Technical Appendix

Technical Appendix contains experimental details and additional experimental results. The code of this paper and some additional analysis will be available at https://github.com/yuniw18/EGformer.

A. Experimental environment

A.1. Discussions on dataset

For experiments of the main paper, we use Structured3D (rgb rawlight) [20] and Pano3D (Matterport3D Train and Test /w Filmic High Resolution) [1] datasets for training and evaluation. Other datasets are not used for the following reasons:

PanosunCG [17] Due to license issue, PanoSunCG dataset is no longer publicly available.

3D60 [21] Recently, it has been pointed out in that the 3D60 dataset encounters an issue with its depth representation [14, 1]. The images in this dataset exhibit a correlation between their pixel brightness and depth, as illustrated in Figure A. Consequently, depth estimation networks can predict the depths solely by analyzing the pixel brightnesses of input 3D60 images. Therefore, 3D60 dataset is unsuitable for evaluating the performances of equirectangular depth estimation networks.

![Figure A: Pixel brightness issue of 3D60 dataset](image)

Matterport3D [3] Because Matterport3D is not natively available in equirectangular format, a stitching process is required to be employed as an equirectangular depth dataset. However, the resulting equirectangular format of Matterport3D varies depending on the stitching algorithm used. This circumstance makes it difficult to use Matterport3D as a benchmark for equirectangular depth estimation tasks. Moreover, Pano3D [1] already includes a rendered version of Matterport3D in its dataset. Hence, using both Pano3D and the Matterport3D for evaluation would be redundant.

Stanford2D3D [2] Figure B shows that the top and down parts of equirectangular images in Stanford dataset are not rendered properly. Since geometric structure is often crucial in enhancing performance of equirectangular depth estimation tasks [7], images with imperfect structure may negatively impact the performance and thus impede fair comparison.

![Figure B: Rendering issue of Stanford dataset](image)

A.2. Common training details

For training, official training split of Pano3D and Structured3D dataset is used. Because the depth values in Pano3D and Structured3D datasets vary in scale, we use scale-and-shift-invariant loss [13] as defined by Eq.(3) to properly train the network with multiple depth dataset [6, 4, 16, 13, 12, 19]. Here, $D$ indicates predicted depths of each method, $D_g$ represents ground truth depths and $D_A$ is aligned depths.

$$s, t = \arg \min_{s, t} (s \cdot D + t - D_g)$$
$$D_A = s \cdot D + t$$

$$L_{pix} = \frac{1}{n} \sum_{k=1}^{n} |(D_A - D_g)|$$

$$L_{grad} = \frac{1}{n} \sum_{k=1}^{n} |\nabla_x (D_A - D_g) + \nabla_y (D_A - D_g)|$$

$$L_{total} = L_{pix} + 0.5 \cdot L_{grad}$$

Because we use scale-and-shift invariant loss function, the output of all methods is post-processed equally to be in the range between $(0, 1)$ via a sigmoid function. Each
method is initialized with Structured3D dataset for 50 epochs with the learning rate of $5 \cdot 10^{-5}$. Continuously, each method is initialized with Pano3D + Structured3D dataset for 20 epochs additionally with the learning rate of $5 \cdot 10^{-5}$ and exponential learning rate decay rate of 0.95. AdamW [9] optimizer is used with the batch size of 1 for all methods.

A.3. Specific training details of each method

When we train each method under our experimental setup, we failed to train some methods (e.g. Panoformer, Yun et al.) when they are trained from scratch. Figure D shows a predicted depth of Panoformer which is trained from scratch. We conjecture that the deconvolution layers [10] in Panoformer cause failures of training if the weight of deconvolution layer is not properly initialized. Therefore, to train all methods as fair as possible, we utilize the pre-trained model in which each author provides in the open-source community (i.e. github) as follows:

**Panoformer [14]** We initialize the weight of convolution and deconvolution layers (except the transformer) in Panoformer network using the official pre-trained model in their github repository. Subsequently, the network is trained under the environment specified in common training details (Section A.2).

**EGformer** We first pre-trained the model from the scratch using Structured3D dataset for 50 epochs. Then, from the scratch again, we initialize the weight of convolution layers (except the transformer) in EGformer using that pre-trained model. Subsequently, the network is trained under the environment specified in common training details (Section A.2).

**Yun et al. [19]** We initialize the weight of convolution layers in Yun et al. (i.e. decoder part) using the official pre-trained model (trained using Structured3D) in their Github repository. Subsequently, the network is trained under the environment specified in common training details (Section A.2).

**Bifuse [18]** We initialize the weight of Bifuse network using the official pre-trained model (trained using Stanford2D3D) in their Github repository. Subsequently, the network is trained under the environment specified in common training details (Section A.2).

**SliceNet [11]** We initialize the weight of SliceNet network using the official pre-trained model (trained using Structured3D) in their Github repository. Then, the network is trained additionally for 20 epochs using Pano3D + Structured3D dataset.

---

1Training with Structured3D for 50 epochs is not executed here because we load the weights of ‘all’ layers in SliceNet using the official pre-trained model which is trained with Structured3D.
B. Network architecture

The overall network architecture of EGformer is described in Figure C, which is similar to Panoformer [14] network architecture. The channel size is set to $C = 32$ in our experimental setup. The yellow blocks are composed of several convolution, activation and normalization layers. Figure E shows examples of them. Further details on network architecture can be found in Code Appendix.

C. Additional experimental results

C.1. Further analysis on local and global attention

As discussed in various studies [15, 8], vision transformer (ViT) [5], in which Yun et al. [19] based on, requires large scale dataset due to lack of inductive bias. To alleviate the data insufficiency of equirectangular depth dataset, therefore, Yun et al. [19] performs transfer learning from the model which is trained with abundant depth dataset of typical 2D images in their paper. However, under our experimental setup, only equirectangular depth datasets are utilized. As discussed in the main paper, we speculate that this environment may impair the performances of Yun et al. significantly. Therefore, to see the pros and cons of global attention more clearly, we conduct an additional experiment. Table A shows the experimental results on Structured3D dataset. EGformer, Panoformer and Yun et al. in Table A is trained with experimental environment described in Section A.3. ‘Yun et al. + transfer’ in Table A represents the model which is trained via the following training environment: We initialize the weight of Yun et al. network using the official pre-trained model which is trained with depth datasets of typical 2D images + Structured3D dataset; then, we fine-tune the network via Structured3D dataset for 10 epochs with $5 \cdot 10^{-5}$ learning rate.

<table>
<thead>
<tr>
<th>Method</th>
<th>Abs. rel</th>
<th>Sq.rel</th>
<th>RMS.lin</th>
<th>$\delta^1$</th>
<th>#Param</th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yun et al. [19]</td>
<td>0.0505</td>
<td>0.0499</td>
<td>0.3475</td>
<td>0.9700</td>
<td>123.7M</td>
<td>589.4G</td>
</tr>
<tr>
<td>Panoformer [14]</td>
<td>0.0394</td>
<td>0.0346</td>
<td>0.2960</td>
<td>0.9781</td>
<td>20.4M</td>
<td>77.7G</td>
</tr>
<tr>
<td>EGformer</td>
<td>0.0342</td>
<td>0.0279</td>
<td>0.2756</td>
<td>0.9810</td>
<td>16.3M</td>
<td>73.9G</td>
</tr>
<tr>
<td>Yun et al. + transfer</td>
<td>0.0342</td>
<td>0.0282</td>
<td>0.2590</td>
<td>0.9842</td>
<td>123.7M</td>
<td>589.4G</td>
</tr>
</tbody>
</table>

Table A: Additional analysis of local and global attention on Structured3D testset. ‘Yun et al. + transfer’ represents the network which executes transfer learning from the model trained with abundant depth datasets of typical 2D images.

As shown in Table A, it is observed that ’Yun et al. + transfer’ yields the best depth estimation results overall which verifies that the number of dataset is critical for the global attention. However, it should be noted that local attention based studies (i.e. Panoformer, EGformer) yield comparable depth estimation results with significantly lower computational cost, fewer parameters and less datasets used as shown in Table A. These results clearly demonstrate that the local attention is much more ’efficient’ than global attention in equirectangular depth estimation tasks. Figure G shows the qualitative results of the methods in Table A. It is shown that EGformer yields comparable or better results than Yun et al. (+ transfer) in terms of details.

C.2. Additional qualitative results

Here, we present the additional qualitative results of each method described in the main paper as shown in Figure G.
Figure G: Additional qualitative results of each method.

References
