Deep Depth Estimation on 360° Images with a Double Quaternion Loss

Brandon Yushan Feng[1], Wangjue Yao[1], Zheyuan Liu[2], Amitabh Varshney[1]
[1] University of Maryland, College Park
[2] University of Virginia
{yfeng97, wyao123, varshney}@umd.edu, zl4ej@virginia.edu

2020 International Conference on 3D Vision (3DV) | 978-1-7281-8128-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/3DV50981.2020.00062

Abstract

While 360° images are becoming ubiquitous due to popularity of panoramic content, they cannot directly work with most of the existing depth estimation techniques developed for perspective images. In this paper, we present a deep-learning-based framework of estimating depth from 360° images. We present an adaptive depth refinement procedure that refines depth estimates using normal estimates and pixel-wise uncertainty scores. We introduce double quaternion approximation to combine the loss of the joint estimation of depth and surface normal. Furthermore, we use the double quaternion formulation to also measure stereo consistency between the horizontally displaced depth maps, leading to a new loss function for training a depth estimation CNN. Results show that the new double-quaternion-based loss and the adaptive depth refinement procedure lead to better network performance. Our proposed method can be used with monocular as well as stereo images. When evaluated on several datasets, our method surpasses state-of-the-art methods on most metrics.

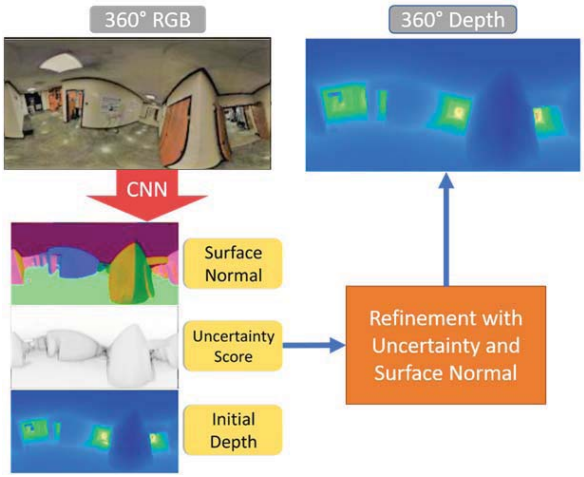

Figure 1. Overview of our method. Our trained CNN takes in a 360° image and for each pixel estimates its depth, normal, and uncertainty of that depth estimate. These three estimates are used by a refinement module to produce the final depth estimate for each pixel. We train the CNN using a novel loss based on the double quaternion representation of the depth and normal.

1. Introduction

Traditional depth estimation uses binocular or multi-view stereo image inputs [4, 9, 30, 34]. Based on explicit geometric constraints, most of these stereo methods infer relative depth through computing stereo disparity, i.e., the distance between a pixel's location in one image to its corresponding location in the other image. The rise of deep learning enables direct training of convolutional neural networks (CNN) for depth estimation by implicitly computing the matching cost between pixels in stereo images.

However, since stereo images are not easily accessible, depth estimation on monocular images serves as a valuable alternative. Deep CNNs for this problem have shown promising results. A unique advantage for this approach is that monocular CNNs can be trained on both monocular image datasets and stereo image datasets.

As virtual and augmented reality (VR and AR) become more commoditized and panoramic cameras become ubiquitous, 360° visual content is becoming more relevant [10, 11, 12]. The interactive nature of VR and AR, fosters an urgent need for methods that estimate depth information from 2D views to instill more creative freedom in content rendering and interaction, including reconstructing the original 3D scenes and synthesizing views from novel angles [7, 8]. However, most of the previous research on depth estimation targets traditional perspective images.

Unlike typical photographs captured on a planar sensor, 360° images have a spherical layout. For 360° stereo images, traditional depth-estimation methods based on binocular disparity are not directly applicable due to the spherical singularity at the stereo epipoles. Moreover, CNNs trained on narrow-field-of-view images for monocular depth estimation perform poorly on 360° monocular images because of the significant domain shift from traditional perspective to wide-field-of-view equirectangular images.

2475-7888/20/$31.00 ©2020 IEEE
DOI 10.1109/3DV50981.2020.00062

Zioulis *et al*. [50] and Lai *et al*. [20] have recently released separate datasets for depth estimation on 360° images. While both datasets provide multi-view stereo images, their choices of baseline distance between cameras are vastly different. This difference signifies a severe drawback of training a CNN that simply takes in a stereo image pair: networks directly trained on stereo images with a particular baseline cannot adapt to different baseline configurations at test time. Moreover, such networks require a fixed baseline in training, making it difficult to aggregate training data from multiple datasets. Therefore, training a depth estimation CNN with monocular input seems more favorable.

Among methods that train CNN for depth estimation, joint estimation of depth and normal is commonly adopted as an augmentation technique. However, to the best of our knowledge, all previous networks that jointly estimate normal and depth consider the errors from depth and normal separately. While Qi *et al*. [32] and Yang *et al*. [46] have proposed depth refinement methods that explicitly link surface normal estimates with depth estimates, their methods are based on the planar sensor camera model for traditional narrow-field-of-view images and do not map well to 360° images. Moreover, their refinement procedures modify all pixel points uniformly in the estimated depth map and do not consider the varying quality across different regions.

In this paper, we present a new framework for 360° depth estimation. We start from a generic CNN that jointly estimates depth and surface normal based on monocular RGB images. We develop a new loss for this joint estimation task, which combines depth and surface normals into a 4D hyperspherical space with a double quaternion approximation. We implement depth refinement using the normal estimates produced by this network. In contrast with previous normal-based refinement methods on perspective images, our new method adaptively adjusts the refinement to the initial depth estimates by an uncertainty score map which is also estimated by the CNN. This uncertainty construct allows us to identify image regions where further refinement could be helpful and avoid unnecessary changes to estimates that the network expects to be accurate. Furthermore, to make full use of available image data, we introduce a stereo loss when training the CNN on stereo-image pairs. After producing two separate monocular estimates of depth and normal for a stereo image pair, the CNN learns to minimize their hyperspherical angular difference. By this design, the monocular network can take advantage of stereo training data without being restricted by a particular stereo baseline distance. Experiments show the improved performance of our proposed framework compared to previous methods on 360° depth estimation.

In summary, our contributions include:

- An adaptive depth refinement framework for 360° images using normal estimates and uncertainty scores.
- A new way to incorporate depth and surface normal estimates for a 3D point into a hyperspherical 4D space using a double quaternion approximation.
- A stereo loss that enables the CNN to learn stereo consistency and remain flexible across datasets with different stereo baseline distances.

2. Related Work

We first present learning-based methods for monocular and stereo depth estimation on 360° images, followed by previous work on using the surface normal to refine depth from perspective images. We then present previous approaches that incorporate quaternion representations in estimating surface normals and approximating 3D motions.

2.1. Depth Estimation on 360° Images

Several methods have been used to perform depth estimation [21, 24, 25, 26, 33, 35, 42, 45] and surface normal estimation [2, 3, 23, 42] on perspective images. Unfortunately, 360° images are distorted by equirectangular projection and contain irregular disparity pattern due to the spherical singularity at the stereo epipoles. Therefore, depth estimation on 360° images requires special adaptations.

One approach for learning on 360° images is to project pixels onto rectified cubemaps and then perform inference using pre-trained CNNs. Huang *et al*. [18] apply the traditional structure-from-motion (SfM) algorithm [17] in 3D scene reconstruction by projecting each 360° video frame onto a cubemap. Monroy *et al*. [29] obtain 360° saliency maps following this approach, but the distortion and discontinuity among cubemap patches are not handled by their method. Cube padding [6, 40] was introduced to help resolve cubemap distortion problem by padding each patch with features from adjacent cubemap patches.

Another approach for 360° depth estimation is to transfer models for perspective images to 360° images. To account for the distortion from equirectangular projection, Su and Graumann [37] modified a CNN trained on perspective images by varying the kernel shape based on its location on the sphere. Su and Graumann [38] improve the previous method by learning a transformation function for kernels pre-trained on perspective images without separately training new kernels for each location. Zioulis *et al*. [49] directly train CNNs on 360° images using rectangular kernels of varying resolutions along with traditional square kernels to cover different distortion levels. They also adopt dilated convolutions [47] to increase the receptive field and enable the networks to gather more global information. Lee *et al*. [22] use a spherical polyhedron to represent 360° images and devise special convolution and pooling kernels for image pixels after they are projected on the polyhedron. Tateno *et al*. [39] deform the kernel sampling grid to com-

pensate for distortions in spherical images. For the similar task of saliency detection on 360° videos, Zhang *et al*. [48] also define kernels on the 360° sphere and resample the kernels on the grid points for every location in the equirectangular projection.

Unsupervised learning through view synthesis has also been exploited to solve depth estimation [16, 46]. De La Garanderie *et al*. [31] use the stereo consistency of perspective images to achieve unsupervised depth estimation on panoramic images. Wang *et al*. [40] explore self-supervised depth estimation from 360° images through cubemap projection. Zioulis *et al*. [50] introduced the view-synthesis approach into the realm of omnidirectional 360° images. Aware of the distortion problem of 360° images, they also adaptively weight the loss contribution of each pixel based on its coordinates on the image grid.

While most previous work on 360° depth estimation focuses on monocular input, Lai *et al*. [20] present a framework for stereo depth estimation on 360° images with a CNN which produces a depth map for a horizontally displaced pair of images. Xie *et al*. [44] further extend this stereo depth estimation framework to include deformable convolution and correlation convolution. Wang *et al*. [41] propose a learnable cost volume approach for spherical stereo depth estimation which also shows promising results.

2.2. Joint Estimation of Depth and Normal

Motivated by the inherent geometric relationship between depth and normal estimates of points on the same surface, several methods include the surface normal information into depth estimation. Wang *et al*. [43] deploy a dense conditional random field on initial estimates of normal and depth, which produces more regularized depth and normal outputs with better geometric consistency. Eigen and Fergus [14] also simultaneously estimate depth, surface normal, and semantic segmentation for perspective images.

Furthermore, the depth-normal relationship can be explicitly constructed. Two spatially close points with similar surface normal estimates are approximately co-planar, and thus they form a vector that is orthogonal to the surface normal. Building upon this assumption, Qi *et al*. [32] introduce a module that refines the depth estimates produced by a CNN using its normal estimates. Likewise, Yang *et al*. [46] formulate this depth-normal relationship as a quadratic minimization problem for a set of linear equations constructed by the local depth and normal estimates in a small region. However, these methods do not consider the varying quality of CNN estimates across different regions.

Lai *et al*. [20] also use the information of surface normal to improve depth estimation. To the best of our knowledge, this is the first work that implements a joint estimation of depth and normal on 360° images. However, their method only includes normal as an auxiliary task of the CNN, without further exploiting the explicit geometric relationship between depth and surface normal.

2.3. Use of Quaternions

Quaternions are widely used in computer graphics to represent rotation transformation of 3D points. By representing surface normal as a pure quaternion, Karakottas *et al*. [19] calculate the angular loss of normal predictions based on the quaternion product of the estimated and ground-truth normal vectors.

As a natural extension to quaternions, double quaternions integrate the rotation and translation components for motion interpolation [15]. Unlike traditional 3D point representation where spatial displacements are separately characterized into translation and rotation, double quaternions provide a unified framework to approximate 3D displacements as a rotation in the 4D space. In other words, the difference between two 3D spatial displacements can be described by their angular distance in 4D.

In this paper, we introduce a method that directly unifies depth and surface normal information into a single measurement based on a double quaternion approximation. With this novel construct, the predicted and the ground-truth depth and normals can be converted into two double quaternions. We thus derive a new loss specifically for joint estimation of depth and surface normals.

We also take advantage of this double quaternion representation to measure the discrepancy between two CNN estimates from a stereo image pair. After transforming the two separate estimates into a homogeneous coordinate system, we derive a stereo loss based on the double quaternion angular distance between these two sets of estimates.

Algorithm 1: Steps to Compute Training Loss

Input: Horizontally Displaced Image Pair I, $I^{'}$
Parameters: Weights θ of CNN
Label: Depth Map D_I and Surface Normal Map S_I
Output: Initial depth $\tilde{\mathrm{D}}_{\text{init}}$, refined depth $\tilde{\mathrm{D}}_I$, estimated normal $\tilde{\mathrm{S}}_I$, total loss L_{total}
For each training iteration
  $\tilde{\mathrm{D}}_{\text{init}}, \tilde{\mathrm{S}}_I = \mathrm{CNN}(I)$ (Section 3.1)
  $\tilde{\mathrm{D}}_{\text{init}'}, \tilde{\mathrm{S}}_{I'} = \mathrm{CNN}(I^{'})$ (Section 3.1)
  $\tilde{\mathrm{D}}_I = \mathrm{Refine}(\tilde{\mathrm{D}}_{\text{init}}, \tilde{\mathrm{S}}_I)$ (Sections 3.2 - 3.3)
  $\tilde{\mathrm{D}}_{I'} = \mathrm{Refine}(\tilde{\mathrm{D}}_{\text{init}'}, \tilde{\mathrm{S}}_{I'})$ (Sections 3.2 - 3.3)
  $L_{\text{DQ}} = \mathrm{DQLoss}(\tilde{\mathrm{D}}_I, \tilde{\mathrm{S}}_I, \mathrm{D}_I, \mathrm{S}_I)$ (Section 3.4)
  $L_{\text{Stereo}} = \mathrm{StereoLoss}(\tilde{\mathrm{D}}_I, \tilde{\mathrm{D}}_{I'})$ (Section 3.5)
  $L_{\text{berHu}} = \mathrm{berHuLoss}(\tilde{\mathrm{D}}_I, \mathrm{D}_I)$ (Section 3.6)
  $L_{\text{total}} = L_{\text{berHu}} + L_{\text{DQ}} + L_{\text{Stereo}}$
  $\theta = \mathrm{Update}(L_{\text{total}}, \theta)$

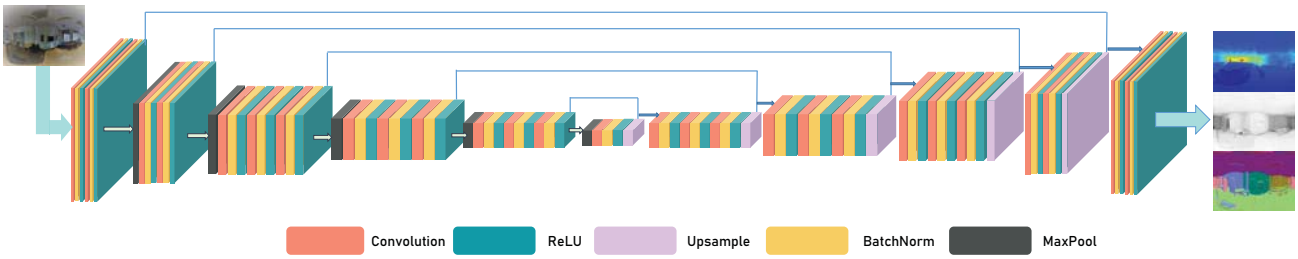

Figure 2. Network Architecture. We adopt the commonly used U-Net architecture for end-to-end per-pixel estimation. The first six blocks in the encoding part are based on the VGG-16 model [36]. The decoding part is symmetric to the encoding part, and it outputs a depth map, an uncertainty map, and a normal map. These three maps are combined to produce a final depth map using the method described in Section 3

3. Method

Our goal is to train a CNN for 360° depth estimation. To exploit the information from surface normals, the CNN produces a normal map and an uncertainty map for initial depth estimates, which we feed into a refinement procedure to produce a final depth map. We derive a loss function based on double quaternions to facilitate better depth-normal joint learning. To further use datasets containing stereo pairs, we introduce a stereo loss also based on double quaternions.

3.1. CNN Architecture

We adopt the commonly used U-Net architecture with skip connections, as shown in Figure 2. For an input RGB image of size $h \times w \times 3$, the CNN produces three separate outputs: 1) $h \times w \times 1$ depth map, 2) $h \times w \times 1$ uncertainty map for depth, and 3) $h \times w \times 3$ normal map. These three output maps are fed into a refinement step detailed in Section 3.2.

3.2. Depth Refinement based on Normal

In general, image-based depth estimation aims to recover the depth value of a 3D point (x, y, z) given its projected pixel location (u, v) in an image.

The depth value for a pixel in a 360° image is defined as the distance of its corresponding 3D point from the camera.

$$r = \sqrt{x^2 + y^2 + z^2} \tag{1}$$

Moreover, the pixel coordinates (u, v) of a 360° image with width w and height h directly correspond to the spherical coordinates (θ, ϕ) of its corresponding 3D point.

$$\phi = 2\pi u \qquad \theta = \pi \frac{2v - 1}{2} \qquad u, v \in [0, 1] \tag{2}$$

The direct conversion between spherical and Cartesian coordinates in 3D is given as follows

$$x = r \sin\phi \sin\theta \qquad y = r \cos\theta \qquad z = r \cos\phi \sin\theta \tag{3}$$

Using equations (2) and (3), we can obtain the relationship that maps 2D grid coordinates to 3D Cartesian coordinates for 360° depth maps.

Using the normal estimates (n_{ix}, n_{iy}, n_{iz}) also produced by the CNN, we can further formulate the following equation based on the orthogonality between surface normal vector and in-plane vector among points (x_i, y_i, z_i) and (x_j, y_j, z_j):

$$n_{ix}(x - x_i) + n_{iy}(y - y_i) + n_{iz}(z - z_i) = 0 \tag{4}$$

$$\frac{n_{ix}x_j + n_{iy}y_j + n_{iz}z_j}{n_{ix}x_i + n_{iy}y_i + n_{iz}z_i} = 1 \tag{5}$$

Then, using an assumption similar to Qi *et al.* [32], for pixels within a small region, we treat their corresponding 3D points as co-planar if their surface normal estimates are also similar. Thus, we obtain an approximately co-planar neighborhood N_i for each image pixel P_i using spatio-angular measures defined as follows:

$$N_i = \{(x_j, y_j, z_j) \mid n_j^\intercal n_i > \alpha, |u_i - u_j| < \beta, |v_i - v_j| < \beta\} \tag{6}$$

where (u_i, v_i) and (u_j, v_j) are the 2D grid coordinates of pixels P_i and P_j, β is the parameter that controls the size of the spatial neighborhood, and α controls the size of the angular neighborhood. A larger value of $n_j^\intercal n_i$ implies a greater likelihood that the corresponding 3D points for P_i and P_j are co-planar.

For each neighbor $P_j \in N_i$, we may obtain an estimate r_{ij} for the depth of r_i of P_i by plugging in the spherical coordinates with equations (3) and (5):

$$r_{ij} = \frac{n_{ix}x_j + n_{iy}y_j + n_{iz}z_j}{n_{ix}\sin\phi_i \sin\theta_i + n_{iy}\cos\theta_i + n_{iz}\cos\phi_i \sin\theta_i} \tag{7}$$

where θ_i and ϕ_i are determined by Eq (2). Note that the calculation in Eq (7) suffers from instability when the denominator is close to zero, producing abnormal values.

Thus, we leave out any depth estimate that violates the following constraints:

$$0 < r_{ij} < 255 \qquad max(\frac{r_{ij}}{r_i}, \frac{r_i}{r_{ij}}) < 10 \tag{8}$$

For any r_{ij} that violates the constraints in Eq (8), we set it as r_i, the original depth estimate of P_i.

3.3. Aggregation with Confidence Scores

For each pixel P_i, we aggregate the estimates of its depth r_i from its neighbors $P_j \in N_i$ by using normalized weights. These weights have two components. First, we use the uncertainty score q_j of pixel P_j from the CNN output to compute its confidence value $\boldsymbol{C}(P_j) = 1 - q_j^2$. An example of the uncertainty score output maps can be seen in Figure 5. Second, the neighbor P_j's contribution is also weighted by $\boldsymbol{W}(P_i, P_j)$, the dot product between their respective normals, n_i and n_j.

Specifically, we aggregate the depth estimates for each pixel P_i with its neighbors as:

$$r_i^N = \frac{\sum_{P_j \in N_i} \boldsymbol{C}(P_j) \cdot \boldsymbol{W}(P_i, P_j) \cdot r_{ij}}{\sum_{P_j \in N_i} \boldsymbol{C}(P_j) \cdot \boldsymbol{W}(P_i, P_j)} \tag{9}$$

with $\boldsymbol{C}(P_j) = 1 - q_j^2$ and $\boldsymbol{W}(P_i, P_j) = n_j^\intercal n_i$.

Finally, the refined $\hat{r_i}$ for P_i is calculated as:

$$\hat{r_i} = \boldsymbol{C}(P_i) \cdot r_i + (1 - \boldsymbol{C}(P_i)) \cdot r_i^N \tag{10}$$

In other words, for a pixel with higher uncertainty and lower confidence, we place a greater reliance on its neighbors to refine its initial depth estimate. On the other hand, if a pixel has a low uncertainty score, the CNN believes this depth estimate is likely accurate, and so the neighbor estimates are less informative. This formulation allows us to adaptively refine the initial CNN estimates and reduce the unnecessary modifications of the already robust estimates.

3.4. Double Quaternion Approximation of Depth and Normal in the Loss function

3.4.1 Constructing the double quaternion

Since a point's spatial coordinate (x, y, z) represents a translation from the coordinate origin $(0, 0, 0)$, a pixel's corresponding depth and surface normal orientation can be viewed as 3D translation and rotation, respectively.

The translation component of a 2D spatial displacement can be viewed as a rotation with respect to the origin of the 3D coordinate system. In fact, similar approximation can be done from 3D to 4D. McCarthy [27, 28] has shown that the homogeneous transform of 3D spatial displacements with rotation and translation is the limiting case of a 4D rotation as the radius of the 4D sphere R approaches infinity. Thus, we combine the 3D depth (translation) and normal (rotation) into one 4D measurement, which is represented by a double quaternion. Specifically, a 3D translation d can be approximated by a rotation on a 4D sphere of radius $R, \lim R \to \infty$ by an angle ψ as $\lim_{\psi \to 0} \sin(\psi) = \psi = \frac{d}{R}$. The double quaternion representing this 4D rotation is:

$$\mathbf{D} = \cos(\frac{\psi}{2}) + \sin(\frac{\psi}{2}) \frac{d}{|d|} \tag{11}$$

Therefore, the 3D translation can be represented by a double quaternion $(\mathbf{D}, \mathbf{D^*})$, and the 3D rotation is represented as $(\mathbf{Q}, \mathbf{Q})$, where

$$\mathbf{Q} = (0, n_x, n_y, n_z) \tag{12}$$

Two double quaternions, ($\mathbf{G_1}$, $\mathbf{H_1}$) and ($\mathbf{G_2}$, $\mathbf{H_2}$) can be composed into a new double quaternion ($\mathbf{G_3}$, $\mathbf{H_3}$), where

$$\mathbf{G_3} = \mathbf{G_1 G_2} \qquad \mathbf{H_3} = \mathbf{H_1 H_2} \tag{13}$$

Moreover, following Ge *et al*. [15], we can compute the spatial distance between two double quaternions $(\mathbf{G_1}, \mathbf{H_1})$ and $(\mathbf{G_2}, \mathbf{H_2})$ as the angle between the respective double quaternion components:

$$\alpha = \cos^{-1}(\mathbf{G_1} \cdot \mathbf{G_2}) \qquad \beta = \cos^{-1}(\mathbf{H_1} \cdot \mathbf{H_2}) \tag{14}$$

3.4.2 Loss function based on double quaternions

Based on Eq (13), we combine the double quaternions representing translation and rotation (Eqs (11) and (12)) into a double quaternion representation for a depth and normal estimation pair:

$$\mathbf{G} = \mathbf{DQ} \qquad \mathbf{H} = \mathbf{D}^* \mathbf{Q} \tag{15}$$

We thus derive a loss function based on the angular distance between the two double quaternions: predicted ($\mathbf{G^{Pred}}$, $\mathbf{H^{Pred}}$) and ground-truth ($\mathbf{G^{GT}}$, $\mathbf{H^{GT}}$) as

$$L_{DQ} = \sqrt{\alpha_{DQ}^2 + \beta_{DQ}^2} \tag{16}$$

where α_{DQ} and β_{DQ} are calculated as in Eq (14)

3.5. Stereo consistency

While training on datasets with stereo pairs, we further impose a stereo loss to minimize the discrepancy between the estimates from the two horizontally displaced images.

Given a known baseline distance b between a horizontal stereo pair, a pixel in one image L can be mapped onto the other image R with the following equation:

$$\phi_R = \phi_L + b \cdot \frac{\cos(\phi_L)}{r_L \cdot \sin(\theta_L)} \qquad \theta_R = \theta_L + b \cdot \frac{sin(\phi_L) cos(\theta_L)}{r_L} \tag{17}$$

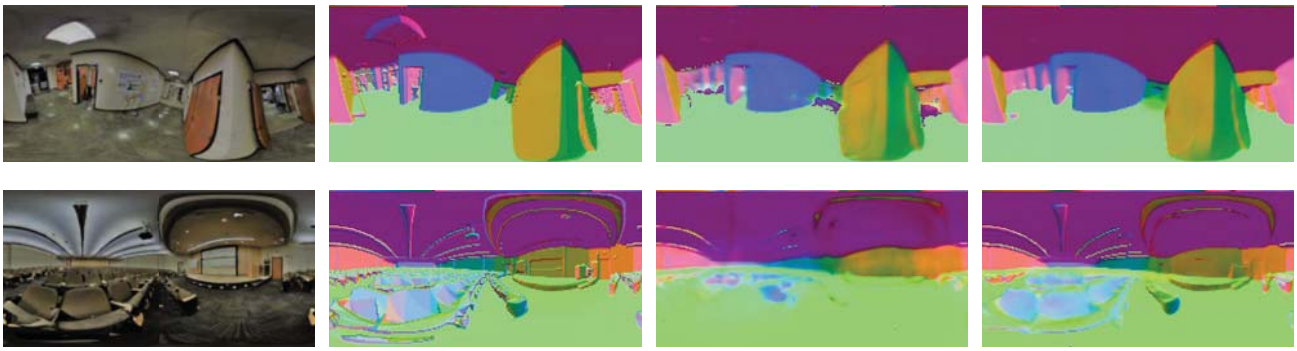

Figure 3. This example shows that training with double quaternion loss also enables the network to produce better surface normal estimates. In the first column, we show the input image for reference. In the third column, we show predictions from the traditional baseline model in which we separately calculate depth and normal loss without combining them into a double quaternion form. In the fourth column, we show results from our full model.

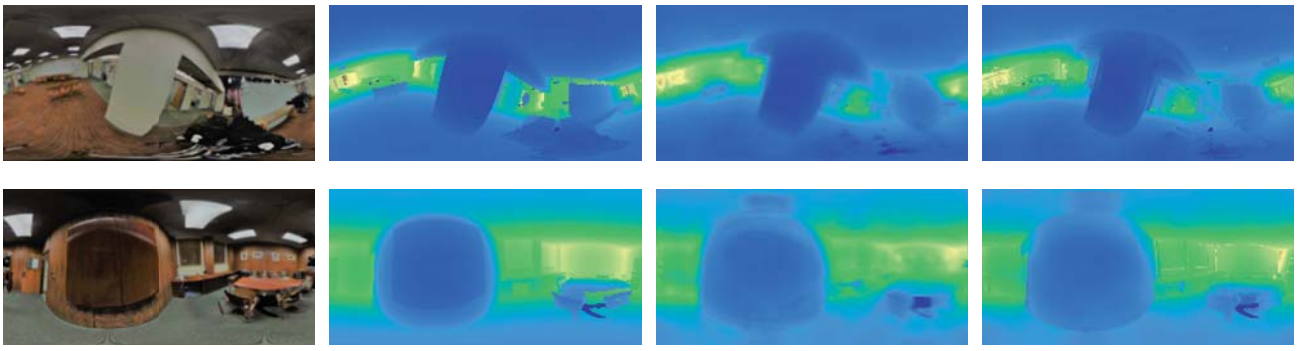

Figure 4. Comparison of initial depth estimates produced by the network and the refined output based on surface normal. In the first column, we show the input image for reference.

Method	RMSE	Log10	AbsRel	δ_1	δ_2	δ_3
UResNet [49]	2.037	0.326	16.906	0.213	0.399	0.560
RectNet [49]	1.738	0.291	16.132	0.240	0.453	0.634
FCRN [21]	0.672	0.101	7.448	0.806	0.932	0.966
PSMNet [5]	0.393	0.059	5.641	0.953	0.975	0.980
SepUNet [20]	0.495	0.042	1.779	0.944	**0.987**	**0.993**
SepUNetS [20]	0.614	0.072	1.841	0.835	0.966	0.985
SepUNetDD [44]	0.392	0.036	2.120	**0.960**	0.987	0.992
Ours	**0.389**	**0.031**	**0.413**	0.954	0.984	0.990

Table 1. Performance Comparison on the ODS dataset [20]. Evaluation statistics for row 1-7 are directly taken from Lai *et al*. [20] and Xie *et al*. [44]. Our method produces superior results in most metrics.

Following the procedure presented in Section 3.2, we combine depth and normal estimates from the stereo pair images into two double quaternions ($\mathbf{G^L}$, $\mathbf{H^L}$) and ($\mathbf{G^R}$, $\mathbf{H^R}$), from which we calculate the stereo loss:

$$L_{\text{Stereo}} = \sqrt{\alpha^2_{\text{Stereo}} + \beta^2_{\text{Stereo}}} \tag{18}$$

where α_{Stereo} and β_{Stereo} are also calculated as in Eq (14).

3.6. Overall Loss function

With the double-quaternion-based losses derived above, we present the overall loss function for network training:

$$L_{\text{total}} = L_{\text{berHu}} + L_{DQ} + L_{\text{Stereo}} \tag{19}$$

Here, L_{berHu} is the reverse Huber loss function for both the depth and normal estimates compared to their respective ground truth [21]. In effect, this loss is equivalent to

Method	RMSE	RMLSE	AbsRel	SqRel	δ_1	δ_2	δ_3
UResNet [49]	0.3374	0.1204	0.0835	0.0416	0.9319	0.9889	0.9968
RectNet [49]	0.2911	0.1017	**0.0702**	0.0297	0.9574	0.9933	0.9979
monoDepth [16]	7.2097	0.8200	0.4747	2.3783	0.2970	0.7900	0.7510
FCRN [21]	0.9410	0.3760	0.3181	0.4469	0.4922	0.7792	0.9150
DCRF [26]	1.1596	0.4400	0.4202	0.7597	0.3889	0.7044	0.8774
Ours	**0.2373**	**0.0907**	0.0859	**0.0213**	**0.9690**	**0.9954**	**0.9988**

Table 2. Performance Comparison the 360D dataset [50]. Evaluation statistics for row 1-5 are directly taken from Zioulis *et al*. [49]. Our method surpasses other methods in all metrics except for AbsRel.

Input RGB image | Predicted uncertainty map | Predicted depth map

Figure 5. This example illustrates that the network learns to produce meaningful uncertainty maps by effectively grasping the object's geometric outline. It places higher uncertainty near object edges, where depth predictions tend to be overly smooth and prone to error.

Cropped input RGB image | Predicted by RectNet [49] | Predicted by our network

Figure 6. More qualitative comparison. Here we show an example from a test image from the 360D dataset [49]. Note that our result largely preserves the geometry of the hallway railings.

the mean absolute error for errors below a threshold c, and equivalent to a weighted mean squared loss for errors larger than c. We follow Laina *et al*. [21] and set c as 20% of the maximal error among all images of the current batch. We follow Lai *et al*. [20] and place extra weight for error at boundary pixels in calculating L_{berHu}.

4. Experiments

We have trained and evaluated the performance of our method on the ODS dataset [20]. It contains $40,000$ frames of indoor scenes from the Stanford 2D-3D-Semantics Dataset [1] with ground truth depth and surface normal. We adopt the same training-validation data split and evaluation metrics as Lai *et al*. [20]. We have also evaluated our method on the 360D dataset provided by Zioulis *et al*. [50].

4.1. Training details

We initialize the encoding blocks of the CNN shown in Figure 2 with the commonly used VGG-16 [36] pre-trained weights. We use the Adam optimizer with its default parameters. We follow the data augmentation procedures detailed in Lai *et al*. [20] to introduce more variability in data. To be consistent with previous work, we train our networks for 40 epochs on this dataset to enable direct comparison of method performance. We adopt the conventional depth estimation metrics [13, 20, 21, 49]. We denote the absolute prediction error of a pixel i as $\mathrm{E}_i = |y_i - \hat{y}_i|$, where y_i is the ground truth depth and $\hat{y}_i$ is the predicted depth. δ_j refers to the percentage of pixels with $\max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i}) < 1.25^j$. The other metrics used and their definitions are listed below:

$$\text{RMSE}: \sqrt{\frac{\sum_{i=1}^{N} \mathrm{E}_i{}^2}{N}} \quad \text{Abs. Rel.}: \frac{\sum_{i=1}^{N} \frac{\mathrm{E}_i}{\hat{y}_i}}{N}$$

$$\text{RMLSE}: \sqrt{\frac{\sum_{i=1}^{N} |\ln y_i - \ln \hat{y}_i|^2}{N}} \quad \text{Sq. Rel.}: \frac{\sum^{N} \frac{\mathrm{E}_i{}^2}{\hat{y}_i}}{N}$$

$$\text{Log10}: \frac{\sum_{i=1}^{N} |\log_{10} y_i - \log_{10} \hat{y}_i|}{N}$$

Method	RMSE	RMLSE	Log10	AbsRel	SqRel	δ_1	δ_2	δ_3
Full model	**0.3894**	**0.2572**	**0.0312**	**0.4130**	**0.6872**	**0.9543**	**0.9836**	**0.9904**
w/o L_{DQ}	0.4731	0.3452	0.0432	0.5830	0.9012	0.9257	0.9718	0.9880
w/o Refinement	0.4114	0.3190	0.0393	0.5535	0.9220	0.9313	0.9780	0.9903
w/o L_{Stereo}	0.3953	0.2622	0.0321	0.4562	0.7068	0.9530	0.9801	0.9904

Table 3. Ablation Results. Evaluation statistics are based on prediction results on the ODS dataset [20]. Results in rows 2-4 show the network performance when trained without the double quaternion loss, depth refinement step, and stereo consistency loss, respectively. Results show that each component in our proposed method contributes to better estimation accuracy.

4.2. Comparison with Other Methods

The network performance of a CNN trained with our method is shown in Tables 1 and 2. Compared to other methods in Tables 1 and 2, our network shows improved performance in almost all metrics. In Figure 6 we show an example where our method better preserves the geometric detail of the scene. We believe this is because our model is aware of the surface normals and can use it to improve depth estimation.

4.3. Ablation Studies

We present the performance comparisons in Table 3. We observe decreased estimation accuracy with the removal of each component of the loss function. Figures 3, 4, and 5 further illustrate the impact of our method. It is worth noting that the network trained with double quaternions shows smoother normal estimates, which could explain the increase in estimate accuracy since the normal-based refinement method relies on accurate normal estimates.

5. Limitations and Conclusion

We have shown how double-quaternion loss is useful in reducing the geometric inconsistency and improving estimation accuracy. Our results indicate that a double quaternion construct could have a meaningful potential for other tasks that involve processing 360° images. We hope our work will bring a new hyperspherical perspective to analyzing omnidirectional visual data, as a complement to the traditional Cartesian (or equirectangular) perspective.

Our method achieves good performance on the testing scenes in the given datasets. One of the assumptions our method makes is that the normals can be estimated well and provide meaningful guidance for depth refinement. Also, the quality of our depth estimation on real world 360° images is dependent on their domain similarity to the training dataset on which the model is trained. Our method does not perform well if either of these assumptions do not hold. In Figure 7 we show an example of the trained model struggling to produce good depth estimation for a scene with a vastly different structure than those in the training data. This also reveals the limitation of monocular depth estimation despite augmenting it with stereo loss - the network needs a large amount of diverse training data to generalize well on uncommon scenarios. Therefore, it is crucial to collect a richly diverse 360° image dataset with labelled depth and surface normals.

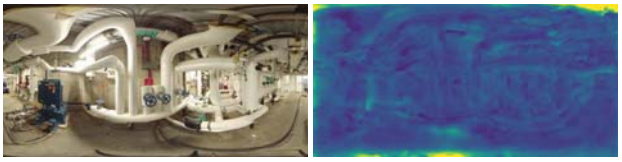

Figure 7. Failure case. The quality of depth estimation produced by our trained network is inherently limited by its training data. Here we show an example where the trained network encounters a scene captured in a mechanical room, which has a vastly different layout than the lecture rooms and hallways used in training data.

Furthermore, as previously discussed, direct learning on 360° images suffers from image distortion, which is not explicitly addressed by our method. In particular, we directly deploy a 2D CNN with regular, square kernels without any modification. Thus, it would be worthwhile to incorporate methods that alleviate the distortion problem, such as modifying convolutional kernels to account for distortion, and directly performing convolution on spheres instead of images with equirectangular projection.

In summary, we present a new framework for 360° depth estimation using CNN. We use the double quaternion formulation to integrate depth and surface normal in loss calculations. Experiments show superior results for the joint depth and normal estimation task. We also extend the double quaternion formulation to establish stereo consistency from the training data without restricting the network to a fixed baseline. We demonstrate quantitative and qualitative results that confirm the benefits of our new approach.

Acknowledgments

We thank the anonymous reviewers for their insightful comments that have significantly improved this paper. This work has been supported in part by the NSF Grants 15-64212 and 18-23321. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.

References

[1] I. Armeni, S. Sax, A. R. Zamir and S. Savarese. "Joint 2D-3D-Semantic Data for Indoor Scene Understanding," arXiv:1702.01105 [cs], Feb. 2017.

[2] A. Bansal, B. Russell and A. Gupta, "Marr Revisited: 2D-3D Alignment via Surface Normal Prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5965-5974, 2016.

[3] A. Bansal, X. Chen, B. Russell, A. Gupta and D. Ramanan, "PixelNet: Representation of the Pixels, by the Pixels, and for the Pixels," arXiv:1702.06506 [cs], Feb. 2017.

[4] S. Bista, I. L. L. da Cunha and A. Varshney, "Kinetic Depth Images: Flexible Generation of Depth Perception," in *The Visual Computer*, 33(10), pp. 1357-1369, 2017.

[5] Jia-Ren Chang and Yong-Sheng Chen, "Pyramid Stereo Matching Network," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5410-5418, 2018.

[6] H. Cheng, C. Chao, J. Dong, H. Wen, T. Liu and M. Sun, "Cube Padding for Weakly-Supervised Saliency Prediction in 360° Videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1420-1429, 2018.

[7] R. Du and A. Varshney, "Social Street View: Blending Immersive Street Views with Geo-Tagged Social Media," in *Proceedings of the 21st International Conference on Web3D Technology*, pp. 77-85, 2016.

[8] R. Du, S. Bista and A. Varshney, "Video Fields: Fusing Multiple Surveillance Videos into a Dynamic Virtual Environment," in *Proceedings of the 21st International Conference on Web3D Technology*, pp. 165-172, 2016.

[9] R. Du, M. Chuang, W. Chang, H. Hoppe and A. Varshney, "Montage4D: Real-time Seamless Fusion and Stylization of Multiview Video Textures," in *Journal of Computer Graphics Techniques*, 8(1), pp. 1-34, 2019.

[10] R. Du, D. Li and A. Varshney. "Project Geollery.com: Reconstructing a Live Mirrored World With Geotagged Social Media,"in *Proceedings of the 24th International Conference on Web3D Technology (Web3D)*, pp. 1-9, July 26-28, 2019.

[11] R. Du, D. Li and A. Varshney. "Geollery: A Mixed Reality Social Media Platform,"in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 1-13, May 2019.

[12] R. Du, D. Li and A. Varshney. "Interactive Fusion of 360° Images for a Mirrored World,"in *2019 IEEE Conference on Virtual Reality and 3D User Interfaces*, pp. 1-13, March 2019.

[13] D. Eigen, C. Puhrsch and R. Fergus. "Depth Map Prediction from a Single Image using a Multi-Scale Deep Network,"in *Advances in Neural Information Processing Systems*, pp. 2366-2374, 2014.

[14] D. Eigen and R. 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*, pp. 2650-2658, 2015.

[15] Q. J. Ge, A. Varshney, J. P. Menon and C. Chang, "Double Quaternions for Motion Interpolation," in *Proceedings of the ASME Design Engineering Technical Conference*, 1998.

[16] C. Godard, O. M. Aodha and G. J. Brostow, "Unsupervised Monocular Depth Estimation with Left-Right Consistency," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 270-279, 2017.

[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, ISBN: 0521540518, Second Edition, 2004.

[18] J. Huang, Z. Chen, D. Ceylan and H. Jin, "6-DOF VR videos with a single 360-camera," in *2017 IEEE Virtual Reality (VR)*, 2017.

[19] A. Karakottas, N. Zioulis, S. Samaras, D. Ataloglou, V. Gkitsas, D. Zarpalas and P. Daras, "360° Surface Regression with a Hyper-Sphere Loss," in *2019 International Conference on 3D Vision (3DV)*, pp. 258-268, 2019.

[20] P. K. Lai, S. Xie, J. Lang and R. Laganière, "Real-Time Panoramic Depth Maps from Omni-directional Stereo Images for 6 DoF Videos in Virtual Reality," in *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 405-412, 2019.

[21] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari and N. Navab, "Deeper Depth Prediction with Fully Convolutional Residual Networks," in *2016 Fourth international conference on 3D vision (3DV)*, pp. 239-248, 2016.

[22] Y. Lee, J. Jeong, J. Yun, W. Cho and K. Yoon, "SpherePHD: Applying CNNs on a spherical polyhedron representation of 360deg images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9181-9189, 2019.

[23] B. Li, C. Shen, Y. Dai, A. Hengel and M. He, "Depth and Surface Normal Estimation From Monocular Images Using Regression on Deep Features and Hierarchical CRFs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1119-1127, 2015.

[24] B. Liu, S. Gould and D. Koller, "Single Image Depth Estimation from Predicted Semantic Labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1253–1260, 2010.

[25] M. Liu, M. Salzmann and X. He, "Discrete-Continuous Depth Estimation from a Single Image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 716–723, 2014.

[26] F. Liu, C. Shen, G. Lin and I. Reid, "Learning Depth from Single Monocular Images Using Deep Convolutional Neural Fields," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2024-2039, 2015.

[27] J. M. McCarthy, "Planar and Spatial Rigid Motion as Special Cases of Spherical and 3-Spherical Motion," in *Journal of Mechanisms, Transmissions, and Automation in Design*, pp. 569-575, Sep. 1983.

[28] J. M. McCarthy, "The Generalization of Line Trajectories in Spatial Kinematics to Trajectories of Great Circles on a Hypersphere," in *Journal of Mechanisms, Transmissions, and Automation in Design*, pp. 60-64, Mar. 1986.

[29] R. Monroy, S. Lutz, T. Chalasani and A. Smolic, "SalNet360:Saliency maps for omni-directional images with CNN," in *Signal Processing: Image Communication*, pp. 26-34, May 2018.

[30] R. Patro, C. Y. Ip, S. Bista and A. Varshney, "Social Snapshot: A System for Temporally Coupled Social Photography," in *IEEE Computer Graphics and Applications*, 31(1), pp. 74-84, 2011.

[31] G. Payen de La Garanderie, A. A. Abarghouei and T. 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)*, pp. 789-807, 2018.

[32] X. Qi, R. Liao, Z. Liu, R. Urtasun and J. Jia, "GeoNet:Geometric Neural Network for Joint Depth and Surface Normal Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 283-291, 2018.

[33] A. Roy and S. Todorovic, "Monocular Depth Estimation Using Neural Regression Forest," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5506-5514, 2016.

[34] A. C. Sankaranarayanan, R. Patro, P. Turaga, A. Varshney and R. Chellappa, "Modeling and Visualization of Human Activities for Multicamera Networks," in *EURASIP Journal on Image and Video Processing*, article 259860, pp. 1-13, 2009.

[35] A. Saxena, S. H. Chung and A. Y. Ng, "Learning Depth from Single Monocular Images," in *Advances in Neural Information Processing Systems*, 2006.

[36] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv:1409.1556 [cs], Sep. 2014.

[37] Y. Su and K. Grauman, "Learning Spherical Convolution for FastFeatures from 360° Imagery," in *Advances in Neural Information Processing Systems*, pp. 529–539, 2017.

[38] Y. Su and K. Grauman, "Kernel Transformer Networks for Compact Spherical Convolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9442-9451, 2018.

[39] K. Tateno, N. Navab and F. Tombari, "Distortion-aware Convolutional Filters for Dense Prediction in Panoramic Images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 707-722, 2018.

[40] F. Wang, H. Hu, H. Cheng, J. Lin, S. Yang, M. Shih, H. Chu and M. Sun, "Self-supervised Learning of Depth and Camera Motion from 360° Videos," in *Asian Conference on Computer Vision*, pp. 53-68, 2018.

[41] N. Wang, B. Solarte, Y. Tsai, W. Chiu, and M. Sun, "360SD-Net: 360° Stereo Depth Estimation with Learnable Cost Volume," in *2020 IEEE International Conference on Robotics and Automation*, 2020.

[42] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price and A. Yuille, "Towards Unified Depth and Semantic Prediction from a Single Image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2800–2809, 2015.

[43] P. Wang, X. Shen, B. Russell, S. Cohen, B. Price and A. Yuille, "SURGE: Surface Regularized Geometry Estimation from a Single Image," in *Advances in Neural Information Processing Systems*, pp. 172–180, 2016.

[44] S. Xie, P. K. Lai, R. Laganière and J. Lang, "Effective Convolutional Neural Network Layers in Flow Estimation for Omni-Directional Images," in *2019 International Conference on 3D Vision (3DV)*, pp. 671-680, 2019.

[45] D. Xi, E. Ricci, W. Ouyang, X. Wang and N. Sebe, "Multi-Scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5354-5362, 2017.

[46] Z. Yang, P. Wang, W. Xu, L. Zhao and R. Nevatia, "Unsupervised Learning of Geometry From Videos With Edge-Aware Depth-Normal Consistency," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[47] F. Yu, V. Koltun and T. Funkhouser, "Dilated Residual Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 472-480, 2017.

[48] Z. Zhang, Y. Xu, J. Yu and S. Gao, "Saliency Detection in 360° Videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 488-503, 2018.

[49] N. Zioulis, A. Karakottas, D. Zarpalas and P. Daras, "Omnidepth: Dense depth estimation for indoors spherical panoramas," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 448-465, 2018.

[50] N. Zioulis, A. Karakottas, D. Zarpalas, F. Alvarez and P. Daras, "Spherical View Synthesis for Self-Supervised 360° Depth Estimation," in *2019 International Conference on 3D Vision (3DV)*, pp. 690-699, 2019.