BiFuse: Monocular 360° Depth Estimation via Bi-Projection Fusion

Fu-En Wang[*1,3] fulton84717@gapp.nthu.edu.tw
Yu-Hsuan Yeh[*2] yuhsuan.eic08g@nctu.edu.tw
Min Sun[1,5] sunmin@ee.nthu.edu.tw
Wei-Chen Chiu[2] walon@cs.nctu.edu.tw
Yi-Hsuan Tsai[4] wasidennis@gmail.com

Abstract

Depth estimation from a monocular 360° image is an emerging problem that gains popularity due to the availability of consumer-level 360° cameras and the complete surrounding sensing capability. While the standard of 360° imaging is under rapid development, we propose to predict the depth map of a monocular 360° image by mimicking both peripheral and foveal vision of the human eye. To this end, we adopt a two-branch neural network leveraging two common projections: equirectangular and cubemap projections. In particular, equirectangular projection incorporates a complete field-of-view but introduces distortion, whereas cubemap projection avoids distortion but introduces discontinuity at the boundary of the cube. Thus we propose a bi-projection fusion scheme along with learnable masks to balance the feature map from the two projections. Moreover, for the cubemap projection, we propose a spherical padding procedure which mitigates discontinuity at the boundary of each face. We apply our method to four panorama datasets and show favorable results against the existing state-of-the-art methods.

1. Introduction

Inferring 3D structure from 2D images has been widely studied due to numerous practical applications. For instance, it is crucial for autonomous systems like self-driving cars and indoor robots to sense the 3D environment since they need to navigate safely in 3D. Among several techniques for 3D reconstruction, significant improvement has been achieved in monocular depth estimation due to the advance of deep learning and availability of large-scale 3D training data. For example, FCRN [16] achieves monocular depth estimation by their proposed up-projection module. However, most of the existing methods are designed for a camera with normal field-of-view (FoV). As 360° camera becomes more and more popular in recent years, the ability to infer the 3D structure of a camera's complete surrounding has motivated the study of monocular 360° depth estimation.

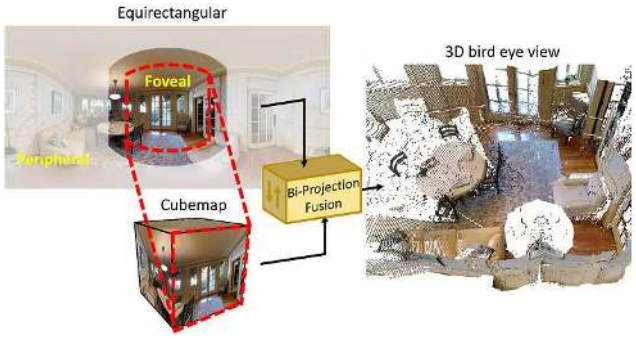

Figure 1. Our BiFuse network estimates the 360° depth from a monocular image using both equirectangular and cubemap projections. A bi-projection fusion component is proposed to leverage both projections inspired by both peripheral and foveal vision of the human eye. Given the estimated 360° depth, a complete 3D point cloud surrounding the camera can be generated to serve downstream applications.

In this paper, we propose an end-to-end trainable neural network leveraging two common projections – equirectangular and cubemap projection – as inputs to predict the depth map of a monocular 360° image. Our main motivation is to combine the capability from both peripheral and foveal vision like the human eye (see Fig. 1 for the illustration). Note that, equirectangular projection provides a wide field-of-view mimicking a peripheral vision, whereas cubemap projection provides a smaller but non-distorted field-of-view mimicking the foveal vision. On the one hand, equirectangular projection allows all surrounding information to be observed from a single 2D image but introduces distortion. On the other hand, cubemap projection avoids distortion but introduces discontinuity at the boundary of

[1]*National Tsing Hua University*
[2]*National Chiao Tung University*
[3]*ASUS AICS Department*
[4]*NEC Labs America*
[5]*MOST Joint Research Center for AI Technology and All Vista Healthcare*
[*]*The authors contribute equally to this paper.*

the cube. Considering both projections would have the complementary property to each other, where we refer to our method as *BiFuse*.

However, the FoV of the foveal vision could be too small, which degrades the effectiveness of our fusion scheme (Fig. 2). To tackle this issue, cube padding (CP) methods [26, 4] have been proposed to expand field-of-view from neighboring faces on the cube. Nevertheless, using cube padding may result in geometric inconsistency at the boundary that introduces non-negligible distortion effect. Therefore, we propose spherical padding (SP) which pads the boundary by considering the spherical geometry and reduces the boundary inconsistency. Finally, instead of naively combining features of both branches (e.g., [31]), we propose a bi-projection fusion procedure with learnable masks to balance the information shared between two projections. The source code and pretrained models are available to the public[1].

We apply our method to four panorama datasets: Matterport3D [3], PanoSUNCG [26], 360D [38] and Stanford2D3D [1]. Our experimental results show that the proposed method performs favorably against the current state-of-the-art (SOTA) methods. In addition, we present extensive ablation study for each of the proposed modules, including the spherical padding and fusion schemes. Our contributions are summarized as follows:

1. We propose an end-to-end two-branch network, which incorporates both equirectangular and cubemap projections, to mimic the combination of peripheral and foveal vision of the human eye, respectively.

2. To share the information of different projections, we propose a bi-projection fusion procedure with learnable masks to balance the information from two projections.

3. We propose spherical padding to extend the field-of-view of cubemap projection and reduce the boundary inconsistency of each face.

2. Related Work

We describe the related work regarding monocular depth estimation and 360° perception in the following.

Monocular Depth Estimation. Saxena *et al*. [20] is one of the pioneer work on learning to estimate monocular depth. After several years of development using classical machine learning approaches, deep learning contributes to the latest significant improvement in performance. Eigen *et al*. [8] first use a deep neural network to estimate the depth map from a single image. Later on, Laina *et al*. [16] utilize ResNet [12] as the encoder and propose an up-projection module for the upsampling procedure along with the reverse Huber loss to improve depth estimation. In addition, Lee *et al*. [17] try to predict depth using several cropped images and combine them in the Fourier domain. To further refine depth predictions, [2, 28, 18, 29, 30] integrate conditional random fields (CRF) into deep neural network to achieve better performance. For instance, Cao *et al*. [2] formulate depth estimation as a classification problem and use CRF to refine the final prediction.

Moreover, other attempts have been made to advance depth estimation. Fu *et al*. [10] use dilated convolution to increase the receptive field and apply the ordinal regression loss to preserve the spatial relation between each neighboring class. With photometric loss, unsupervised training for depth estimation [11, 37, 33, 34, 32, 25, 15] can be achieved. Godard *et al*. [11] use stereo pairs to predict disparity based on the left-right consistency, while Zhou *et al*. [37] propose two networks to estimate both depth and ego-motion from video sequences. In addition, Yang *et al*. [32] use depth-normal consistency to improve depth prediction. However, for the above-mentioned methods, they are designed for a camera with normal FoV without considering the property of 360° images.

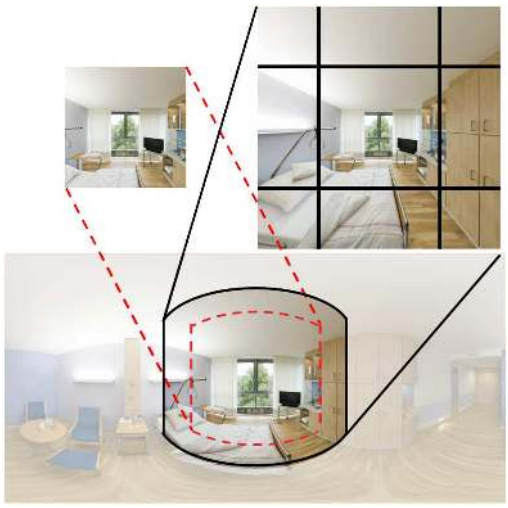

Figure 2. Field-of-view (FoV) comparison. Equirectangular projection has the largest FoV compared to each face on the cubemap projection with (solid-line) or without (dash-line) the proposed spherical padding.

360° Perception. Recently, omnidirectional cameras has become a popular media, which encourages people to work on panorama related tasks [27, 39]. For instance, due to the large field-of-view, room layout can be inferred from panorama [39, 35, 31]. However, the performance usually suffers from the distortion of equirectangular projection. To overcome this issue, several approaches are proposed. Cheng *et al*. [4] convert panorama into cubemap. For each face, they replace the original zero padding with their pro-

[1] https://fuenwang.ml/project/bifuse

posed cube padding method to remove the boundary inconsistency. Built upon [4], Wang *et al*. [26] use cubemap and cube padding for unsupervised panorama depth estimation.

To make the network aware of the distortion, spherical convolution methods are proposed recently [6, 9, 22, 23]. Considering this property, Zioulis *et al*. [38] propose OmniDepth and adopt spherical layers in [23] as the preprocessing module. However, it still remains a challenge when applying spherical CNNs using deeper networks on the depth task. Ederet *et al*. [7] tackle the 360° depth estimation as multi-task learning of depth, surface normal, and plane boundary. However, the surface normal from depth map is usually noisy especially in real-world scenarios which limits the scalability outside synthetic scenes. Different from existing works above, we improve the learning mechanism via utilizing a two-branch network from the perspective of the human eye system and propose a spherical padding scheme to maintain the geometric consistency in the cubemap representation. Our experiments shows that our method achieves state-of-the-art performance in both real-world and synthetic scenes.

3. Our Approach

In this paper, we aim to take advantage of two different representations for 360° images, equirectangular and cubemap projections, for improving the monocular 360° depth estimation. In the following, we sequentially detail the cubemap projection with our proposed spherical padding procedure in Sec. 3.1 and 3.2, bi-projection fusion scheme in Sec. 3.3, and the overall network architecture in Sec. 3.3.

3.1. Preliminary

For a cubemap representation with sides of equal length w, we denote its six faces as $f_i, i \in \{B, D, F, L, R, U\}$, corresponding to the ones on the back, down, front, left, right and up, respectively. Each face can be treated as the image plane of an independent camera with focal length $\frac{w}{2}$, in which all these cameras share the same center of projection (i.e., the center of the cube) but with different poses. When we set the origin of the world coordinate system to the center of the cube, the extrinsic matrix of each camera coordinate system can be simply defined by a rotation matrix R_{f_i} and zero translation. Given a pixel p_i on the image plane f_i with its coordinate (x, y, z) on the corresponding camera system, where $0 \leq x, y \leq w-1$ and $z = \frac{w}{2}$, we can transform it into the equirectangular representation by a simple mapping:

$$\begin{aligned} q_i &= R_{f_i} \cdot p_i \,, \\ \theta_{f_i} &= \arctan(\frac{q_i^x}{q_i^z}) \,, \\ \phi_{f_i} &= \arcsin \frac{q_i^y}{|q_i|} \,, \end{aligned} \tag{1}$$

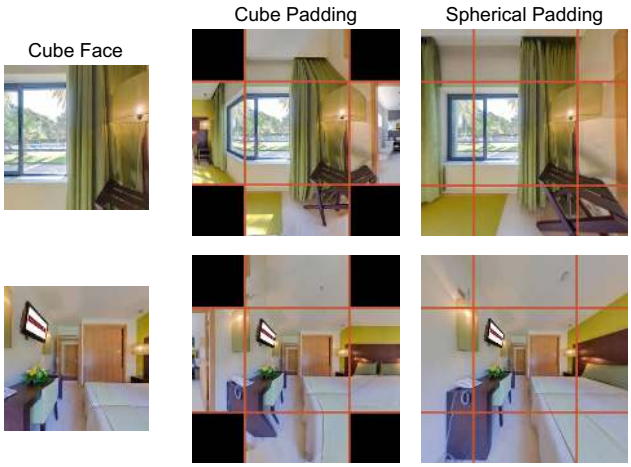

Figure 3. Spherical padding v.s. cube padding. Cube padding directly pads the feature of the connected faces. In addition to obvious inconsistency at the boundary, the values of four corners are undefined. In [4], the values are only chosen by the closest side. In our proposed spherical padding, the padding area is calculated with spherical projection. As a result, both the missing corner and inconsistency at the boundary can be addressed.

where θ_{f_i} and ϕ_{f_i} are longitude and latitude in equirectangular projection; and q_i^x, q_i^y, q_i^z are the x, y, z components of q_i respectively. As this mapping is reversible, we are able to easily perform both equirectangular-to-cube and cube-to-equirectangular transformations, which are denoted as **E2C** and **C2E**, respectively. A more detailed illustration is shown in the supplementary material.

3.2. Proposed Spherical Padding

Due to the distortion in the equirectangular projection, directly learning a typical convolutional neural network to perform monocular depth estimation on equirectangular images would lead to unstable training process and unsatisfying prediction [4]. In contrast, the cubemap representation suffers less from distortion but instead produces large errors since the discontinuity across the boundaries of each face [4, 26]. In order to resolve this issue for cubemap projection, Cheng *et al*. [4] propose the cube padding (**CP**) approach to utilize the connectivity between faces on the cube for image padding. However, solely padding the feature map of a face by using the features from its neighboring faces does not follow the characteristic of perspective projection. Therefore, here we propose the spherical padding (**SP**) method, which pads the feature according to spherical projection. As such, we can connect each face with the geometric relationship. A comparison between the cube padding [4] and our proposed spherical padding is illustrated in Fig. 3.

The most straightforward way to apply spherical padding for cubemap is to first transform all the faces into a unified

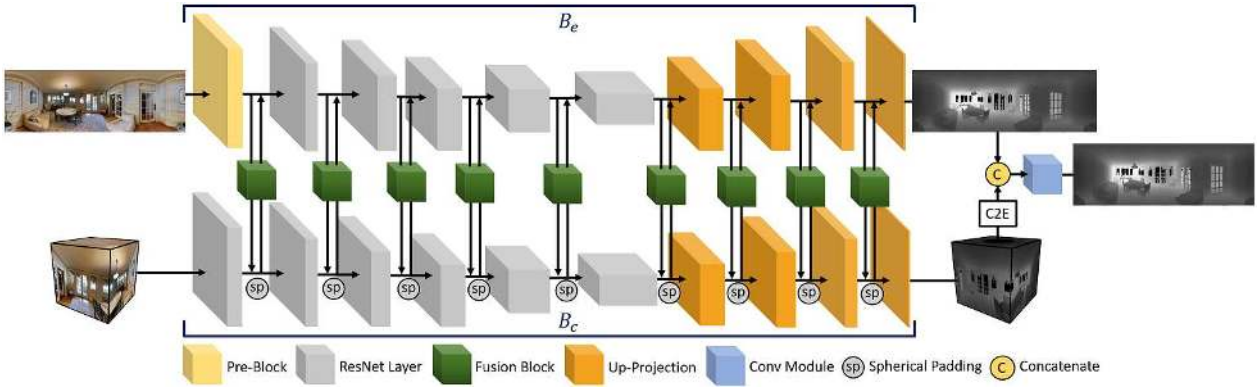

Figure 4. The proposed BiFuse Network. Our network consists of two branches B_e and B_c. The input of B_e is an RGB equirectangular image, while B_c takes the corresponding cubemap as input. We replace the first convolution layer in B_e with a Pre-Block [38, 23]. For the decoder, we adopt up-projection [16] modules. For each convolution and up-projection layer in B_c, we apply our spherical padding to connect feature maps of six faces. Most importantly, between feature maps from B_e and B_c, we use the proposed bi-projection fusion module to share information between two feature representations. Finally, we add a Conv module [24] to unify two depth predictions from B_e and B_c.

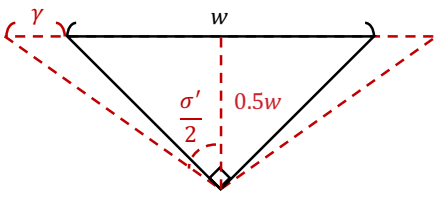

Figure 5. The cubemap with length w and padding size γ. We keep the focal length the same ($0.5w$) and calculate a new FoV σ'.

equirectangular image by C2E. Then, we extend the original FoV $\sigma = 90°$ to σ', and map it back to the cubemap by E2C. As a result, we can pad them on each face completely without missing parts (i.e., undefined areas in cube padding of Fig. 3) and with consistent geometry. Specifically, given a cubemap with side length w and Fov $\sigma = 90°$, the C2E transformation is identical to the inverse calculation of (1). When we apply spherical padding with padding size γ, which is determined by the padding size in the convolution layer (e.g.,γ=1 for a 3×3convolution layer), we update the side length of a cube face to $w' = w + 2\gamma$, and the corresponding FoV becomes $\sigma' = 2\arctan\frac{w/2+\gamma}{w/2}$ after padding, as illustrated in Fig. 5. Hence, for mapping from equirectangular image back to the padded cubemap, we should use both w' and σ' to derive the correct E2C transformation for spherical padding.

Efficient Transformation. We have described the overall concept of our spherical padding. However, the above procedure consists of both C2E and E2C transformations, which could require heavy computational cost. Therefore, we simplify this procedure by deriving a direct mapping function between two cube faces. Given two cube faces f_i and f_j, we first denote the geometric transformation between their camera coordinate systems as a rotation matrix $R_{f_i \to f_j}$. Then the mapping from a pixel p_i in f_i to f_j can be established upon the typical projection model of pinhole cameras:

$$
\begin{aligned}
K &= \begin{bmatrix} w/2 & 0 & w/2 \\ 0 & w/2 & w/2 \\ 0 & 0 & 1 \end{bmatrix}, \\
p_j &= K \cdot R_{f_i \to f_j} \cdot p_i, \\
x &= \frac{p_j^x}{p_j^z},\ y = \frac{p_j^y}{p_j^z},
\end{aligned}
\tag{2}
$$

where (x, y) represents the 2D location of p_i after being mapped onto the image plane of f_j. Since this mapping only needs to be computed once for all the pixels on the padding region, the computational cost of applying spherical padding is comparable with cube padding, without any E2C or C2E transformation included.

3.3. Proposed BiFuse Network

We have introduced our spherical padding method that enlarges the field-of-view while maintaining the geometric consistency at the boundary, to improve the cubemap representation as one branch of the proposed BiFuse network. In Fig. 4, we show our complete two-branch network motivated by the human eye system with peripheral and foveal vision.

Overall, our model consists of two encoder-decoder branches which take the equirectangular image and cubemap as input, respectively, where we denote the equirectangular branch as B_e and the cubemap one as B_c. As mentioned in Sec. 1, each branch has its benefit but also suffers from some limitations. To jointly learn a better model while sharing both advantages, we utilize a bi-projection fusion block that bridges the information across two branches, which will be described in the following. To generate the fi-

nal prediction, we first convert the prediction of cubemap to the equirectangular view and adopt a convolution module to combine both predictions.

Bi-Projection Fusion. To encourage the information shared across two branches, we empirically find that directly combining feature maps [31] from B_e and B_c would result in unstable gradients and training procedure, and thus it is keen to develop a fusion scheme to balance two branches. Inspired by the recent works in multi-tasking [5, 36], we focus on balancing the feature map from two different representations. To achieve this goal, we propose a bi-projection fusion module H: given feature maps h_e and h_c from B_e and B_c in each layer respectively, we estimate the corresponding feature maps $h'_e = H_e(h_e)$ and $h'_c = H_c(\text{C2E}(h_c))$, where H_e and H_c indicate a convolution layer.

To produce feature maps that benefit both branches, we first concatenate h'_e and h'_c, and then pass it to a convolution layer with the *sigmoid* activation to estimate a mask M to balance the fusion procedure. Finally, we generate feature maps $\bar{h}_e$ and $\bar{h}_c$ as the input to the next layer as:

$$\begin{aligned} \bar{h}_e &= h_e + M \cdot h'_c \,, \\ \bar{h}_c &= h_c + \text{E2C}((1-M)) \cdot \text{E2C}(h'_e) \,. \end{aligned} \tag{3}$$

Note that we use C2E and E2C operations in the fusion procedure to ensure that features and the mask M are in the same projection space.

Loss Function. We adopt the reverse Huber loss [16] as the objective function for optimizing predictions from both B_e and B_c:

$$\mathcal{B}(x) = \begin{cases} |x| & |x| \le c \,, \\ \frac{x^2+c^2}{2c} & |x| > c \,. \end{cases} \tag{4}$$

The overall objective function is then written as:

$$\mathcal{L} = \sum_{i \in P} \mathcal{B}(D_e^i - D_{GT}^i) + \mathcal{B}(\text{C2E}(D_c)^i - D_{GT}^i) \,, \tag{5}$$

where D_e and D_c are the predictions produced by B_e and B_c respectively; D_{GT} is the ground truth depth in the equirectangular representation; and P indicates all pixels where there is a valid depth value in the ground truth map. We note that the C2E operation is required on converting D_c into the equirectangular form before computing the loss.

Network Architecture. For each branch, we adopt the ResNet-50 [12] architecture as the encoder and use the up-projection module proposed by [16] as the decoder. Similar to [38] that considers the equirectangular property, we replace the first convolution layer of ResNet-50 in B_e with a spherical Pre-Block that has multi-scale kernels with size of (3,9), (5,11), (5,7), and (7,7), where their output feature maps are concatenated together as a 64-channel feature map and further fed into the next layer. In the cubemap branch B_c, we replace the original zero padding operation with our spherical padding among every adjacent layer (Fig. 4).

Furthermore, the proposed bi-projection fusion block as in (3) is inserted between every two layers between B_e and B_c in both encoder and decoder, in which each H_e and H_c in one fusion module contains a convolution layer which has the same channel number as the input feature map. Finally, to combine the predictions from B_e and B_c, we adopt a module with several convolution layers as in [24].

3.4. Implementation Details

We implement the network using the PyTorch [19] framework. We use Adam [14] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Our batch size is 16 and the learning rate is set to 0.0003. For training our model, we first learn B_e and B_c branches independently without using the fusion scheme as the warm-up training stage for 40 epochs, and then update only bi-projection fusion modules for another 40 epochs. Finally, we train the entire network for 20 epochs.

4. Experimental Results

In this section, we conduct experiments on four panorama benchmark datasets: Matterport3D [3], PanoSUNCG [26], 360D [38] and Stanford2D3D [1], both quantitatively and qualitatively. We mainly compare our method with the baseline FCRN [16] and the OmniDepth [38] approach, which is the current state-of-the-art for single panorama depth estimation. In addition, we compare different variants of the proposed framework to validate the effectiveness of our designed modules. Source code and models will be made available to the public.

4.1. Evaluation Metric and Datasets

We evaluate the performance by standard metrics in depth estimation, including MAE, MRE, RMSE, RMSE (log), and δ. Details of each dataset are introduced below and we use the same setting to compare all the methods.

Matterport3D. Matterport3D contains 10,800 panorama and the corresponding depth ground truth captured by Matterport's Pro 3D Camera. This dataset is the largest real-world dataset for indoor panorama scenes, which makes it challenging as the depth map from ToF sensors usually has noise or missing value in certain areas. In practice, we filter areas with missing values during training. To train and test our network, we follow the official split which takes 61 rooms for training and the others are for testing. We resize the resolution of image and depth map into 512×1024.

Stanford2D3D. Stanford2D3D is collected from three kinds of buildings in the real world, containing six large-scale indoor areas. The dataset contains 1413 panoramas

Table 1. Quantitative results on real-world datasets: Matterport3D and Stanford2D3D.

Dataset	Method	MRE ↓	MAE ↓	RMSE ↓	RMSE (log) ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
Matterport3D	FCRN [16]	0.2409	0.4008	0.6704	0.1244	0.7703	0.9174	0.9617
	OmniDepth (bn) [38]	0.2901	0.4838	0.7643	0.1450	0.6830	0.8794	0.9429
	Equi	0.2074	0.3701	0.6536	0.1176	0.8302	0.9245	0.9577
	Cube	0.2505	0.3929	0.6628	0.1281	0.7556	0.9135	0.9612
	Ours w/ fusion	**0.2048**	**0.3470**	**0.6259**	**0.1134**	**0.8452**	**0.9319**	**0.9632**
Stanford2D3D	FCRN [16]	0.1837	0.3428	0.5774	0.1100	0.7230	0.9207	0.9731
	OmniDepth (bn) [38]	0.1996	0.3743	0.6152	0.1212	0.6877	0.8891	0.9578
	Equi	0.1428	0.2711	0.4637	0.0911	0.8261	0.9458	0.9800
	Cube	0.1332	0.2588	0.4407	0.0844	0.8347	0.9523	0.9838
	Ours w/ fusion	**0.1209**	**0.2343**	**0.4142**	**0.0787**	**0.8660**	**0.9580**	**0.9860**

Table 2. Quantitative results on virtual-world datasets: PanoSUNCG and 360D.

Dataset	Method	MRE ↓	MAE ↓	RMSE ↓	RMSE (log) ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
PanoSUNCG	FCRN [16]	0.0979	0.1346	0.3973	0.0692	0.9223	0.9659	0.9819
	OmniDepth [38]	0.1143	0.1624	0.3710	0.0882	0.8705	0.9365	0.9650
	Equi	0.0687	0.0836	0.2902	0.0496	0.9529	0.9787	0.9886
	Cube	0.0628	0.0891	0.2946	0.0508	0.9453	0.9780	0.9890
	Ours w/ fusion	**0.0592**	**0.0789**	**0.2596**	**0.0443**	**0.9590**	**0.9823**	**0.9907**
360D	FCRN [16]	0.0699	0.1381	0.2833	0.0473	0.9532	0.9905	0.9966
	OmniDepth [38]	0.0931	0.1706	0.3171	0.0725	0.9092	0.9702	0.9851
	Equi	**0.0606**	0.1172	0.2667	0.0437	0.9667	0.9920	0.9966
	Cube	0.0613	0.1167	0.2739	0.0447	0.9688	0.9908	0.9956
	Ours w/ fusion	0.0615	**0.1143**	**0.2440**	**0.0428**	**0.9699**	**0.9927**	**0.9969**

and we use one of official splits that takes fifth area (area_5) for testing, and the others are for training. During training and testing, we resize the resolution of image and depth map into 512 × 1024.

PanoSUNCG. PanoSUNCG contains 103 scenes of SUNCG [21] and has 25,000 panoramas. In experiments, we use the official training and testing splits, where 80 scenes are for training and 23 for testing. For all panoramas, we resize them to 256 × 512 and filter out pixels with depth values larger than 10 meters.

360D. 360D dataset is collected by OmniDepth [38], including two synthetic datasets, SunCG and SceneNet and two realistic datasets, Stanford2D3D and Matterport3D. They use path tracing renderer to render four datasets and place spherical cameras in the virtual environment to acquire photo-realistic panoramas with the resolutions 256 × 512. For each panorama, they apply augmentation by 90°, 180° and 270°. In total, 360D contains 35,977 panoramas, where 34,679 of them are used for training and the rests are for testing.

4.2. Overall Performance

We first present results of using two baselines, each with a single branch, and compare them with our proposed two-branch framework: 1) **Equi**: the equirectangular branch B_e without bi-projection fusion; 2) **Cube**: the cubemap branch B_c with cube padding [4] without our fusion scheme; 3) **Ours w/ fusion**: our final model of applying the proposed spherical padding to the cubemap branch B_c and integrating our bi-projection fusion to both branches.

In Table 1 and 2, we show quantitative comparisons on four datasets as mentioned above. Overall, our fusion model performs favorably against FCRN [16] and OmniDepth [38], as well as our baselines using the single branch (i.e., Equi or Cube). This validates the effectiveness of the proposed two-branch network, in which the equirectangular view provides a larger field-of-view and the cubemap one focuses on non-distorted regions.

Moreover, on Matterport3D and Stanford2D3D, we find that the official implementation of OmniDepth (originally designed for the 360D [38] dataset) has difficulty to converge on these two datasets, and thus we add batch normalization [13] to successfully train the model, which is denoted as **OmniDepth (bn)** in Table 1.

Qualitative Comparisons. From Fig. 6 to 9, we present qualitative results of depth maps on four datasets. Compared to the FCRN and OmniDepth methods, our model is

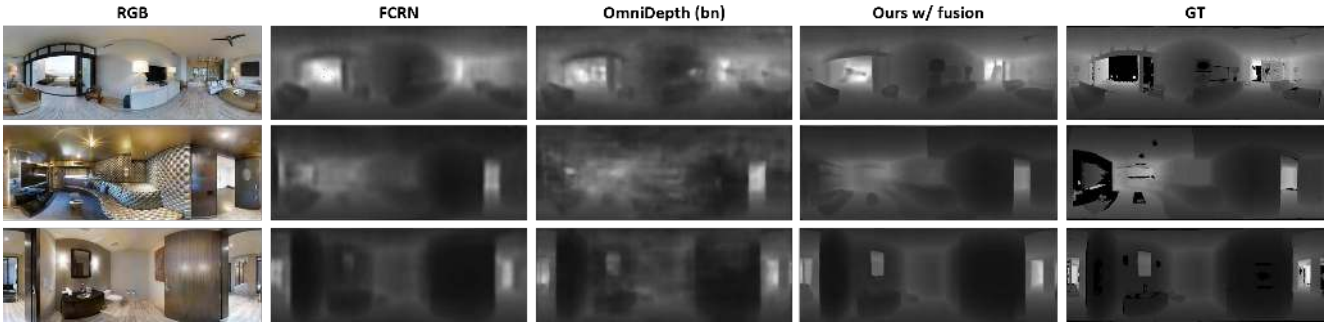

Figure 6. Qualitative results of Matterport3D. The black area in the ground truth depth map indicates invalid pixels.

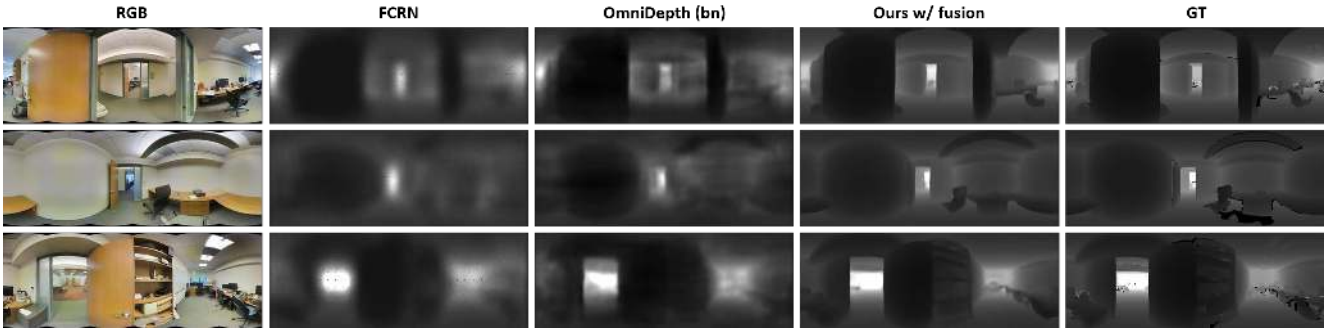

Figure 7. Qualitative results of Stanford2D3D. The black area in the ground truth depth map indicates invalid pixels.

Table 3. Comparison of padding methods on the cubemap branch.

Dataset	Method	MRE	MAE	RMSE
Matterport3D	Cube w/ zp	0.2577	0.4136	0.6934
	Cube w/ cp	0.2505	0.3929	0.6628
	Cube w/ sp	**0.2254**	**0.3660**	**0.6327**
Stanford2D3D	Cube w/ zp	0.1457	0.2667	0.4511
	Cube w/ cp	0.1332	0.2588	0.4407
	Cube w/ sp	**0.1259**	**0.2388**	**0.4269**
PanoSUNCG	Cube w/ zp	0.1195	0.1367	0.3441
	Cube w/ cp	0.0628	0.0891	0.2946
	Cube w/ sp	**0.0600**	**0.0840**	**0.2874**
360D	Cube w/ zp	0.0761	0.1382	0.2819
	Cube w/ cp	0.0610	0.1163	0.2722
	Cube w/ sp	**0.0588**	**0.1145**	**0.2614**

able to produce sharper results around boundaries. This can be attributed by the foveal view capturing detailed information, while the peripheral view with larger FoV provides global context.

4.3. More Results and Ablation Study

Effect of Spherical Padding. To further study the effects of spherical padding in the cubemap, we compare the proposed spherical padding (SP) with the other two padding methods, i.e., zero padding (ZP) and cube padding (CP).

Quantitative results on only the cubemap branch are shown in Table 3. By applying our spherical padding, the cubemap branch B_c outperforms other padding methods significantly. In addition, Fig. 10 shows qualitative comparisons of applying different padding methods. When using zero padding, the depth maps of six faces have obvious boundary artifacts. After using cube padding, the boundary effect becomes more smooth, but it is still observable because the cube padding does not follow the geometric relationship. By applying the proposed spherical padding, we are able to maintain the boundary as spherical padding is calculated using the spherical projection.

Table 4. Qualitative results of fusion methods on Matterport3D.

Method	MRE	MAE	RMSE
Yang *et al*. [31]	0.2662	0.4842	0.7364
Average	0.2658	0.4405	0.7202
Ours	**0.2048**	**0.3470**	**0.6259**

Fusion Schemes. To validate our fusion module, we conduct two baselines on Matterport3D using the fusion method proposed in [31] via directly adding up two feature maps and the feature averaging scheme. We show this ablation study in Table 4. From the results, our method is consistently better than the baselines. For instance, the MAE is improved by 28% and 23% comparing with Yang *et al*. [31] and Average. In addition, we find the training of this baseline is unstable and has the convergence problem as the

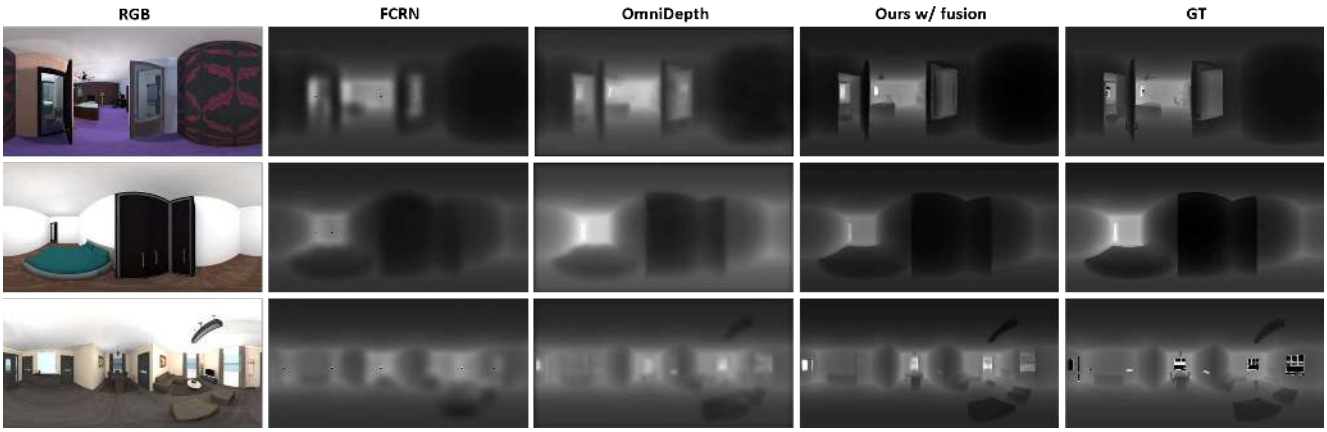

Figure 8. Qualitative results of PanoSUNCG. The black area in the ground truth depth map indicates invalid pixels.

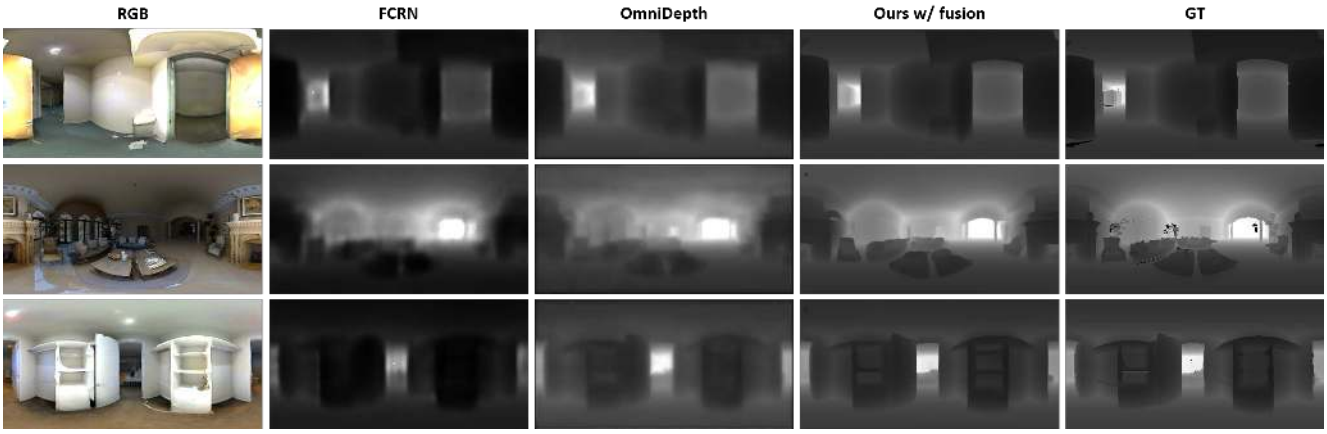

Figure 9. Qualitative results of 360D. The black area in the ground truth depth map indicates invalid pixels.

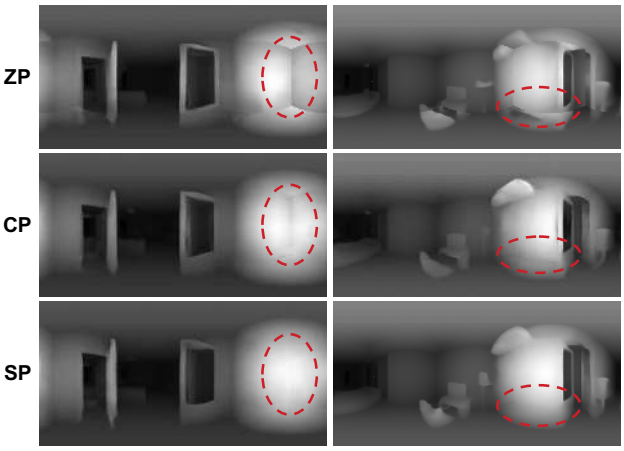

Figure 10. Qualitative result of different padding methods. For clear visualization, we plot the inverse depth to compare different padding methods.

gradients from different branches cannot be well balanced. This shows the benefit of integrating our bi-projection fusion scheme which applies several masks to balance features of two branches.

5. Conclusions

In this paper, we propose an end-to-end 360° depth estimation network which incorporates both equirectangular and cubemap projections to mimic peripheral and foveal vision as the human eye. Since the two projections have the complementary property, we fuse their features by our bi-projection fusion module. Furthermore, to extend the field-of-view of the cubemap projection and eliminate the boundary inconsistency of each cube face, we propose spherical padding which connects features from neighboring faces. Experimental results demonstrate that our method achieves state-of-the-art performance.

Acknowledgements. This project is funded by Ministry of Science and Technology of Taiwan (MOST-109-2636-E-009-018, MOST 109-2634-F-007-016, MOST 108-2634-F-007-006, MOST Joint Research Center for AI Technology and All Vista Healthcare, and Taiwan Computing Cloud) .

References

[1] Iro Armeni, Sasha Sax, Amir Roshan Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *CoRR*, 2017. 2, 5

[2] Y. Cao, Z. Wu, and C. Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks, 2018. 2

[3] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from rgb-d data in indoor environments. In *International Conference on 3D Vision (3DV)*, 2017. 2, 5

[4] Hsien-Tzu Cheng, Chun-Hung Chao, Jin-Dong Dong, Hao-Kai Wen, Tyng-Luh Liu, and Min Sun. Cube padding for weakly-supervised saliency prediction in 360° videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3, 6

[5] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang. Segflow: Joint learning for video object segmentation and optical flow. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 5

[6] Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations (ICLR)*, 2018. 3

[7] M. Eder, P. Moulon, and L. Guan. Pano popups: Indoor 3d reconstruction with a plane-aware network. In *International Conference on 3D Vision (3DV)*, 2019. 3

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*. 2014. 2

[9] Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning so(3) equivariant representations with spherical cnns. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[10] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[11] Clement Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 5

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. 6

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014. 5

[15] Hsueh-Ying Lai, Yi-Hsuan Tsai, and Wei-Chen Chiu. Bridging stereo matching and optical flow via spatiotemporal correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[16] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *International Conference on 3D Vision (3DV)*, 2016. 1, 2, 4, 5, 6

[17] Jae-Han Lee, Minhyeok Heo, Kyung-Rae Kim, and Chang-Su Kim. Single-image depth estimation based on fourier domain analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[18] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[19] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 5

[20] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2009. 2

[21] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[22] Yu-Chuan Su and Kristen Grauman. Kernel transformer networks for compact spherical convolution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[23] Yu-Chuan Su and Kristen Grauman. Learning spherical convolution for fast features from 360°imagery. In *Advances in Neural Information Processing Systems (NIPS)*. 2017. 3, 4

[24] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 5

[25] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[26] Fu-En Wang, Hou-Ning Hu, Hsien-Tzu Cheng, Juan-Ting Lin, Shang-Ta Yang, Meng-Li Shih, Hung-Kuo Chu, and Min Sun. Self-supervised learning of depth and camera motion from 360° videos. In *Asian Conference on Computer Vision (ACCV)*, 2018. 2, 3, 5

[27] Ning-Hsu Wang, Bolivar Solarte, Yi-Hsuan Tsai, Wei-Chen Chiu, and Min Sun. 360sd-net: 360° stereo depth estimation with learnable cost volume. *arXiv:1911.04460*, 2019. 2

[28] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L. Yuille. Towards unified depth and semantic prediction from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[29] Dan Xu, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Multi-scale continuous crfs as sequential deep

networks for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[30] Dan Xu, Wei Wang, Hao Tang, Hong Liu, Nicu Sebe, and Elisa Ricci. Structured attention guided convolutional neural fields for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[31] Shang-Ta Yang, Fu-En Wang, Chi-Han Peng, Peter Wonka, Min Sun, and Hung-Kuo Chu. Dula-net: A dual-projection network for estimating room layouts from a single rgb panorama. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 5, 7

[32] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry from videos with edge-aware depth-normal consistency. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2

[33] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[34] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[35] Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao. Panocontext: A whole-room 3d context model for panoramic scene understanding. In *European Conference on Computer Vision (ECCV)*, 2014. 2

[36] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang. Joint task-recursive learning for semantic segmentation and depth estimation. In *European Conference on Computer Vision (ECCV)*, 2018. 5

[37] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[38] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. Omnidepth: Dense depth estimation for indoors spherical panoramas. In *European Conference on Computer Vision (ECCV)*, 2018. 2, 3, 4, 5, 6

[39] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2