1. Abstract

Due to the lack of space in the main paper, we provide more details of the proposed method and experimental results in the supplementary material. Sec. 2 adds more details of tangent projection. Sec. 3 provides the detailed calculation progress of the Collaborative Depth Distribution Classification (CDDC) module. Sec. 4 introduces our loss function, and Sec. 5 presents a detailed description of the used benchmark datasets and metrics. In the Sec. 6 and Sec. 7, we show additional comparison results and visual results about experiments. Furthermore, we discuss the rationality of SFA module in the Sec. 8 and show some comparison results on real data in the Sec. 9.

2. More Details of Tangent Projection

We start by introducing an example of the tangent projection (TP) [1]. As shown in Fig. 1, $P_s$ is a point on the sphere surface, $O$ is the center of the sphere, $P_c$ is the center of the tangent plane, and $P_t$ is the intersection point of the tangent plane and the extension line of $OP_s$. As both $P_s$ and $P_c$ are on the sphere surface, we represent their spherical coordinates as $(\theta_s, \phi_s)$ and $(\theta_c, \phi_c)$, respectively. Then, we can obtain the planar coordinate $(u_t, v_t)$ of the point $P_t$ on the tangent plane as follows:

$$u_t = \frac{\cos(\phi_s) \sin(\theta_s - \theta_c)}{\cos(c)},$$

$$v_t = \frac{\cos(\phi_s) \sin(\phi_s) - \sin(\phi_s) \cos(\theta_s - \theta_c)}{\cos(c)} - \frac{1}{\gamma} \sin(\phi_c) \sin(\phi_s) \cos(\theta_s - \theta_c).$$

(1)

And the inverse transformations are:

$$\theta_s = \theta_c + \tan^{-1}\left(\frac{u_t \sin(\sigma)}{\gamma \cos(\phi_s) \cos(\sigma) - v_t \sin(\phi_s) \sin(\sigma)}\right),$$

$$\phi_s = \sin^{-1}\left(\cos(\sigma) \sin(\phi_c) + \frac{1}{\gamma} \sin(\sigma) \cos(\phi_c)\right),$$

(2)

where $\gamma = \sqrt{u_t^2 + v_t^2}$ and $\sigma = \tan^{-1}\gamma$. With Eq.1 and Eq. 2, we can convert the points on the sphere and pixels in TP patches to each other. In addition, we can convert the spherical points into pixels in the ERP image with $(u_e, v_e) = \left(\frac{\theta_e + \phi_e}{\frac{w}{2\pi}}, \frac{\phi_e + \phi_e}{\frac{h}{\pi}}\right)$, where $w$ and $h$ are the width and height of the ERP image, respectively. Therefore, given the spherical coordinate of a TP patch center, we can achieve the mapping between the pixels in the ERP images and those in the corresponding TP patches.

The number of TP patches projected from a 360° spherical image depends on the sampling latitudes (the range of latitude is from -90° to 90°) and the sampling number at each latitude. For instance, in Omnifusion [12], TP patches are sampled...
Figure 1. An example of TP and ERP. Two TP patches are projected from two different areas (red area and yellow area).

Figure 2. (a) An ERP image; (b) TP patches with the patch number $N = 10$, which are sampled at three latitudes; (c) TP patches with $N = 18$, which are sampled at four latitudes; (d) TP patches with $N = 26$, which are sampled at five latitudes; (e) TP patches with $N = 46$, which are sampled at six latitudes.

Figure 3. TP patches with different patch sizes.

Figure 2, we can see that with the patch number increased, the area of the
overlapping regions increased correspondingly. As shown in Table. 2, too few patches can not provide sufficient regional structural information, while too many patches lead to the redundancy of regional information. As a result, we chose to use a relatively small patch number of 18.

We fix the patch FoV to $80^\circ$ and compare TP patches with different patch sizes of $32 \times 32$, $64 \times 64$, $128 \times 128$, and $256 \times 256$ in Fig. 3, and it demonstrates that different patch sizes do not affect the content in each TP patch, but a large patch size does produce TP patches with more details. However, as shown in Table. 3 of the main paper, too large patch size will increase computational costs and the redundancy of regional structural information (the amount of pixels in the overlapping regions), which may further influence the prediction from holistic contextual information and decrease the overall performance. As a result, we chose to use a relatively large patch size of $128 \times 128$.

For the patch FoV, we fix the patch size to $128 \times 128$, and change the patch FoVs to obtain a set of TP patches, as shown in Fig. 4. Compared with the complete view of Fig. 2a, too small FoV causes the loss of the scene information, while too large FoV causes the redundancy of information in the overlapping areas. As a result, we chose to use patch FoV $80^\circ$.

### 3. More Discussion of Feature Similarity

Let us consider two pixels in two different TP patches but corresponding to the same pixel in the ERP image (See Fig. 5). Using a look-up table, we can easily search the locations in two different TP patches $TP_1$ and $TP_2$ w.r.t. the pixel $A$. However, when re-projecting the pixel values in $TP_1$ and $TP_2$ to $A$ of ERP, we have to know which one is better (as done by OmniFusion), rendering the look-up table inapplicable for TP-to-ERP projection. We will clarify this accordingly.

### 4. More Details of Collaborative Depth Distribution Classification (CDDC)

In this section, we introduce the calculation process of the collaborative depth distribution classification (CDDC) module in detail.

First, given an ERP image with the size of $H_e \times W_e \times 3$, we follow the gnomonic projection to obtain $N$ TP patches with the size of $H_t \times W_t \times 3$. Through the feature extractors, we can obtain the ERP feature map $f^E$ with the size $H_e/2 \times W_e/2 \times C_e$ and TP feature maps $\{f^T_n\}, n = 1, \ldots, N$ with the size of $H_t/2 \times W_t/2 \times C_t \times N$, as the inputs of CDDC module. Then we summarize the detailed layer-by-layer network configurations in Table. 1. Especially, we introduce network configurations in four parts: holistic depth distribution classification, holistic depth prediction, regional depth distribution classification, and regional depth prediction.
In the holistic depth distribution classification, given the output of the transformer encoder, embedding tokens $T_{kout}[0]$, we select the first token $T_{kout}[0]$ to calculate the bin center vector $c^H$ as

$$c^H_i = D_{min} + \left(\frac{w^H_i}{2} + \sum_{j=1}^{i-1} w^H_j\right),$$

$$w^H_i = {D_{max} - D_{min} \over \sum_{b=1}^{B} (\text{mlp}(T_{kout}[0]))_i + \epsilon},$$

where $i, j = 1, \ldots, B, w^H$ is the bin widths of the holistic distribution histogram, $\text{mlp}$ denotes a multi-layer perceptron (MLP) head with a ReLU activation, $(D_{min}, D_{max})$ is the depth range of the dataset, $B$ denotes the number of depth distribution bins, and $\epsilon$ is a small constant to ensure that each value of $w^H$ is positive. For the holistic depth prediction, the bin centers $c^H$ are linearly blended with a probability score map $P^H$ to predict the depth value at each pixel $(i, j)$:

$$D^H(i, j) = \sum_{b=1}^{B} P^H(i, j) \cdot c^H_b.$$
the first dimension of the concatenation with the average operation, to obtain the tensor $T^R$. Then we combine $T^R$ with the spatial locations of index map $M$ to obtain a feature map $M_{key}$. Moreover, we introduce the embedding vectors $T^H$ of the ERP branch. With $M_{key}$ as the “keymap” and $T^H$ as the “queries”, we can predict the probability score map $P^R$ and further output the ERP format regional depth map $D^R$.

Figure 6. Overview of the CDDC module with two steps.

5. More Details of Loss Functions

As introduced in the main paper, our loss consists of two terms: the pixel-wise depth loss and the holistic distribution loss. For the pixel-wise depth loss, following existing works [9, 12], we adopt BerHu loss [10] for pixel-wise depth supervision, which is formulated as

$$L_{depth} = \sum_{i \in P} B(D^i - D_{GT}^i), \quad (7)$$

$$B(x) = \begin{cases} 
|x|, & |x| \leq c \\
\frac{x^2 + c^2}{2c}, & |x| > c
\end{cases} \quad (8)$$

where $D_{GT}$ is the ERP format ground truth, $P$ indicates pixels which are valid in the ground truth depth map. $c$ is a threshold hyper-parameter and set to 0.2 empirically [9, 12].

Furthermore, following [3], we employ the bi-directional Chamfer Loss [7] to encourage the holistic depth bin centers $c^H(b)$ to be consistent with the distribution of all depth values ($X$) in the ground truth map as:

$$L_{bin} = Cha(X, c^H(b)) \quad (9)$$

$$Cha(X_1, X_2) = \sum_{x \in X_1} \min_{y \in X_2} \|x - y\|^2 + \sum_{y \in X_2} \min_{x \in X_1} \|x - y\|^2 \quad (10)$$
Finally, the total loss is the summation of both two terms:

\[ L_{\text{total}} = L_{\text{depth}} + \lambda L_{H_{\text{bin}}}. \]  

(11)

For the balance weight \( \lambda \), we follow [3] and set \( \lambda = 0.1 \) for all our experiments.

6. More Details of Datasets and Metrics

We conduct experiments on three benchmark datasets: Stanford2D3D [2], Matterport3D [4] and 3D60 dataset [16]. Note that Stanford2D3D dataset and Matterport3D dataset are real-world datasets, while 3D60 dataset is composed of two synthetic datasets (SunCG [14] and SceneNet [8]) and two real-world datasets (Stanford2D3D and Matterport3D). Stanford2D3D contains 1413 panoramic samples and we split it into 1,000 samples for training, 40 samples for validation and 373 samples for testing. Matterport3D is the largest real-world dataset for indoor panorama scenes containing 10,800 panoramas and we follow the official split to split it into 33875 samples for training, 800 samples for validation, and 1298 samples for testing. As the largest 360° depth estimation dataset, 3D60 totally contains 35973 panoramic samples where 33875 of them are used for training, 800 samples for validation, and 1298 samples for testing. During training and testing, we resize the resolution of the panorama and depth map in the former two datasets into 512 × 1024. For 3D60, we set the input size into 256 × 512.

7. Additional Comparison Results

As shown in the open source code of PanoFormer [13], the authors applied the masking strategy for the Stanford2D3D dataset:

1. mask = torch.ones([512, 1024])
2. mask[0:int(512*0.15),:] = 0
3. mask[512-int(512*0.15):512,:] = 0

Therefore, we apply the same masking strategy for the Stanford2D3D dataset and compare with PanoFormer in Table 2. With the masking strategy, our HRDFuse outperforms PanoFormer [13] by a significant margin, e.g., 5.8% (Abs Rel), 11.3% (Sq Rel), 5.3% (RMSE). Furthermore, we compare our method with the PanoFormer on 3D60 dataset in Table 3. Note that PanoFormer did not provide the pre-trained models on the 3D60 dataset, we re-train the PanoFormer for 60 epochs with the official hyper-parameters and same experiment setting as OmniFusion and UniFuse. Our HRDFuse outperforms PanoFormer by a large margin.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Method</th>
<th>Patch size/FoV</th>
<th>Abs Rel ↓</th>
<th>Sq Rel ↓</th>
<th>RMSE ↓</th>
<th>RMSE(log) ↓</th>
<th>( \delta_1 ) ↑</th>
<th>( \delta_2 ) ↑</th>
<th>( \delta_3 ) ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stanford2D3D</td>
<td>PanoFormer* [13]</td>
<td>–/–</td>
<td>0.1131</td>
<td>0.0723</td>
<td>0.3557</td>
<td>0.2454</td>
<td>0.8808</td>
<td>0.9623</td>
<td>0.9855</td>
</tr>
<tr>
<td>Stanford2D3D</td>
<td>PanoFormer† [13]</td>
<td>–/–</td>
<td>0.0721</td>
<td>0.0506</td>
<td>0.3187</td>
<td>0.1949</td>
<td>0.9260</td>
<td>0.9766</td>
<td>0.9922</td>
</tr>
<tr>
<td>Stanford2D3D</td>
<td>HRDFuse*, Ours</td>
<td>128 × 128 / 80°</td>
<td>0.0984</td>
<td>0.0530</td>
<td>0.3452</td>
<td>0.1465</td>
<td>0.8941</td>
<td>0.9778</td>
<td>0.9923</td>
</tr>
<tr>
<td>Stanford2D3D</td>
<td>HRDFuse†, Ours</td>
<td>128 × 128 / 80°</td>
<td>0.0730</td>
<td>0.0469</td>
<td>0.3265</td>
<td>0.1311</td>
<td>0.9213</td>
<td>0.9807</td>
<td>0.9934</td>
</tr>
<tr>
<td>Stanford2D3D</td>
<td>HRDFuse*, Ours</td>
<td>256 × 256 / 80°</td>
<td>0.0935</td>
<td>0.0508</td>
<td>0.3106</td>
<td>0.1422</td>
<td>0.9140</td>
<td>0.9798</td>
<td>0.9927</td>
</tr>
<tr>
<td>Stanford2D3D</td>
<td>HRDFuse†, Ours</td>
<td>256 × 256 / 80°</td>
<td>0.0679</td>
<td>0.0449</td>
<td>0.3017</td>
<td>0.1271</td>
<td>0.9327</td>
<td>0.9826</td>
<td>0.9935</td>
</tr>
</tbody>
</table>

Table 2. Quantitative comparison with the SOTA methods. * represents that the model is re-trained following the official setting. † represents that the model is evaluated with the masking strategy in PanoFormer [13].

8. Additional Visual Results

More visual comparisons on Stanford2D3D and Matterport3D. In Fig. 7a, we perform qualitative comparisons with the SOTA methods, UniFuse. [9] and OmniFusion [12], on the Stanford2D3D dataset and Matterport3D dataset, whose samples are from real-world scenes. From the visual results, we confirm that our HRDFuse predicts the depth maps which are more precise and contain more structural details than other methods.

More visual comparisons on 3D60. In Fig. 7b, we perform qualitative comparisons with the SOTA methods, UniFuse. [9] and OmniFusion [12], on the 3D60 dataset, which contains both real-world and synthetic samples. From the visual results, we further confirm the superiority of our HRDFuse.
<table>
<thead>
<tr>
<th>Datasets</th>
<th>Method</th>
<th>Patch size/FoV</th>
<th>Abs Rel ↓</th>
<th>Sq Rel ↓</th>
<th>RMSE ↓</th>
<th>RMSE(log) ↓</th>
<th>( \delta_1 ^\uparrow )</th>
<th>( \delta_2 ^\uparrow )</th>
<th>( \delta_3 ^\uparrow )</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D60</td>
<td>FCRN [11]</td>
<td>–/–</td>
<td>0.0699</td>
<td>0.2833</td>
<td>-</td>
<td>-</td>
<td>0.9532</td>
<td>0.9905</td>
<td>0.9966</td>
</tr>
<tr>
<td></td>
<td>RectNet [16]</td>
<td>–/–</td>
<td>0.0702</td>
<td>0.0297</td>
<td>0.2911</td>
<td>0.1017</td>
<td>0.9574</td>
<td>0.9933</td>
<td>0.9979</td>
</tr>
<tr>
<td></td>
<td>Mapped Convolution [6]</td>
<td>–/–</td>
<td>0.0965</td>
<td>0.0371</td>
<td>0.2966</td>
<td>0.1413</td>
<td>0.9068</td>
<td>0.9854</td>
<td>0.9967</td>
</tr>
<tr>
<td></td>
<td>BiFuse with fusion [15]</td>
<td>–/–</td>
<td>0.0615</td>
<td>-</td>
<td>0.2440</td>
<td>-</td>
<td>0.9699</td>
<td>0.9927</td>
<td>0.9969</td>
</tr>
<tr>
<td></td>
<td>UniFuse with fusion [9]</td>
<td>–/–</td>
<td>0.0466</td>
<td>-</td>
<td>0.1968</td>
<td>-</td>
<td>0.9835</td>
<td>0.9965</td>
<td>0.9987</td>
</tr>
<tr>
<td></td>
<td>ODE-CNN [5]</td>
<td>–/–</td>
<td>0.0467</td>
<td>0.0124</td>
<td>0.1728</td>
<td>0.0793</td>
<td>0.9814</td>
<td>0.9967</td>
<td>0.9989</td>
</tr>
<tr>
<td></td>
<td>OmniFusion (1-iter) [12]</td>
<td>128 ( \times ) 128 / 80( ^\circ )</td>
<td>0.0469</td>
<td>0.0127</td>
<td>0.1880</td>
<td>0.0792</td>
<td>0.9827</td>
<td>0.9963</td>
<td>0.9988</td>
</tr>
<tr>
<td></td>
<td>OmniFusion (2-iter) [12]</td>
<td>128 ( \times ) 128 / 80( ^\circ )</td>
<td>0.0430</td>
<td>0.0114</td>
<td>0.1808</td>
<td>0.0735</td>
<td>0.9859</td>
<td>0.9969</td>
<td>0.9989</td>
</tr>
<tr>
<td></td>
<td>PanoFormer* [13]</td>
<td>–/–</td>
<td>0.0442</td>
<td>0.0124</td>
<td>0.1691</td>
<td>0.0676</td>
<td>0.9861</td>
<td>0.9966</td>
<td>0.9987</td>
</tr>
<tr>
<td></td>
<td>HRDFuse,Ours</td>
<td>128 ( \times ) 128 / 80( ^\circ )</td>
<td>0.0363</td>
<td>0.0103</td>
<td>0.1565</td>
<td>0.0594</td>
<td>0.9888</td>
<td>0.9974</td>
<td>0.9990</td>
</tr>
<tr>
<td></td>
<td>HRDFuse,Ours</td>
<td>256 ( \times ) 256 / 80( ^\circ )</td>
<td>\textbf{0.0358}</td>
<td>\textbf{0.0100}</td>
<td>\textbf{0.1555}</td>
<td>\textbf{0.0592}</td>
<td>\textbf{0.9894}</td>
<td>0.9973</td>
<td>\textbf{0.9990}</td>
</tr>
</tbody>
</table>

Table 3. Quantitative comparison with the SOTA methods. * represents that the model is re-trained following the official setting.

9. Discussion on the rationality of SFA module

As shown in the Fig. 8, for a scene with simple structure, our SFA module make the index map centralized to several representative TP patches with higher frequency of index, e.g., 4, 6, 10 (Fig. 8(c)). Combined with the Fig. 8(d), we can observe that the feature alignment based on the feature similarity in the SFA module tends to employ the most representative regional depth distributions to avoid the redundant usage. Meanwhile, facing the special depth values, SFA module will introduce the corresponding the regional depth distributions to predict them (e.g., with index 11, 14). Especially, with the scene structure becoming more complex, the more TP patches are needed to describe the holistic depth information, as shown in the Fig. 9. The frequency of TP index is more balanced.
(a) Visual comparisons on Stanford2D3D and Matterport3D.

(b) Visual comparisons on 3D60 dataset.

Figure 7. More visual comparison results.
Figure 8. The visualization of (a) ERP depth ground truth, (b) TP index map (colored according to the attached color card), (c) TP index frequency and (d) TP depth patches with the corresponding index numbers from a scene with simple structure.

Figure 9. The visualization of (a) ERP depth ground truth, (b) TP index map (colored according to the attached color card), (c) TP index frequency and (d) TP depth patches with the corresponding index numbers from a scene with complex structure.
10. Visual comparisons and discussion on real data.

To better compare the generation capability of our HRDFuse and other SoTA methods, we capture the two real images which records the indoor scene (considering the limited max depth value, we ignore the outdoor scene) and directly use the models trained on Matterport3D training dataset to predict their depth maps. As shown in the Fig. 10, we can observe that our HRDFuse predicts more precise depth maps for the captured scenes. By contrast, the results of PanoFormer [13] tend to be blurry and over-smooth on unseen scenes.

Figure 10. Visual comparisons on real data (captured by Ricoh Theta Z1).
References


