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

Brandon Yushan Feng¹, Wangjue Yao¹, Zheyuan Liu², Amitabh Varshney¹
¹ University of Maryland, College Park
² University of Virginia
{yfeng97, wyao123, varshney}@umd.edu, zl4ej@virginia.edu

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.

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.
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 equiangular 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, with-
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. 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.

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

(1)

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

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

(2)

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

$$x = r \sin \phi \sin \theta \quad y = r \cos \theta \quad z = r \cos \phi \sin \theta$$

(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 $(x_j, y_j, z_j)$:

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

(4)

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

(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:

$$N_i = \{(x_j, y_j, z_j) \mid n_{ix}^T u_i > \alpha, |u_i - u_j| < \beta, |v_i - v_j| < \beta\}$$

(6)

where $(u_i, v_i)$ and $(u_j, v_j)$ are the 2D grid coordinates of pixels $P_i$ and $P_j$, $\beta$ is the parameter that controls the size of the spatial neighborhood, and $\alpha$ controls the size of the angular neighborhood. A larger value of $n_i^T n_i$ 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):

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

(7)

where $\theta_i$ and $\phi_i$ 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:

\[ 0 < r_{ij} < 255 \quad \max\left(\frac{r_{ij}}{r_i}, \frac{r_j}{r_{ij}}\right) < 10 \]  

(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 \( C(P_j) = 1 - q_j^2 \). 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 \( W(P_i, P_j) \), 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:

\[ r_i^N = \frac{\sum_{P_j \in N_i} C(P_j) \cdot W(P_i, P_j) \cdot r_{ij}}{\sum_{P_j \in N_i} C(P_j) \cdot W(P_i, P_j)} \]  

(9)

with \( C(P_j) = 1 - q_j^2 \) and \( W(P_i, P_j) = n_i^T n_j \).

Finally, the refined \( \hat{r}_i \) for \( P_i \) is calculated as:

\[ \hat{r}_i = C(P_i) \cdot r_i + (1 - C(P_i)) \cdot r_i^N \]  

(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 \( \psi \) as \( \lim_{R \to \infty} \sin(\psi) = \psi = \frac{d}{\sqrt{d^2 + \sqrt{d^2 + R^2}}} \).

The double quaternion representing this 4D rotation is:

\[ D = \cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\psi}{2}\right) \frac{d}{\sqrt{d^2 + \sqrt{d^2 + R^2}}} \]  

(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

\[ \mathbf{Q} = (0, n_x, n_y, n_z) \]  

(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

\[ \mathbf{G}_3 = \mathbf{G}_1 \mathbf{G}_2 \quad \mathbf{H}_3 = \mathbf{H}_1 \mathbf{H}_2 \]  

(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:

\[ \alpha = \cos^{-1}(\mathbf{G}_1 \cdot \mathbf{G}_2) \quad \beta = \cos^{-1}(\mathbf{H}_1 \cdot \mathbf{H}_2) \]  

(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:

\[ \mathbf{G} = \mathbf{DQ} \quad \mathbf{H} = \mathbf{D}^*\mathbf{Q} \]  

(15)

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

\[ L_{DQ} = \sqrt{\alpha_{DQ}^2 + \beta_{DQ}^2} \]  

(16)

where \(\alpha_{DQ}\) and \(\beta_{DQ}\) 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:

\[ \phi_R = \phi_L + b \frac{\cos(\phi_L)}{r_L \cdot \sin(\theta_L)} \quad \theta_R = \theta_L + b \frac{\sin(\phi_L)\cos(\theta_L)}{r_L} \]  

(17)
Following the procedure presented in Section 3.2, we combine depth and normal estimates from the stereo pair images into two double quaternions \((G^L, H^L)\) and \((G^R, H^R)\), from which we calculate the stereo loss:

\[
L_{\text{Stereo}} = \sqrt{\alpha_{\text{Stereo}}^2 + \beta_{\text{Stereo}}^2}
\]

where \(\alpha_{\text{Stereo}}\) and \(\beta_{\text{Stereo}}\) 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:

\[
L_{\text{total}} = L_{\text{berHu}} + L_{\text{DQ}} + L_{\text{Stereo}}
\]

Here, \(L_{\text{berHu}}\) 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
<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>RMLSE</th>
<th>AbsRel</th>
<th>SqRel</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>UResNet [49]</td>
<td>0.3374</td>
<td>0.1204</td>
<td>0.0835</td>
<td>0.0416</td>
<td>0.9319</td>
<td>0.9889</td>
<td>0.9968</td>
</tr>
<tr>
<td>RectNet [49]</td>
<td>0.2911</td>
<td>0.1017</td>
<td>0.0702</td>
<td>0.0297</td>
<td>0.9574</td>
<td>0.9933</td>
<td>0.9979</td>
</tr>
<tr>
<td>monoDepth [16]</td>
<td>7.2097</td>
<td>0.8200</td>
<td>0.4747</td>
<td>2.3783</td>
<td>0.2970</td>
<td>0.7900</td>
<td>0.7510</td>
</tr>
<tr>
<td>FCRN [21]</td>
<td>0.9410</td>
<td>0.3760</td>
<td>0.3181</td>
<td>0.4469</td>
<td>0.4922</td>
<td>0.7792</td>
<td>0.9150</td>
</tr>
<tr>
<td>DCRF [26]</td>
<td>1.1596</td>
<td>0.4400</td>
<td>0.4202</td>
<td>0.7597</td>
<td>0.3889</td>
<td>0.7044</td>
<td>0.8774</td>
</tr>
<tr>
<td>Ours</td>
<td>0.2373</td>
<td>0.0907</td>
<td>0.0859</td>
<td>0.0213</td>
<td>0.9690</td>
<td>0.9954</td>
<td>0.9988</td>
</tr>
</tbody>
</table>

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.

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.

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.

4. Experiments

We have trained and evaluated the performance of our method on the ODS dataset [20], which 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 $E_i = |y_i - \hat{y}_i|$, where $y_i$ is the ground truth depth and $\hat{y}_i$ is the predicted depth. $\delta_j$ 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:

\[
\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} E_i^2}{N}} \quad \text{Abs. Rel.:} \quad \frac{\sum_{i=1}^{N} E_i}{y_i} \\
\text{RMLSE} = \sqrt{\frac{\sum_{i=1}^{N} |\ln y_i - \ln \hat{y}_i|^2}{N}} \quad \text{Sq. Rel.:} \quad \frac{\sum_{i=1}^{N} E_i^2}{y_i^2} \\
\text{Log10}: \quad \frac{\sum_{i=1}^{N} |\log_{10} y_i - \log_{10} \hat{y}_i|}{N} \]
<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>RMLSE</th>
<th>Log10</th>
<th>AbsRel</th>
<th>SqRel</th>
<th>δ₁</th>
<th>δ₂</th>
<th>δ₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full model</td>
<td>0.3894</td>
<td>0.2572</td>
<td>0.0312</td>
<td>0.4130</td>
<td>0.9543</td>
<td>0.9836</td>
<td>0.9904</td>
<td></td>
</tr>
<tr>
<td>w/o $L_{DQ}$</td>
<td>0.4731</td>
<td>0.3452</td>
<td>0.0432</td>
<td>0.5830</td>
<td>0.9012</td>
<td>0.9257</td>
<td>0.9718</td>
<td>0.9880</td>
</tr>
<tr>
<td>w/o Refinement</td>
<td>0.4114</td>
<td>0.3190</td>
<td>0.0393</td>
<td>0.5535</td>
<td>0.9220</td>
<td>0.9313</td>
<td>0.9780</td>
<td>0.9903</td>
</tr>
<tr>
<td>w/o $L_{Stereo}$</td>
<td>0.3953</td>
<td>0.2622</td>
<td>0.0321</td>
<td>0.4562</td>
<td>0.7068</td>
<td>0.9530</td>
<td>0.9801</td>
<td>0.9904</td>
</tr>
</tbody>
</table>

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.

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


