SphereFusion: Efficient Panorama Depth Estimation via Gated Fusion

Qingsong Yan[1], Qiang Wang [2,*], Kaiyong Zhao [3], Jie Chen [4],
Bo Li [5], Xiaoweo Chu [6], Fei Deng [1]
[1]Wuhan University, Wuhan, China [2]HIT (Shenzhen), Shenzhen, China [3]XGRIDS, Shenzhen, China
[4]HKBU, Hong Kong, China [5]HKUST, Hong Kong, China [6]HKUST (Guangzhou), Guangzhou, China
yanqs_whu@whu.edu.cn, qiang.wang@hit.edu.cn, kyzhao@xgrids.com, chenjie@comp.hkbu.edu.cn
bli@cse.ust.hk, xwchu@ust.hk, fdeng@sgg.whu.edu.cn

Abstract

Due to the rapid development of panorama cameras, the task of estimating panorama depth has attracted significant attention from the computer vision community, especially in applications such as robot sensing and autonomous driving. However, existing methods relying on different projection formats often encounter challenges, either struggling with distortion and discontinuity in the case of equirectangular, cubemap, and tangent projections, or experiencing a loss of texture details with the spherical projection. To tackle these concerns, we present SphereFusion, an end-to-end framework that combines the strengths of various projection methods. Specifically, SphereFusion initially employs 2D image convolution and mesh operations to extract two distinct types of features from the panorama image in both equirectangular and spherical projection domains. These features are then projected onto the spherical domain, where a gate fusion module selects the most reliable features for fusion. Finally, SphereFusion estimates panorama depth within the spherical domain. Meanwhile, SphereFusion employs a cache strategy to improve the efficiency of mesh operation. Extensive experiments on three public panorama datasets demonstrate that SphereFusion achieves competitive results with other state-of-the-art methods, while presenting the fastest inference speed at only 17 ms on a 512×1024 panorama image.

1. Introduction

Depth estimation is an important task in computer vision that helps to understand the 3D environment. In particular, the panorama image has a 360° field of view (FOV) and can reconstruct the entire surrounding environment in one shot [48, 59]. With the development of consumer-level panorama cameras, such as Ricoh Theta, Samsung Gear360, and Insta360 ONE, it becomes an intriguing task to estimate the depth map from the panorama image [34, 37, 42, 46, 51], and many related datasets have been generated to facilitate the research of panorama depth estimation [1, 3, 6, 57, 59].

[*]Corresponding author

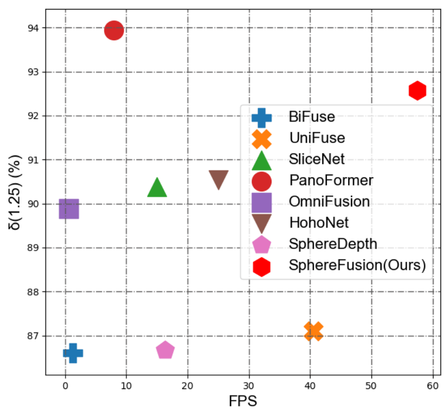


Figure 1. Comparison with BiFuse [48], UniFuse [29], SliceNet [37], PanoFormer [42], OminiFusion [34], SphereDepth [51], HohoNet [46] on Stanford2D3D [3] with resolution of 512 × 1024. The horizontal axis is the FPS, and the vertical axis is $\delta(1.25)(\%)$, which counts the percentage of the absolute relative difference between the prediction and the ground truth that is less than 1.25. The higher FPS and higher $\delta(1.25)(\%)$ mean better.

However, it is challenging to find a proper way to represent the panorama image. The most popular equirectangular projection faces huge distortion around the poles and poor discontinuity near the borders. The cubemap projection [44] and the tangent projection [17, 34, 42] project the panorama image to several planes to avoid distortion but introduce serious discontinuity problems and have to rely on a well-designed fusion strategy to merge them. The spherical projection [51] can deal with distortion and discontinuity by approximating the sphere, but it is hard to capture details of the panorama image and cannot directly handle high-resolution panorama images.

Based on the characteristics of these projections, different strategies are used to extract features from panorama images. The equirectangular projection represents the panorama image in one image plane and can directly utilize 2D image convolution to extract features. Nevertheless, the equirectangular projection faces huge distortion and needs specially designed 2D convolution kernels to extract reliable features [7, 47, 59]. Like the equirectangular projection, the cubemap projection and tangent projection employ multiple planes to represent the panorama image and use 2D image convolution to extract features. However, they require a suitable mechanism to fuse features from different planes and maintain global consistency [8, 34, 36, 42]. [20, 23, 27, 51]. On the other hand, the spherical projection is an ideal way to represent the panorama image, and utilizes mesh operation to extract features. Nonetheless, the mesh operation struggles to capture details and exhibits inefficiency [20, 23, 27, 51]. Meanwhile, BiFuse [48] and UniFuse [29] try to combine the strengths of different projections. Although they achieve better depth estimation results, they still suffer from distortion around the poles.

To this end, this paper proposes SphereFusion, which unites equirectangular and spherical projections. SphereFusion first represents the panorama image by equirectangular and spherical projections, then extracts two types of features, and finally fuses them through a fusion module to estimate the panorama depth in the spherical projection. To balance the accuracy and efficiency, SphereFusion implements a lightweight encoder to extract features [24] and utilizes a cache strategy to reduce the computation complexity of the mesh operation. Fig. 1 compares the efficiency and quality of the depth map with state-of-the-art methods [29, 34, 37, 42, 46, 48, 51]. SphereFusion obtains high-quality depth maps and achieves around 60 FPS during inference on the panorama image with a resolution of 512×1024. Our contributions are summarized as follows.

1. We propose a panorama depth estimation method, SphereFusion, to estimate the depth map in the spherical projection, which uses the features from the equirectangular projection to improve the details.
2. We design a feature fusion module, GateFuse, which selects reliable features from two projections to improve the quality of the depth map.
3. Experiments demonstrate that SphereFusion achieves competitive results on three public panorama datasets and can produce point clouds with less noise and higher completeness. Besides, SphereFusion achieves 60 FPS of inference speed on an NVIDIA RTX 3090, outperforming existing methods.

2. Related Work

2.1. Perspective Depth Estimation

Over the past few decades, the perspective 2D image from the classical pinhole camera model has attracted much interest in inferring depth estimation for 3D sensing applications. The traditional methods, such as Make3D [41] and Pop-up [25], leverage the prior depth distribution and regress the pixel-wise depth via a Markov random field (MRF). Due to the success of deep learning, researchers have also utilized its capability of multi-level feature extraction to improve the depth quality. As the first trial, Eigen [19] and Laina [32] design end-to-end convolutional neural networks to regress the depth map from an RGB image. Furthermore, several recent studies [4, 54] take advantage of the visual transformer (ViT) [14], which captures the relationship between each pixel patch to regress the depth map with a global context. To improve the model generalization and depth consistency, Midas [38] focuses on the training strategy of mixing different datasets and tuning the loss function. Other studies, such as [2, 18, 35, 52], introduce some constraints like the surface normal or the semantic information to co-optimize the depth map. However, a key challenge in training models for depth estimation is the lack of high-quality depth labels, which motivates the studies of unsupervised learning [5, 22, 50, 56, 58]. SfMLearner [58] leverages photometric consistency to estimate the relative pose and depth simultaneously. MonoDepth2 [22] filters out the outliers that violate photometric consistency to deal with occlusions. In addition, SfMLearner-SC [5] explores the depth consistency between video frames to restrict the depth of each pixel. Although depth estimation for 2D perspective images has achieved considerable performance, they cannot be directly applied to the panorama image. The main reason is that the panorama image needs a particular type of projection, which usually causes distortion or discontinuity, where network for perspective vision are hard to capture reliable features.

2.2. Panorama Depth Estimation

Unlike the perspective image, which only provides a limited FOV, the panorama image has a 360° FOV. Compared to the development of perspective depth estimation, panorama depth estimation is still in its infancy stage. Most of the recent studies [31, 42, 43, 46, 48, 59] rely on special projection methods to transform the panorama image to the 2D perspective image and estimate the depth using the perspective neural network structures. The equirectangular projection is the most common projection, allowing all the surrounding information to be observed from a single 2D image. OmniDepth [59] proposes the first end-to-end network based on the equirectangular projection for panorama depth estimation along with a large synthetic dataset. SliceNet [37] uses sliced feature maps to estimate the depth map through LSTM. The cubemap projection transforms the panorama image into six perspective images to estimate depth maps and then fuse them through a specially designed module [8, 48]. The tangent projection approximates the

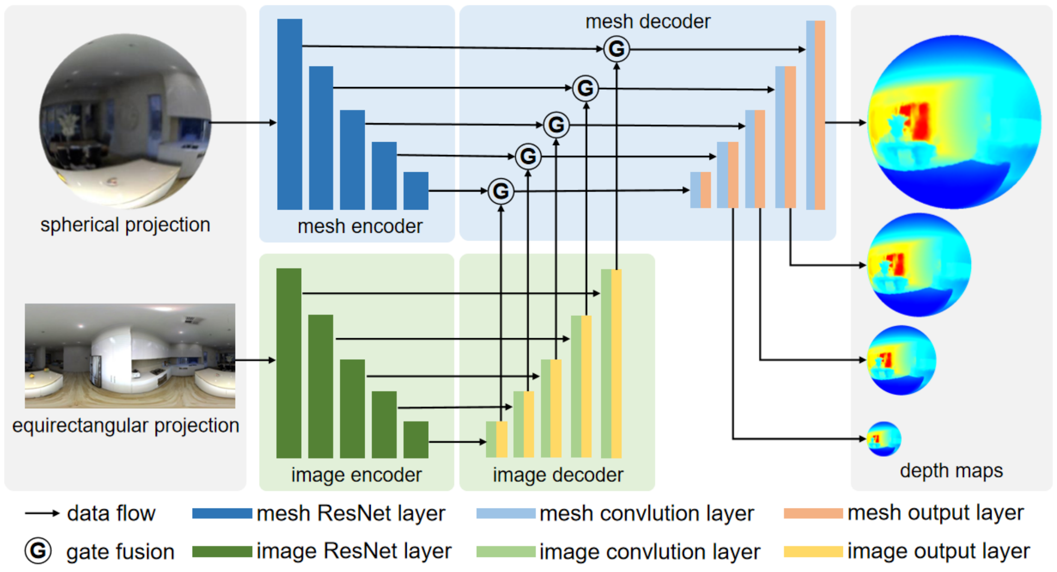


Figure 2. Given a panorama image in the equirectangular projection and the spherical projection, SphereFusion simultaneously extracts features by a 2D image encoder and a mesh encoder, which follows the ResNet structure [24], then fuses these features by the Gate Fusion module in the spherical projection, and finally estimates the depth map through the mesh decoder.

sphere through more perspective images. OmniFusion [34], PanoFormer [42], and 360MonoDepth [39] samples perspective images from the spherical surface [17] and applies the existing convolutional or ViT models on each tangent patch. Despite the success of these methods, the main problem of those projections is introducing distortion and discontinuity in certain areas of the panorama image. Although several subsequent studies have attempted to improve the depth quality of these regions, such as designing special convolutional kernels [9, 13, 16, 45, 47, 49, 53], applying the deformable convolution [7, 21], and multi-task learning [15, 30, 55], we argue that the influence of distortion and discontinuity cannot be removed entirely. In addition to changing the convolution kernel, BiFuse [48] and UniFuse [29] combine the equirectangular projection and the cubemap projection.

To eliminate the drawbacks of the above two projection methods, some recent studies attempt to process the panorama image in the spherical domain. SpherePHD [33] uses the icosahedral spherical mesh to represent the panorama image and extract semantic maps. SphereDepth [51] modifies the mesh operation from the SubdivNet [27], which is more efficient than MeshCNN [23] and MeshNet [20]. In addition, S2CNN [12] uses the spheric harmonics function to build networks but is unsuitable for dense estimation. US2CNN [28] builds grids and manually assigns weight to build the network.

3. Method

In this section, we describe the details of SphereFusion, as shown in Fig. 2. We first describe how to represent the panorama image in Section 3.1 and extract features in the spherical projection in Section 3.2. Then, we show the pipeline of our SphereFusion in Section 3.3. Finally, we present the loss function in Section 3.4.

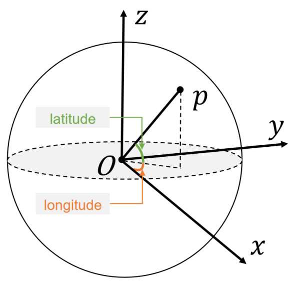
3.1. Panorama Representation

Finding a suitable way to represent the panorama image is the key to high-quality panorama depth estimation. The equirectangular projection [59] suffers from distortion on the poles and discontinuity at the borders, but it can directly use 2D convolution to extract features from images. The cubemap and tangent projection have small distortion but need a special mechanism to fuse different patches [34, 36, 42]. The spherical projection [51] is an ideal way to represent the panorama image, but mesh convolution is hard to extract texture features compared with 2D image convolution. In light of these, we simultaneously employ the equirectangular and spherical projections.

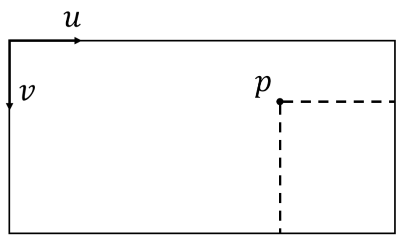
The equirectangular projection uses a 2D image plane to represent the panorama image, as Fig. 3(b) shows, where the image plane is built on the latitude and longitude of the sphere surface. Given a pixel $p = (u, v)$ on the image plane, we can calculate its position on the sphere surface by Eq. 1, where W is the image width and H is the image height.

$$\begin{cases} longitude & = (2u/W - 1) \times \pi \\ latitude & = (v/H - 0.5) \times \pi \end{cases} \tag{1}$$

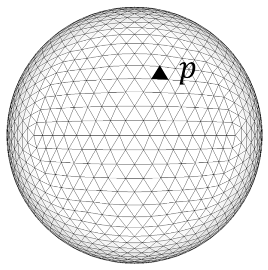
The spherical projection is based on the icosahedron spherical mesh [17], which can approximate the sphere by a higher mesh resolution (MR), where MR represents the



(a) Panorama Image

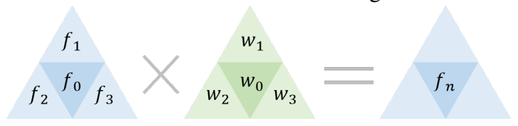


(b) Equirectangular Projection

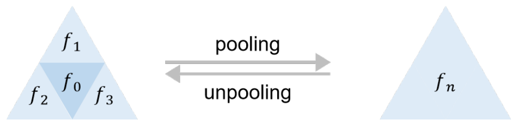


(c) Spherical Projection

Figure 3. The ideal representation of a panorama image is the sphere, but it is impractical. The equirectangular projection is the most popular method, but it suffers from distortion at the poles and discontinuity at the borders. The spherical mesh can approximate the sphere, and their difference becomes smaller with higher MR.



(a) Mesh Convolution



(b) Mesh Pooling

Figure 4. Mesh Operations includes Mesh Convolution and the Mesh Pooling/Unpooling [51], which relies on the relationship between triangles of the spherical mesh.

times of loop subdivision [27] is applied on the icosahedron spherical mesh and determines the number of triangles in the spherical mesh by 20×4^{MR}. One triangle in the spherical mesh represents one pixel in equirectangular projection, as Fig. 3(c) shows, Tangent [17] points out that a panorama image in the equirectangular projection with higher image resolution needs a spherical mesh with higher MR. In our implementation, we use the triangle center (x, y, z) to represent the whole triangle and can calculate its position on the sphere surface by Eq. 2.

$$\begin{cases} longitude & = atan(y, x) \\ latitude & = atan(z, \sqrt{x^2 + y^2}) \end{cases} \tag{2}$$

According to Eq. 1 and Eq. 2, we define the E2S (equirectangular projection to spherical projection) and S2E (spherical projection to equirectangular projection).

3.2. Mesh Operations

As we utilize the spherical projection to represent the panorama image, we need mesh convolution and the mesh pooling/unpooling to extract features, which is inspired by SubdivNet [27] and SphereDepth [51].

Mesh convolution relies on the FAF (face adjacent face), which describes the topological relationship between triangles of the spherical mesh, as Fig. 4(a) shows. Each triangle in the mesh has three neighbors and can extract features by linear interpolation by Eq. 3, where $w_i (i = 0, 1, 2, 3)$ are the weight parameters, b_0 is the bias parameter, f_0 is the feature of the center triangle, f_1, f_2, f_3 are the feature of adjacent triangles, f_n is the extracted features.

$$f_n = \sum_{i=0}^{3} w_i f_i + b_0 \tag{3}$$

The mesh pooling and unpooling are fundamental components of constructing an encoder-decoder structure. Similar to the image pooling, mesh pooling merges four triangles into one, as shown in Fig. 4(b) and calculates the feature of the new f_n triangle through features of f_0, f_1,f_2, and f_3 by the mesh max pooling. The mesh unpooling is the opposite of the mesh pooling, which splits one triangle into four triangles by loop subdivision.

3.3. Our Framework

3.3.1 The Network Encoder

Given a panorama image, we represent it by the spherical and equirectangular projection and employ the mesh encoder and image encoder to extract features, respectively. The network structure of the mesh encoder follows the ResNet [24]. Considering the extremely high computational complexity of mesh operations [27, 51], the mesh encoder uses the simplest ResNet18, and generates five scales of spherical features F_{sp}, of which the channels are 64, 64, 128, 256, and 512. SphereFusion randomly initializes the mesh encoder and trains it from scratch. The image encoder directly uses the ResNet50 to extract image features F_{eq}, of which the channels are 64, 256, 512, 1024, and 2048. As 2D image convolution can not capture reliable features on distorted regions [59], we add a lightweight image encoder to extend the receptive field. Meanwhile, the image decoder aligns the number of channels of F_{eq} to F_{sp} to facilitate the subsequent feature fusion. SphereFusion initializes the image encoder by a pre-trained weight and randomly initializes the image decoder.

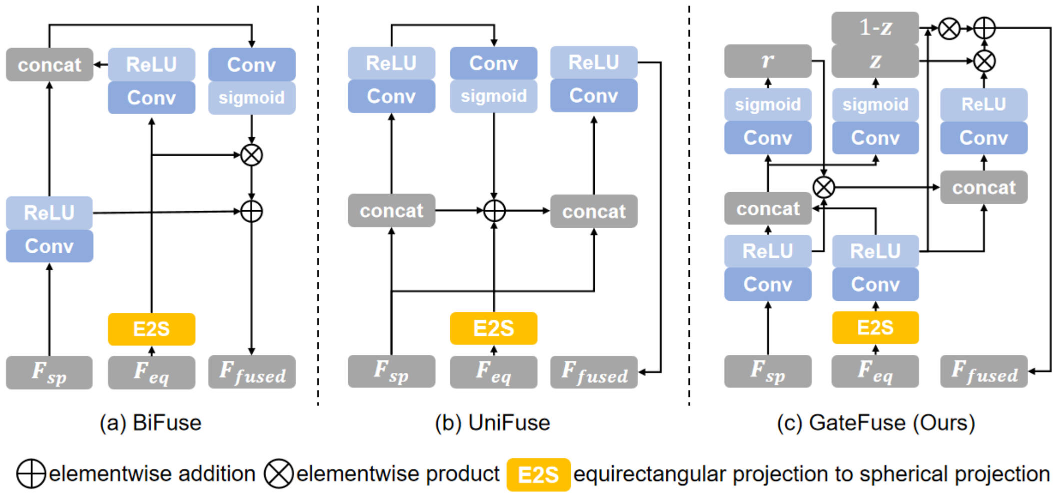


Figure 5. We implement BiFuse [48], UniFuse [29], and our GateFuse to fuse features from spherical projection F_{sp} and equirectangular projection F_{eq}. Unlike BiFuse and UniFuse select features from F_{eq} and fuse them to F_{sp}, GateFuse selects features from F_{sp} and F_{eq}.

3.3.2 The Fusion Module

After extracting features from the spherical projection F_{sp} and the equirectangular projection F_{eq}, SphereFusion uses the GateFuse module to fuse these features in the spherical domain. The core idea of the GateFuse module is to enhance the F_{sp} by F_{eq} through a reset gate and a forget gate inspired by GRU [10]. For extracted features in each scale, GateFuse first transforms F_{eq} to the spherical projection through the E2S module, then concatenates these features to estimate a reset gate value r and a forget gate value z, where r selects features from F_{sp} and z selects features from F_{eq}. Finally, GateFuse adds these features to get fused features F_{fused}. Compared with BiFuse [48] and UniFuse [29], which use concatenate features to estimate a mask to select reliable features from F_{eq}, GateFuse simultaneously selects reliable features from F_{sp} and F_{eq}, instead of simply trusting one type of features. To better compare different fusion modules, we implement BiFuse, UniFuse, and our GateFuse and visualize these modules in Fig. 5.

3.3.3 The Network Decoder

With fused features, we construct a mesh decoder to estimate the panorama depth. Following the network structure of UNet [40], SphereFusion uses skip-connection to concatenate fused features F_{fused} with features from the mesh encoder F_{sp}, and uses the mesh unpooling to reconstruct a high-resolution panorama depth map. Specifically, the mesh decoder has six layers, and the number of channels in each layer are 1024, 512, 64, 32, 32, and 32, respectively. Meanwhile, each layer contains one mesh unpooling layer to gradually increase the MR of the spherical mesh from 2 to 7 for 360D and 3 to 8 for Matterport3D and Stanford2D3D. The mesh decoder outputs multi-resolution panorama depth maps to speed up the training procedure.

3.3.4 Inference Efficiency

Unlike 2D image convolution, which can directly find adjacent pixels by coordinate, the mesh operation needs to compute the FAF of each triangle to identify nearby triangles in each layer, which will become more complex with higher MR. However, as the mesh pooling/unpooling changes the spherical mesh, mesh convolution layers between two mesh pooling/unpooling layers use a spherical mesh with the same MR. Meanwhile, the mesh decoder and the mesh encoder at the same scale use a spherical mesh with the same MR. Based on these findings, SphereFusion first identifies all active spherical meshes, then merges these spherical meshes with the same MR, and then calculates the corresponding FAF once. During training and testing, SphereFusion stores these connectivity information in cache memory without recalculating FAF for each layer, which can significantly improve inference efficiency.

3.4. Loss Function

Following recent works [37, 48], we use the BerHu loss [32] during training as Eq. 4 shows, where y is the ground truth depth and $\hat{y}$ is the predicted depth, and the threshold T is set to 0.2 in all our experiments.

$$\mathcal{L}(y,\hat{y}) = \begin{cases} |y-\hat{y}|, & |y-\hat{y}| < T \\ \frac{(y-\hat{y})^2+T^2}{2T}, & |y-\hat{y}| \geq T \end{cases} \tag{4}$$

To accelerate the training, SphereFusion predicts depth maps with multiple scales and extendsEq. 4 to multi-scale, as Eq. 5 shows, where s is the scale, s_i is the weight, V_i is the valid pixel, N_i is the number of the valid pixel.

$$Loss = \sum_{i<s} s_i \frac{\sum_{p\in V_i} \mathcal{L}(y,\hat{y})}{N_i} \tag{5}$$

4. Experiments

4.1. Datasets

360D [59] is a synthetic panorama dataset and contains 35977 panorama images with a resolution of 256×512.
Matterport3D [6] is a large real-world dataset, which has 10800 panorama images. We resize all panorama images and depth maps into 512×1024 during training and testing.
Stanford2D3D [3] is a real-world indoor dataset that contains 1413 panoramas. We resize all panorama images and depth maps into 512×1024.

4.2. Implementation

We implement our method by Jittor [26]. On 360D, we train the network with 30 epochs, setting the batch size to 4 and the initial learning rate to 0.0002. On Matterport3D and Stanford2D3D, we train the network with 30 epochs, setting the batch size to 2 and the initial learning rate to 0.0001. On all datasets, we train on one Nvidia Tesla V100 and halve the learning rate after every ten epochs. Since Matterport3D and Stanford2D3D utilize the same sensors to capture panorama images, and the size of Stanford2D3D is relatively small, we combine their training data but test each dataset separately.

4.3. Quantitative Evaluation

During the quantitative evaluation, we follow common evaluation metrics [48] and use MAE, MRE, RMSE, RMSE (log), and δ and compare with state-of-the-art panorama depth estimation methods [29, 34, 37, 42, 46, 48, 51]. To deal with outliers, we ignore pixels whose depth is outside of the range $0.1 \sim 10$ meters for 360D [59] and $0.1 \sim 16$ for Stanford2D3D [3] and Matterport3D [6]. Table 1 shows evaluation results on three datasets, including the quality of the depth map and the inference efficiency.

On 360D [59], our method achieves the best results on MRE and MAE and ranks second on RMSE(log), δ_1, δ_2, and δ_3. On Matterport3D [6], our method ranks second on MRE, MAE, and RMSE(log). As for Stanford2D3D [3], our method achieves the lowest MRE, MRE, and RMSE(log) values while ranking second in RMSE and δ_1. Overall, our method achieves competitive performance with the state-of-the-art methods. Notably, compared to SphereDepth [51], which only utilizes the mesh operation, SphereFusion significantly improves the quality of the depth map by the fusion strategy. Moreover, SphereFusion achieves comparable results with only a simple ResNet structure compared to OmniFusion [34] and PanoFormer [42], which use the ViT as the encoder, demonstrating the importance of choosing the proper projection.

In addition to the quality of panorama depth maps, we also compare the inference efficiency of different methods. We average the inference time by predicting the depth map

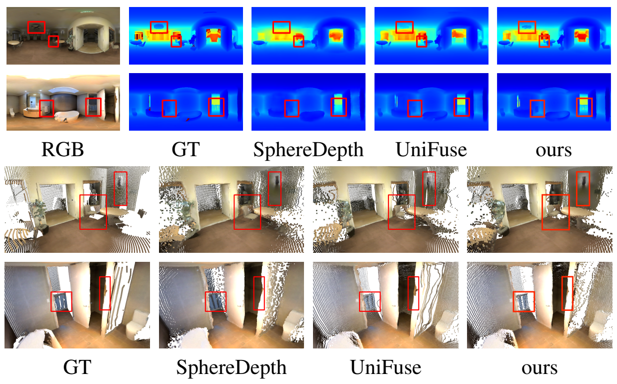


Figure 6. We select two scenes from 360D [59] and visualize depth maps and point clouds.

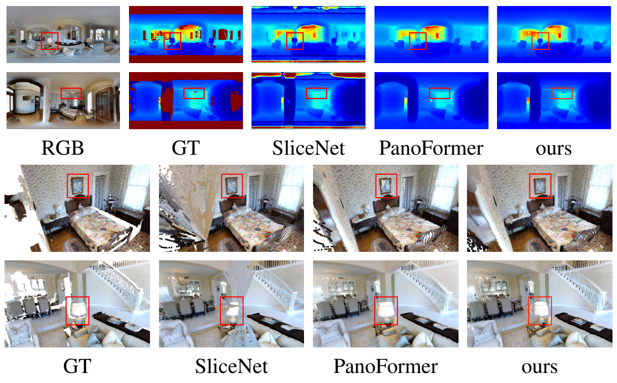


Figure 7. We select two scenes from Matterport3D [6] and estimate depth maps by SliceNet [37], PanoFormer [42], and our method SphereFusion.

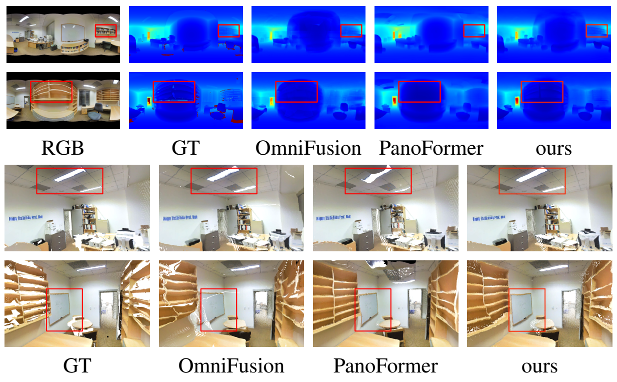


Figure 8. We select two scenes from Stanford2D3D [3] and show depth maps and corresponding point clouds.

of 100 panorama images with a resolution of 512×1024 on a single RTX 3090 to obtain a reliable inference time. SphereFusion is the most efficient method, requiring only 0.0174 seconds per image during inference. Compared with SphereDepth [51], our method achieves higher efficiency by using a lighter mesh network and the cache strategy to store the FAF during inference. Although OmniFusion [34] and PanoFormer [42] achieve higher depth map quality, they require longer inference time.

In summary, SphereFusion benefits from choosing the suitable projection and can obtain comparable reconstruc-

Table 1. Quantitative evaluation results of depth maps, where ‘S2D3D’ is the short for Standard2D3D and ‘M3D’ is the short for Matterport3D. ‘—’ means no data is available from the original paper. We mark out top three methods for better comparison.

Dataset	Method	MRE↓	MAE↓	RMSE↓	RMSE(log)↓	δ_1 ↑	δ_2 ↑	δ_3 ↑	Time(s)↓
S2D3D	FCRN [32]	0.1837	0.3428	0.5774	0.1100	0.7230	0.9207	0.9731	—
	OmniDepth [59]	0.1996	0.3743	0.6152	0.1212	0.6877	0.8891	0.9578	—
	BiFuse [48]	0.1209	0.2343	0.4142	0.0787	0.8660	0.9580	0.9860	0.7825
	SliceNet* [37]	**0.0998**^{3}	**0.1737**^{2}	0.3728	0.0765	0.9038	0.9623	0.9843	0.0668
	UniFuse [29]	0.1114	0.2082	0.3691	**0.0721**^{3}	0.8711	0.9664	0.9882	**0.0247**^{2}
	HohoNet [46]	0.1014	**0.2027**^{3}	0.3834	**0.0668**^{2}	**0.9054**^{3}	0.9693	0.9771	**0.0400**^{3}
	PanoFormer [42]	—	—	**0.3083**^{1}	—	**0.9394**^{1}	**0.9838**^{1}	**0.9941**^{1}	0.1253
	OmniFusion [34]	**0.0950**^{2}	—	**0.3474**^{3}	0.1599	0.8988	**0.9769**^{2}	**0.9924**^{2}	1.5885
	SphereDepth [51]	0.1158	0.2323	0.4512	0.0754	0.8666	0.9642	0.9863	0.0612
	SphereFusion	**0.0899**^{1}	**0.1654**^{1}	**0.3194**^{2}	**0.0611**^{1}	**0.9257**^{2}	**0.9755**^{3}	**0.9904**^{3}	**0.0174**^{1}
M3D	FCRN [32]	0.2409	0.4008	0.6704	0.1244	0.7703	0.9174	0.9617	—
	OmniDepth [59]	0.2901	0.4838	0.7643	0.1450	0.6830	0.8794	0.9429	—
	BiFuse [48]	0.2048	0.3470	0.6259	0.1134	0.8452	0.9319	0.9632	0.7825
	SliceNet [37]	0.1764	0.3296	0.6133	0.1045	0.8716	0.9483	0.9716	0.0668
	UniFuse [29]	**0.1063**^{2}	**0.2814**	0.4941	**0.0701**	**0.8897**^{3}	**0.9623**^{3}	0.9831	**0.0247**^{2}
	HohoNet [46]	0.1488	**0.2862**^{3}	0.5138	0.0871	0.8786	0.9519	0.9771	**0.0400**^{3}
	PanoFormer [42]	—	—	**0.3635**^{1}	—	**0.9184**^{2}	**0.9804**^{1}	**0.9916**^{2}	0.1253
	OmniFusion [34]	**0.0900**^{1}	—	**0.4261**^{2}	0.1483	**0.9189**^{1}	**0.9797**^{2}	**0.9931**^{1}	1.5885
	SphereDepth [51]	0.1205	0.3311	0.5922	**0.0806**^{3}	0.8620	0.9519	0.9770	0.0612
	SphereFusion	**0.1145**^{3}	**0.2852**^{2}	**0.4885**^{3}	**0.0733**^{2}	0.8701	0.9613	**0.9838**^{3}	**0.0174**^{1}
360D	FCRN [32]	0.0699	0.1381	0.2833	0.0473	0.9532	0.9905	0.9966	—
	OmniDepth [59]	0.0931	0.1706	0.3171	0.0725	0.9092	0.9702	0.9851	—
	BiFuse [48]	0.0615	0.1143	0.2440	0.0428	0.9699	0.9927	0.9969	0.6971
	SliceNet [37]	0.0467	**0.1134**^{3}	**0.1323**^{1}	**0.0212**^{1}	0.9788	0.9952	0.9969	0.0595
	UniFuse [29]	**0.0466**^{3}	**0.0996**^{2}	0.1968	**0.0315**^{3}	0.9835	0.9965	0.9987	**0.0221**^{2}
	PanoFormer [42]	—	—	**0.1429**^{2}	—	**0.9876**^{1}	**0.9975**^{1}	**0.9991**^{1}	0.1116
	OmniFusion [34]	**0.0430**^{2}	—	**0.1808**^{3}	0.0735	**0.9859**^{3}	**0.9969**^{3}	**0.9989**^{3}	1.4151
	SphereDepth [51]	0.0550	0.1145	0.2364	0.0369	0.9743	0.9944	0.9978	**0.0545**^{3}
	SphereFusion	**0.0417**^{1}	**0.0894**^{1}	0.1813	**0.0286**^{2}	**0.9869**^{2}	**0.9970**^{2}	**0.9989**^{2}	**0.0155**^{1}

* We recalculate all metrics using open-source models.

tion results with more efficient inference efficiency using only a lightweight network model.

4.4. Qualitative Evaluation

For the qualitative evaluation, we visualize depth maps of different methods in Fig. 6, Fig. 7, and Fig. 8. Furthermore, we convert them to point clouds to compare different methods in the 3D space and visualize point clouds by Meshlab [11] with the same rendering settings.

On 360D [59], we compare our method with SphereDepth [51] and UniFuse [29], and visualize depth maps and corresponding point clouds in Fig. 6. UniFuse has better reconstruction results in the middle areas but struggles around the polar regions, such as the lights on the ceiling. SphereDepth reconstructs the ceiling region but suffers from losing details, such as edges of the door and the wall. SphereFusion combines the strengths of two projections and can reconstruct details and polar regions at the same time.

On Matterport3D [6], we compare our method with SliceNet [37] and PanoFormer [42], and Fig. 7 shows results. SliceNet suffers from poles, as it only uses the equirectangular projection and extracts features by the 2D image encoder. PanoFormer and SphereFusion achieve better results using the tangent and spherical projections.

On Stanford2D3D [3], we compare our method with OmniFusion [51] and PanoFormer [29]. Fig. 8 shows depth maps and corresponding point clouds. OmniFusion and PanoFormer use the tangent projection and attempt to reduce discontinuities using more complex feature fusion mechanisms. However, OmniFusion fails to merge different tangent patches and has noticeable gaps in the point cloud, while PanoFormer is smoother and loses some details. Al-

Table 2. Ablation studies on network encoder and different fusion strategies.

2D Encoder	Mesh Encoder	Fusion Module	MRE↓	MAE↓	RMSE↓	RMSE(log)↓	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_3\uparrow$
✓	×	×	0.0461	0.0969	0.2081	0.0315	0.9833	0.9964	0.9986
×	✓	×	0.0572	0.1180	0.2372	0.0374	0.9755	0.9956	0.9985
✓	✓	BiFuse [48]	**0.0415**	**0.0888**	0.1824	0.0288	0.9861	0.9969	0.9988
✓	✓	UniFuse [29]	0.0427	0.0915	0.1837	0.0290	0.9868	0.9969	0.9988
✓	✓	GateFuse (ours)	0.0417	0.0894	**0.1813**	**0.0286**	**0.9869**	**0.9970**	**0.9989**

though OmniFusion and PanoFormer achieve better reconstruction results than SphereFusion, our method only uses a simple encoder based on ResNet, demonstrating the importance of choosing the proper projection.

Overall, SphereFusion utilizes the spherical projection to avoid distortion and discontinuities and the equirectangular projection to extract visual features, achieving comparable results with state-of-the-art methods with a lighter network and higher inference efficiency.

4.5. Ablation Studies

We conduct several ablation studies to study the influence of different components of the SphereFusion. We first compare the network encoder to show the importance of using two encoders to extract features from the panorama image. We then study how to fuse features from the spherical and equirectangular projection. Throughout all ablation experiments, we use the 360D dataset.

4.5.1 Network Encoder

To evaluate the contribution of each encoder, we build two networks to estimate panorama depth, where each network only uses one type of encoder. Table 2 shows the performance of different network structures. The network that only uses the mesh encoder obtains the worst results, which cannot reconstruct details of the panorama image only through the mesh operation, as SphereDepth [51] does. The network that only uses the 2D image encoder ranks second in Table 2, which can achieve higher performance but cannot deal with distortion and discontinuity. SphereFusion outperforms others and achieves the best results, proving that combining the 2D image encoder and the mesh encoder can obtain higher-quality depth maps.

4.5.2 Fusion Strategy

The fusion strategy fuses features from different panorama projections. To compare different fusion strategies, we implement BiFuse [48], UniFuse [29], and our GateFuse to fuse features in the spherical projection. Table 2 shows the results of different fusion strategies. UniFuse obtains the worst results, and BiFuse ranks second. Our GateFuse achieves the best results on RMSE, RMSE(log), δ, and ranks second on MAE and MRE. We visualize depth maps from different fusion strategies for better comparison in Fig. 9, where BiFuse and UniFuse fail to reconstruct details.

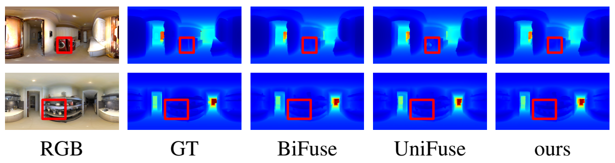


Figure 9. We select two scenes from 360D [59] and compare depth maps from different fusion strategies. Our GateFuse can reconstruct more details, and we mark out these regions with red boxes.

4.6. Limitations

We propose SphereFuion for panorama depth estimation by using the 2D image convolution and the mesh convolution, which achieves competitive results with a lighter network and the highest inference efficiency. However, the cache strategy requires additional GPU memory to store FAF information during training and testing. Furthermore, SphereFusion requires huge GPU memory during training and does not support ultra-high resolution panorama images, such as 1024×2048 [39]. We still need more in-depth research to improve the mesh operation and the quality of panorama depth estimation.

5. Conclusion

This paper introduces a novel panorama depth estimation method, SphereFusion, which combines the strengths of equirectangular and spherical projection. SphereFusion uses 2D image convolution to complement mesh convolution by using a gate fusion module to select reliable features from two encoders and estimates the panorama depth map in the spherical domain to avoid distortion and discontinuity. Meanwhile, SphereFusion utilizes a lighter network and a cache strategy to improve the inference efficiency. Experiments conducted on three popular datasets indicate that SphereFusion achieves competitive results and maintains an impressive inference efficiency of up to 60 FPS for 512×1024 panorama images on one NVIDIA RTX3090.

6. Acknowledgments

This work was supported by the Shenzhen Science and Technology Program (Grant No. KJZD20240903 104103005, JCYJ20220818102414030, JSGGKQTD202 21101115655027).

References

[1] Georgios Albanis, Nikolaos Zioulis, Petros Drakoulis, Vasileios Gkitsas, Vladimiros Sterzentsenko, Federico Alvarez, Dimitrios Zarpalas, and Petros Daras. Pano3d: A holistic benchmark and a solid baseline for 360deg depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3727–3737, 2021. 1

[2] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018. 2

[3] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 1, 6, 7

[4] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4009–4018, 2021. 2

[5] Jia-Wang Bian, Huangying Zhan, Naiyan Wang, Zhichao Li, Le Zhang, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth learning from video. *International Journal of Computer Vision*, 129(9):2548–2564, 2021. 2

[6] 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. *arXiv preprint arXiv:1709.06158*, 2017. 1, 6, 7

[7] Hong-Xiang Chen, Kunhong Li, Zhiheng Fu, Mengyi Liu, Zonghao Chen, and Yulan Guo. Distortion-aware monocular depth estimation for omnidirectional images. *IEEE Signal Processing Letters*, 28:334–338, 2021. 2, 3

[8] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1420–1429, 2018. 2

[9] Xinjing Cheng, Peng Wang, Yanqi Zhou, Chenye Guan, and Ruigang Yang. Omnidirectional depth extension networks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 589–595. IEEE, 2020. 3

[10] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 5

[11] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. 7

[12] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018. 3

[13] Greire Payen de La Garanderie, Amir Atapour Abarghouei, and Toby P Breckon. Eliminating the blind spot: Adapting 3d object detection and monocular depth estimation to 360 panoramic imagery. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 789–807, 2018. 3

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 2

[15] Marc Eder, Pierre Moulon, and Li Guan. Pano popups: Indoor 3d reconstruction with a plane-aware network. In *2019 International Conference on 3D Vision (3DV)*, pages 76–84. IEEE, 2019. 3

[16] Marc Eder, True Price, Thanh Vu, Akash Bapat, and Jan-Michael Frahm. Mapped convolutions. *arXiv preprint arXiv:1906.11096*, 2019. 3

[17] Marc Eder, Mykhailo Shvets, John Lim, and Jan-Michael Frahm. Tangent images for mitigating spherical distortion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12426–12434, 2020. 1, 3, 4

[18] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 2

[19] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014. 2

[20] Yutong Feng, Yifan Feng, Haoxuan You, Xibin Zhao, and Yue Gao. Meshnet: Mesh neural network for 3d shape representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8279–8286, 2019. 2, 3

[21] Clara Fernandez-Labrador, Jose M Facil, Alejandro Perez-Yus, Cédric Demonceaux, Javier Civera, and Jose J Guerrero. Corners for layout: End-to-end layout recovery from 360 images. *IEEE Robotics and Automation Letters*, 5(2): 1255–1262, 2020. 3

[22] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3828–3838, 2019. 2

[23] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2, 3

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 3, 4

[25] Derek Hoiem, Alexei A Efros, and Martial Hebert. Automatic photo pop-up. *ACM Trans. Graph.*, pages 577–584, 2005. 2

[26] Shi-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. Jittor: a novel deep learning framework

with meta-operators and unified graph execution. *Science China Information Sciences*, 63(222103):1–21, 2020. 6

[27] Shi-Min Hu, Zheng-Ning Liu, Meng-Hao Guo, Jun-Xiong Cai, Jiahui Huang, Tai-Jiang Mu, and Ralph R Martin. Subdivision-based mesh convolution networks. *ACM Transactions on Graphics (TOG)*, 41:1–16, 2022. 2, 3, 4

[28] Chiyu Jiang, Jingwei Huang, Karthik Kashinath, Philip Marcus, Matthias Niessner, et al. Spherical cnns on unstructured grids. *arXiv preprint arXiv:1901.02039*, 2019. 3

[29] Hualie Jiang, Zhe Sheng, Siyu Zhu, Zilong Dong, and Rui Huang. Unifuse: Unidirectional fusion for 360 panorama depth estimation. *IEEE Robotics and Automation Letters*, 6 (2):1519–1526, 2021. 1, 2, 3, 5, 6, 7, 8

[30] Lei Jin, Yanyu Xu, Jia Zheng, Junfei Zhang, Rui Tang, Shugong Xu, Jingyi Yu, and Shenghua Gao. Geometric structure based and regularized depth estimation from 360 indoor imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 889–898, 2020. 3

[31] Masum Shah Junayed, Arezoo Sadeghzadeh, Md Baharul Islam, Lai-Kuan Wong, and Tarkan Aydın. Himode: A hybrid monocular omnidirectional depth estimation model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5212–5221, 2022. 2

[32] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 2, 5, 7

[33] Yeonkun Lee, Jaeseok Jeong, Jongseob Yun, Wonjune Cho, and Kuk-Jin Yoon. Spherephd: Applying cnns on 360° images with non-euclidean spherical polyhedron representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3

[34] Yuyan Li, Yuliang Guo, Zhixin Yan, Xinyu Huang, Ye Duan, and Liu Ren. Omnifusion: 360 monocular depth estimation via geometry-aware fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2801–2810, 2022. 1, 2, 3, 6, 7

[35] Vaishakh Patil, Christos Sakaridis, Alexander Liniger, and Luc Van Gool. P3depth: Monocular depth estimation with a piecewise planarity prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1610–1621, 2022. 2

[36] Chi-Han Peng and Jiayao Zhang. High-resolution depth estimation for 360-degree panoramas through perspective and panoramic depth images registration. *arXiv preprint arXiv:2210.10414*, 2022. 2, 3

[37] Giovanni Pintore, Marco Agus, Eva Almansa, Jens Schneider, and Enrico Gobbetti. Slicenet: deep dense depth estimation from a single indoor panorama using a slice-based representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11536–11545, 2021. 1, 2, 5, 6, 7

[38] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2

[39] Manuel Rey-Area, Mingze Yuan, and Christian Richardt. 360monodepth: High-resolution 360deg monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3762–3772, 2022. 3, 8

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 5

[41] 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*, 31(5):824–840, 2008. 2

[42] Zhijie Shen, Chunyu Lin, Kang Liao, Lang Nie, Zishuo Zheng, and Yao Zhao. Panoformer: Panorama transformer for indoor 360 depth estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, pages 195–211. Springer, 2022. 1, 2, 3, 6, 7

[43] Zhijie Shen, Chunyu Lin, Lang Nie, Kang Liao, and Yao Zhao. Neural contourlet network for monocular 360 depth estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(12):8574–8585, 2022. 2

[44] Robert Skupin, Yago Sanchez, Y-K Wang, Miska M Hannuksela, J Boyce, and Mathias Wien. Standardization status of 360 degree video coding and delivery. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2017. 1

[45] Yu-Chuan Su and Kristen Grauman. Kernel transformer networks for compact spherical convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9442–9451, 2019. 3

[46] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Hohonet: 360 indoor holistic understanding with latent horizontal features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2573–2582, 2021. 1, 2, 6, 7

[47] Keisuke Tateno, Nassir Navab, and Federico Tombari. Distortion-aware convolutional filters for dense prediction in panoramic images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 707–722, 2018. 2, 3

[48] Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. Bifuse: Monocular 360 depth estimation via bi-projection fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 462–471, 2020. 1, 2, 3, 5, 6, 7, 8

[49] Ning-Hsu Wang, Bolivar Solarte, Yi-Hsuan Tsai, Wei-Chen Chiu, and Min Sun. 360sd-net: 360 stereo depth estimation with learnable cost volume. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 582–588. IEEE, 2020. 3

[50] Jamie Watson, Oisin Mac Aodha, Victor Prisacariu, Gabriel Brostow, and Michael Firman. The temporal opportunist:

Self-supervised multi-frame monocular depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1164–1174, 2021. 2

[51] Qingsong Yan, Qiang Wang, Kaiyong Zhao, Bo Li, Xiaowen Chu, and Fei Deng. Spheredepth: Panorama depth estimation from spherical domain. In *2022 Tenth international conference on 3D vision (3DV)*, 2022. 1, 2, 3, 4, 6, 7, 8

[52] Wei Yin, Yifan Liu, and Chunhua Shen. Virtual normal: Enforcing geometric constraints for accurate and robust depth prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[53] S Yu-Chuan and G Kristen. Flat2sphere: Learning spherical convolution for fast features from 360 imagery. In *Proceedings of International Conference on Neural Information Processing Systems (NIPS)*, 2017. 3

[54] Weihao Yuan, Xiaodong Gu, Zuozhuo Dai, Siyu Zhu, and Ping Tan. New crfs: Neural window fully-connected crfs for monocular depth estimation. *arXiv preprint arXiv:2203.01502*, 2022. 2

[55] Wei Zeng, Sezer Karaoglu, and Theo Gevers. Joint 3d layout and depth prediction from a single indoor panorama image. In *European Conference on Computer Vision*, pages 666–682. Springer, 2020. 3

[56] Wang Zhao, Shaohui Liu, Yezhi Shu, and Yong-Jin Liu. Towards better generalization: Joint depth-pose learning without posenet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9151–9161, 2020. 2

[57] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *European Conference on Computer Vision*, pages 519–535. Springer, 2020. 1

[58] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1851–1858, 2017. 2

[59] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. Omnidepth: Dense depth estimation for indoors spherical panoramas. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 448–465, 2018. 1, 2, 3, 4, 6, 7, 8