_i(f^{-1}(\begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}, r_k))) .
\]

It is worth noting that our work and OmniMVS [41] deal with different scenarios. The goal of OmniMVS is to estimate an omnidirectional depth map from “a single viewpoint.” They use the information gathered from a rig of four fisheye cameras to predict a 360° depth map. In contrast, adhering to the convention of most MVS methods, we aim to reconstruct the geometry of an entire scene by using multiple 360° images captured from “multiple different viewpoints.” Therefore, when estimating the depth map of a reference view, our 360° spherical sweeping algorithm needs to derive the relationship between two viewpoints and then
construct the cost volume based on these two 360° images. This is not the case for the methods proposed by OnmiMVS and SweepNet [40].

### 3.3. Multi-scale Cost Volume

After warping the feature maps of the source views to the virtual spheres of the reference view using 360 spherical sweeping, we aggregate the warped feature maps to construct a cost volume with a variance-based cost metric. Inspired by the recent multi-scale MVS work [7, 16, 43], the cost volume is built in a coarse-to-fine fashion for storing adaptive information of depth hypotheses.

As shown in Fig. 3, the three stages construct the cost volumes using a predefined number of depth hypotheses: $D_1$, $D_2$, and $D_3$. In the first stage (the coarsest one), we uniformly sample $D_1$ depth hypotheses within a predefined depth interval $[d_{min}, d_{max}]$ because at this time we do not have any information about the scene depth.

In the second and the third stages, for each pixel in the warped feature map, we set its range of the sphere radius for depth hypothesis according to the uncertainty of the depth probability volume (discussed in Section 3.4) regularized from the previous stage. Assuming the depth values have a Gaussian distribution, we can estimate a per-pixel degree of uncertainty of the depth probability volume by calculating the standard deviation $\sigma_s$ at pixel $x$ in the stage $s$:

$$\sigma_s(x) = \sqrt{\frac{1}{D_s} \sum_{j=1}^{D_s} P_s^j \cdot (d_s^j(x) - \widehat{d}_s(x))^2},$$  \hspace{1cm} (11)

where $P_s^j$, $d_s^j(x)$, and $\widehat{d}_s(x)$ denote the probability volume of the $j^{th}$ depth hypothesis sphere, the predicted depth of the $j^{th}$ hypothesis sphere, and the $j^{th}$ depth hypothesis, respectively. $D_s$ denotes the number of hypothesis spheres.

By utilizing the distribution of the depth values estimated in the previous stage, we progressively narrow down the hypothesis range of the incoming stage based on the idea of confidence interval. We set the hypothesis range $R_s(x)$ in stage $s$ for pixel $x$ in the depth map:

$$R_s(x) = [d_{s-1}(x) - \lambda \sigma_{s-1}(x), d_{s-1}(x) + \lambda \sigma_{s-1}(x)].$$

The value of $\lambda$ is set to 1.5 for all results in this paper. We observed that the results are not sensitive to the value of $\lambda$ because the model would learn to adjust the uncertainty intervals through training. With the spatially varying uncertainty estimation, we can efficiently narrow down the depth interval and reduce the number of depth samples.

### 3.4. Depth Regression and Loss Function

We apply 3D CNNs to regularize the cost volumes and generate the probability volumes [24, 44]. Specifically, the regularization network is a 3D U-Net that consists of a series of downsampling and upsampling layers that take features in different resolutions into account. After the convolutional layer, a softmax operation is applied on the probability volume along the depth direction. Finally, the network outputs the estimated depth value as the expected value computed from all depth hypotheses:

$$D = \sum_{d=d_{min}}^{d=d_{max}} d \cdot P(d),$$

where $d_{min}$ and $d_{max}$ denote the minimum and maximum depth samples, respectively.

Following previous works [24, 44], we consider depth estimation as a multi-class regression problem, and our loss function use the $l_1$ norm to measure the difference between the ground truth depth and the estimated depth. Because
the cost volumes in three stages are regularized separately, the total loss is the weighted sum of the $l_1$ loss in the three stages:

$$\mathcal{L}(D, D_{GT}) = \sum_{s=1}^{3} \lambda_s \cdot \left( \sum_{p \in P_{valid}} \| D_{GT}^s(p) - D^s(p) \|_1 \right),$$

where $D = \{D^s\}_{s=1}^{N}$, $D_{GT} = \{D_{GT}^s\}_{s=1}^{N}$, $P_{valid}$ is the set of valid pixels in the ground truth depth, and $\lambda_s$ is the weight for $s^{th}$ stage.

4. Synthetic Dataset: EQMVS

To train our 360MVSNet with supervised learning, a dataset for multi-view stereo with equirectangular projection is indispensable. However, the existing multi-view stereo datasets such as DTU [1] contains only object-centric scenes with perspective images. To address this problem, we generate a large-scale synthetic dataset EQMVS of indoor scene images in the equirectangular format.

4.1. Data Acquisition

Similar to previous work [42], we leverage the textured meshes from two large-scale real-world indoor scene datasets: Stanford2D3D [2], and Matterport3D [5]. We use a physically-based path tracer renderer (the Cycles renderer from Blender software [10]) to render the 360° images. We split each indoor scene in the original 3D dataset into multiple smaller scenes according to the provided semantic labels. Thus, each scene in our dataset is a single room or region. For each scene, we render a set of RGB images and ground truth depth maps by placing panoramic cameras with equirectangular projection at multiple locations inside the scene. To address holes in the original 3D scene mesh, we also generate a corresponding mask for labeling the invalid pixels and exclude them in the training and testing process. To generate more dense viewpoints in a single scene, we interpolate the camera position from the original data of the 3D dataset to create a sufficient number of data.

Our EQMVS dataset comprises 1,014 scenes, represented by 53,114 triplets of RGB images, depth maps, and masks in total. We also include the camera positions for rendering the images. Compared to the previous perspective multi-view stereo datasets, we introduce a more challenging scene data with a large variety of indoor environments, rather than a single object captured by controlled camera trajectory. Our dataset can be used as a new benchmark for indoor scene reconstruction using 360° images.

5. Experiments and Results

5.1. Implementation Details

Training. Our network was implemented using the PyTorch [52] framework and trained on a single NVIDIA Quadro RTX8000 GPU. Following MVSNet [44], the number of input images is set to $N = 3$ with 1 reference image and 2 source images. We adopt a three-scale cost volume, and the number of depth hypotheses for each stage is 160, 32, and 8. We split the scenes in EQMVS into the training set containing 825 scenes and the testing set containing the remaining 189 scenes.

Post-processing. Similar to previous depth-map-based MVS methods [34, 44, 4], our method requires a post-processing step for converting the predicted depth maps to a point cloud. Because there are no existing depth fusion methods for 360° images, we apply simple filtering rules to remove outliers when merging the depth maps. We consider the photometric and geometric consistencies between the predicted depth maps: For photometric consistency, we filter out the pixel with a probability lower than 0.3. For geometric consistency, we mutually project the pixels between the views to ensure the depth values are consistent. Please refer to supplementary materials for more implementation details.

5.2. Performance

Equal-effort quantitative evaluation. We follow the standard evaluation metrics proposed by the DTU dataset [1] to compute the accuracy and completeness of the reconstructed point clouds. The accuracy metric measures the distance from the estimated point clouds to the ground truth point clouds. And the completeness metric measures the distance from the ground truth point clouds to the estimated point clouds. We also evaluate the overall score introduced by MVSH [43] which takes the average of accuracy and completeness.

We compare our methods against both traditional geometry-based methods [34, 4] and a learning-based methods, MVSNet [44], over the EQMVS test set. We do not compare with the later methods built on MVS-Net [16, 43, 7, 5] because they focus on reducing the memory consumption. We do not compare our method with OmniMVS [41] and SweepNet [42] because the objectives are
Figure 4: **Equal-effort qualitative comparisons** on example test scenes of EQMVS. We use 25 360° images to reconstruct scene (a)-(d) and 49 images for scene (e). For COLMAP, openMVS, and MVSNet, we convert each 360° image into six normal FoV images using cubemap projection. We report the accuracy/completeness/overall scores under each image.

Figure 5: Our method (the red diamond) outperforms traditional methods, (COLMAP [34] and openMVS [4]), and deep learning-based method [44] in completeness and overall score using smaller number of input images (25 v.s. 100-500). To achieve similar completeness score, previous method (openMVS [4]) requires 20× input images.

different. In this experiment, we use input images captured at the same locations for all methods, meaning it takes similar efforts to collect the input data. Because our method is the first learning-based method using 360° images as input, in order to conduct a fair comparison, the input test images are warped from equirectangular projection to cubemap projection for previous methods to avoid distortion since previous works are all designed for normal FoV im-
Figure 6: **Qualitative comparison of a real-world scene.** We compared results generated by our method and other methods on a real-world scene. Our method uses only 11 equirectangular $360^\circ$ images to reconstruct the scene, while other methods use 66 images. Compared to other methods, our method reconstructs the scene more completely.

ages. We chose the cubemap projection because it is the most commonly used representation in computer graphics for representing spherical data with perspective images. As shown in Table 1, our method outperforms all the other methods in terms of completeness and overall score.

We present the qualitative results in Figure 4. Due to space limitations, we can only present the reconstruction results from one perspective here. The supplementary materials provide the full 3D point clouds. Compared to other methods, our method is less sensitive to the areas of visual overlapping regions due to its broad FoV and continuous information within a single $360^\circ$ image. In contrast, other methods struggle to reconstruct the entire environment and often only recover a small region of the scene.

**Evaluation on the number of cameras.** The experiment in Figure 5 reveals that previous methods require about 500 images to achieve similar completeness and overall quality scores as our method, which only uses 25 images. This suggests that the user has to spend about $20 \times$ efforts on collecting data for previous methods. Our method is less impressive in terms of accuracy score. The reason is that we only apply simple filtering rules when merging the depth maps due to the lack of depth fusion algorithms for $360^\circ$ images. As a result, we cannot weed out the outliers in a robust way as other methods do. We leave it as a future work. Nevertheless, it is worth noting that previous methods fail to reconstruct the scene with only 25 images.

**Qualitative results for real-world $360^\circ$ images.** To demonstrate the generalization ability of our proposed model, we test our model with real-world images in equi-rectangular format from [30]. The camera positions of the images are recovered by OpenMVG [31]. Figure 6 shows the comparisons with other methods on a scene. Our method can reconstruct most of the scene by using only 11 $360^\circ$ images, while other methods use 66 images but can only reconstruct small fractions of the scene. Although our model is trained on the synthetic images, without any fine tuning to the real-world data, it shows high robustness on the real-world scenes.

**6. Conclusions and Future Works**

In this paper, we propose 360MVSNet, a deep learning-based multi-view stereo method that can reconstruct the 3D structure of an indoor scene from $360^\circ$ images. We propose a novel 360 spherical sweeping module and use it to construct multi-scale cost volumes. High-resolution depth maps can be obtained by regressing the cost volumes. We then merge the depth maps of all views together to reconstruct the final scene point cloud. We also present a large-scale synthetic dataset, EQMVS for training 360MVSNet. We demonstrated that our methods achieve the best reconstruction results among all the compared methods.

There are a number of limitations that deserve further investigation. First, our method requires estimating camera parameters for the input $360^\circ$ images, while the structure from motion (SfM) methods for $360^\circ$ images are not as robust as the ones for normal FoV images. Similarly, there are no existing depth map fusion algorithms for $360^\circ$ images. We would like to explore SfM and depth fusion algorithms for $360^\circ$ images. Another problem is the lack of a real-world indoor environment database with $360^\circ$ images for multi-view stereo. In the future, we would love to capture more real-world scenes with multiple $360^\circ$ images.

**Acknowledgments.** This work was supported in part by the Ministry of Science and Technology (MOST) under grants 110-2634-F-002-051. It was also funded in part by JSPS KAKENHI Grant Number JP21F20075.
References


