Entangled View-Epipolar Information Aggregation for Generalizable Neural Radiance Fields

Supplementary Material

1. Implementation Details

1.1. Generalizable 3D Representation $\boldsymbol{z}$

EVE-NeRF alternates the aggregation of epipolar and view dimensions' features through the VEI and EVI modules, resulting in the generation of generalizable 3D representation $\boldsymbol{z}$ that aligns with NeRF's coordinates. The pseudocode for the computation of $\boldsymbol{z}$ is as follows:

Algorithm 1: EVE-NeRF:PyTorch-like Pseudocode

Input: viewpoints difference $\Delta \boldsymbol{d}^s$, extracted convolution features $\boldsymbol{f}^c \in \mathbb{R}^{N\times M\times C}$, numbers of aggregation module N_{layer}
Output: generalizable 3D representation $\boldsymbol{z}$

$\boldsymbol{X} = \boldsymbol{f}^c$;
$i = 1$;
while $i \leq N_{layer}$ **do**
  $\boldsymbol{h} = \boldsymbol{X}$;
  $\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W_Q}, \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W_K}, \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W_V}$;
  $\boldsymbol{X} = \text{VEI}\left(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \Delta \boldsymbol{d}^s\right)$;
  $\boldsymbol{Mean}, \boldsymbol{Var} = \text{mean\&var}\left(\boldsymbol{V}, \text{dim} = 1\right)$;
  $\boldsymbol{w}^v = \text{sigmoid}\left(\text{AE}\left(\boldsymbol{Mean}, \boldsymbol{Var}\right)\right)$;
  $\boldsymbol{X} = \boldsymbol{X} \cdot \boldsymbol{w}^v$;
  $\boldsymbol{X}' = \boldsymbol{X}.\text{permute}\left(1, 0, 2\right)$;
  $\boldsymbol{Q}' = \boldsymbol{X}'\boldsymbol{W}'_{\boldsymbol{Q}}, \boldsymbol{K}' = \boldsymbol{X}'\boldsymbol{W}'_{\boldsymbol{K}}, \boldsymbol{V}' = \boldsymbol{X}'\boldsymbol{W}'_{\boldsymbol{V}}$;
  $\boldsymbol{X}' = \text{EVI}\left(\boldsymbol{Q}', \boldsymbol{K}', \boldsymbol{V}', \Delta \boldsymbol{d}^s\right)$;
  $\boldsymbol{Max} = \max\left(\boldsymbol{V}', \text{dim} = 1\right)$;
  $\boldsymbol{w}^e = \text{sigmoid}\left(\text{self-attn}\left(\boldsymbol{Max}\right)\right)$;
  $\boldsymbol{X}' = \boldsymbol{X}' \cdot \boldsymbol{w}^e$;
  $\boldsymbol{X} = \boldsymbol{X}'.\text{permute}\left(1, 0, 2\right) + \boldsymbol{h}$;
  $i = i + 1$;
end
$\boldsymbol{z} = \text{mean}(\boldsymbol{X}, \text{dim} = 1) \in \mathbb{R}^{N\times C}$;

1.2. Difference of Views $\Delta \boldsymbol{d}^s$

$\Delta \boldsymbol{d}^s$ serves as an additional input to attention computation in the view transformer and the epipolar transformer, allowing the model to learn more information about the differences in views. The pseudo-code for computing $\Delta \boldsymbol{d}^s$ is shown in Algorithm 2.

1.3. Difference of Camera Poses $\Delta \boldsymbol{pose}$

$\Delta \boldsymbol{pose}$ provides camera disparity information for multi-view calibration, which is merged with epipolar aggregation features to obtain geometry consistency prior. The pseudo-code to compute $\Delta \boldsymbol{pose}$ is shown in Algorithm 3.

Algorithm 2: $\Delta \boldsymbol{d}^s$:PyTorch-like Pseudocode

Input: the target ray direction $\boldsymbol{d}_t \in \mathbb{R}^3$, the source ray direction $\boldsymbol{d}_s \in \mathbb{R}^{M\times 3}$, the number of sampling points along the target ray N
Output: $\Delta \boldsymbol{d}^s$

$\boldsymbol{d}_t = \boldsymbol{d}_t.\text{unsqueeze}(0).\text{repeat}(M, 1)$;
$\boldsymbol{d}_{diff} = \boldsymbol{d}_t - \boldsymbol{d}_s$;
$\boldsymbol{d}_{diff} = \boldsymbol{d}_{diff}/\text{torch.norm}(\boldsymbol{d}_{diff}, \text{dim} = -1, \text{keepdim=True})$;
$\boldsymbol{d}_{dot} = \text{torch.sum}(\boldsymbol{d}_t * \boldsymbol{d}_s)$;
$\boldsymbol{\Delta d^s} = \text{torch.cat}([\boldsymbol{d}_{diff}, \boldsymbol{d}_{dot}], \text{dim} = -1)$;
$\boldsymbol{\Delta d^s} = \boldsymbol{\Delta d^s}.\text{unsqueeze}(0).\text{repeat}(N, 1, 1) \in \mathbb{R}^{N\times M\times 4}$;

Algorithm 3: $\Delta \boldsymbol{pose}$:PyTorch-like Pseudocode

Input: the target pose matrix $\boldsymbol{P}_t \in \mathbb{R}^{3\times 4}$, the source pose matrix $\boldsymbol{P}_s \in \mathbb{R}^{M\times 3\times 4}$
Output: $\Delta \boldsymbol{pose}$

$M = \boldsymbol{P}_s.shape[0]$;
$\boldsymbol{P}_t = \boldsymbol{P}_t.\text{unsqueeze(dim=0).repeat}(M, 1, 1)$;
$\boldsymbol{R}_t = \boldsymbol{P}_t[:, :3, :3]$;
$\boldsymbol{R}_s = \boldsymbol{P}_s[:, :3, :3]$;
$\boldsymbol{T}_t = \boldsymbol{P}_t[:, :3, -1]$;
$\boldsymbol{T}_s = \boldsymbol{P}_s[:, :3, -1]$;
$\Delta \boldsymbol{R} = \boldsymbol{R}_t @ \boldsymbol{R}_s^T.\text{view}(M, 9)$;
$\Delta \boldsymbol{T} = \boldsymbol{T}_t - \boldsymbol{T}_s^T$;
$\Delta \boldsymbol{pose} = \text{torch.cat}([\Delta \boldsymbol{R}, \Delta \boldsymbol{T}], \text{dim=-1}) \in \mathbb{R}^{M\times 12}$;

1.4. Additional Technical Details

EVE-NeRF network details. Our lightweight CNN consists of 4 convolutional layers with a kernel size of 3×3 and a stride of 1. BatchNorm layers and ReLU activation functions are applied between layers. The final output feature map has a dimension of 32. The VEI and EVI modules have 4 layers, which are connected alternately. Both the View Transformer and Epipolar Transformer have the same network structure, in which the dimension of hidden features is 64 and we use 4 heads for the self-attention module in transformer layers. For the transformer in Multi-View Calibration, the features dimension is 64 and head is 4, consisting of 1 blocks. For

Input	Layer	Output
input	Conv2d(3, 32, 3, 1)+BN+ReLU	conv0
conv0	Conv2d(32, 32, 3, 1)+BN+ReLU	conv1
conv1	Conv2d(32, 32, 3, 1)+BN	conv2_0
(conv0, conv2_0)	Add(conv0, conv2_0) + ReLU	conv2_1
conv2_1	Conv2d(32, 32, 3, 1)+BN+ReLU	**conv3**

Table 1. Network architecture of the lightweight CNN, where **conv3** is the output features. Conv2d(c_{in}, c_{out}, k, s) stands for a 2D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. BN stands for Batch Normalization Layer. ReLU stands for ReLU nonlinearity activation function. Add(x, y) means add x and y.

Input	Layer	Output
input	Conv1d(128, 64, 3, 1)+LN+ELU	conv1_0
conv1_0	MaxPool1d	conv1
conv1	Conv1d(64, 128, 3, 1)+LN+ELU	conv2_0
conv2_0	MaxPool1d	conv2
conv2	Conv1d(128, 128, 3, 1)+LN+ELU	conv3_0
conv3_0	MaxPool1d	conv3
conv3	TrpsConv1d(128, 128, 4, 2)+LN+ELU	x_0
[conv2;x_0]	TrpsConv1d(256, 64, 4, 2)+LN+ELU	x_1
[conv1;x_1]	TrpsConv1d(128, 32, 4, 2)+LN+ELU	x_2
[Input;x_2]	Conv1d(64, 64, 3, 1)+Sigmoid	**output**

Table 2. Network architecture of the 1D convolution AE. Conv2d(c_{in}, c_{out}, k, s) stands for a 1D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. LN stands for Layer Normalization Layer. ELU and Sigmoid stand for ELU and Sigmoid nonlinearity activation function separately. MaxPool1d is a 1D max pooling layer with a stride of 2. TrpsConv1d stands for transposed 1D convolution. $[\cdot;\cdot]$ means concatenation.

the AE network in Along-Epipolar Perception and the conditioned NeRF decoder are set the same as the experimental setups of GeoNeRF [7] and IBRNet [12], respectively. The network architectures of the lightweight CNN, the AE network, and the conditioned NeRF decoder are provided in Table 1, 2, and 3 respectively.

Naïve dual network details. To further validate the rationality of EVE-NeRF's dual-branch structure, in Sec 5.3, we compared our method with two naïve dual network architectures: the Naïve Dual Transformer and the Dual Transformer with Cross-Attention Interaction. The Naïve Dual Transformer's first branch is GNT [11], and the second branch is GNT with epipolar aggregation followed by view aggregation. The dual branch predicts colors of each branch via a tiny MLP network directly. And the final color is the average pooling of the two branch colors. GNT demonstrated that using volume rendering to calculate color values does not enhance GNT's performance. Hence, we consider it fair to compare EVE-NeRF with these two dual-branch networks. The Dual Transformer with Cross-Attention Interaction builds

Input	Layer	Output
$\boldsymbol{z}$	Linear(64, 128)	bias
$\gamma(\boldsymbol{p})$	Linear(63, 128)	x0_0
x0_0,bias	Mul(x0_0,bias)+ReLU	x0
x0	Linear(128, 128)	x1_0
x1_0,bias	Mul(x1_0,bias)+ReLU	x1
x1	Linear(128, 128)	x2_0
x2_0,bias	Mul(x2_0,bias)+ReLU	x2
x2	Linear(128, 128)	x3_0
x3_0,bias	Mul(x3_0,bias)+ReLU	x3
x3	Linear(128, 128)	x4_0
x4_0,bias	Mul(x4_0,bias)+ReLU	x4
[x4;$\gamma(\boldsymbol{p})$]	Linear(191, 128)	x5_0
x5_0,bias	Mul(x5_0,bias)+ReLU	x5
x5	Linear(128, 16)+ReLU	alpha_raw
alpha_raw	Mul(4, 16)	alpha0
alpha0	Linear(16,16)+ReLU	alpha1
alpha1	Linear(16,1)+ReLU	**alpha**
[x5;$\gamma(\boldsymbol{d})$]	Linear(191,64)+ReLU	x6
x6	Linear(64, 3)+Sigmoid	**rgb**

Table 3. Network architecture of the conditioned NeRF decoder. $\boldsymbol{z}$, $\boldsymbol{p}$, and $\boldsymbol{d}$ stand for the generalizable features, the coordinates of 3D sampling points, and the directions of rays, individually. γ stands for positional encoding in NeRF. Linear(c_{in}, c_{out}) stands for a linear layer with input channels c_{in} and output channels c_{out}. Mul stands for element-wise multiplication. MHA($head$, dim) stands for a multi-head-attention layer with the number of head $head$ and attention dimension dim. $[\cdot;\cdot]$ means concatenation.

upon the Naïve Dual Transformer by adding a cross-attention layer for inter-branch interaction. These dual network architectures are illustrated in Figure 1.

2. Multi-View Epipolar-Aligned Feature Extraction

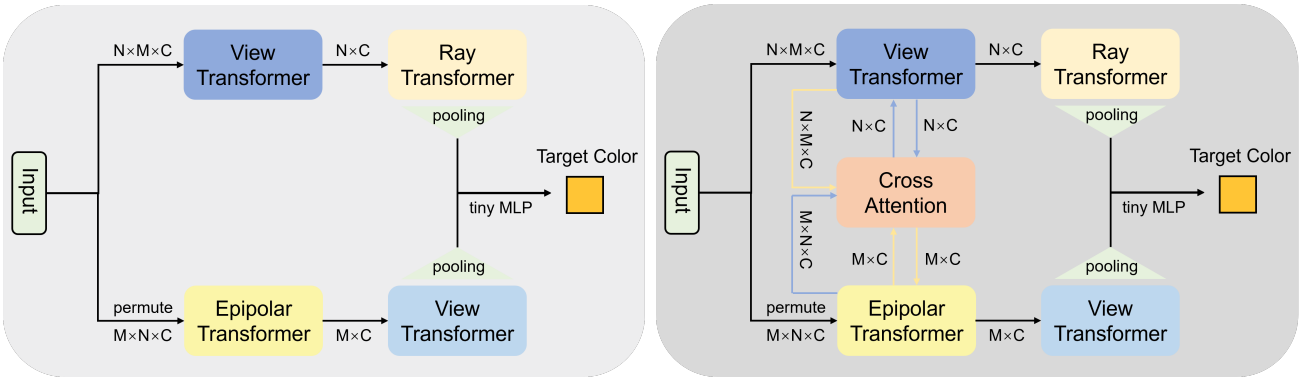
Let $\boldsymbol{K}_t$ and $\boldsymbol{P}_t = [\boldsymbol{R}_t, \boldsymbol{t}_t]$ represent the camera intrinsic and extrinsic parameters for the target view, and let $\boldsymbol{u}_t$ be the pixel coordinates corresponding to the target ray $\mathcal{R}$. In this case, $\mathcal{R}$ can be parameterized in the world coordinate system based on the delta parameter as follows:

$$\mathcal{R}(\delta) = \boldsymbol{t}_t + \delta \boldsymbol{R}_t \boldsymbol{K}_t^{-1} [\boldsymbol{u}_t^\top, 1]^\top. \tag{1}$$

Next, we sample N points $\{\boldsymbol{p}_i\}_{i=1}^N = \{\mathcal{R}(\delta_i)\}_{i=1}^N$ along $\mathcal{R}$ and project them onto the j-th source view:

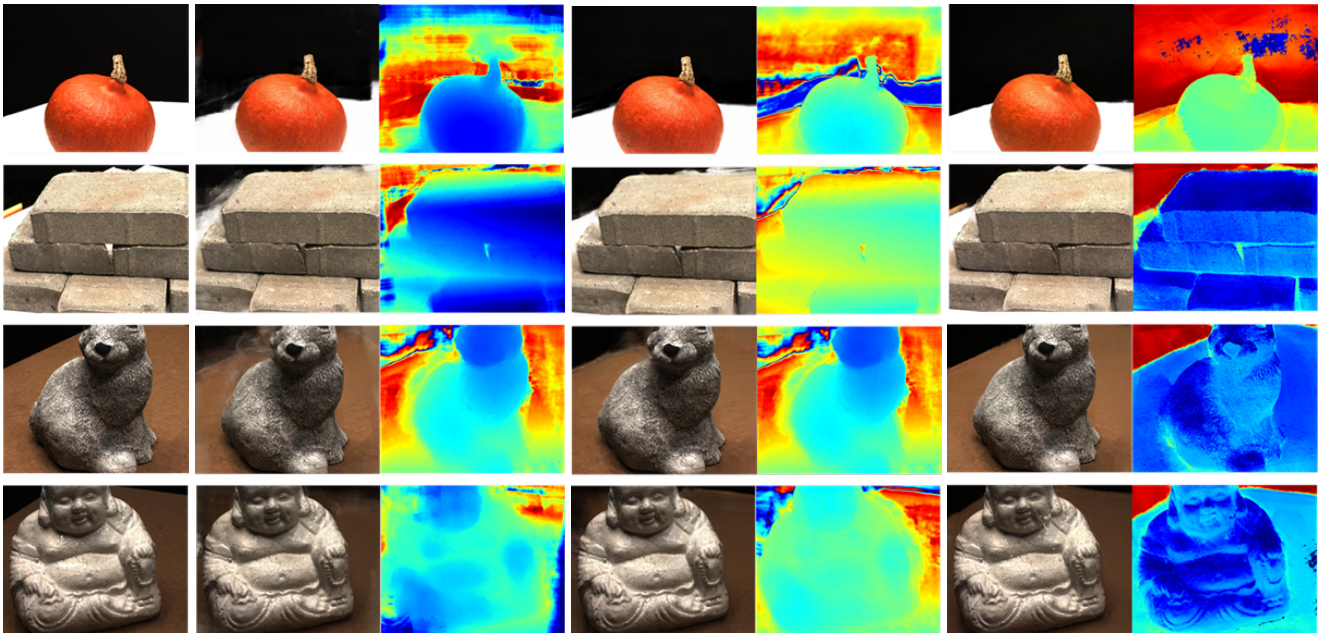
$$d_j^i [u_j^{i\top}, 1]^\top = \boldsymbol{K}_j \boldsymbol{R}_j^{-1} (\boldsymbol{p}_i - \boldsymbol{t}_j), \tag{2}$$

where u_j^i is the 2D coordinates of the i-th sampled point's projection onto the j-th source view, and d_j^i is the corresponding depth. Clearly, the projection points of these sampled points lie on the corresponding epipolar line in



(a) Naïve Dual Transformer (b) w/ Cross-Attention Interaction

Figure 1. Naïve dual network architecture. We design 2 baselines of dual networks for comparison: a) the Naïve Dual Transformer and b) the Dual Transformer with Cross-Attention Interaction. Table 4 demonstrates that our proposed method, EVE-NeRF, exhibits superior generalization capabilities for novel view synthesis.



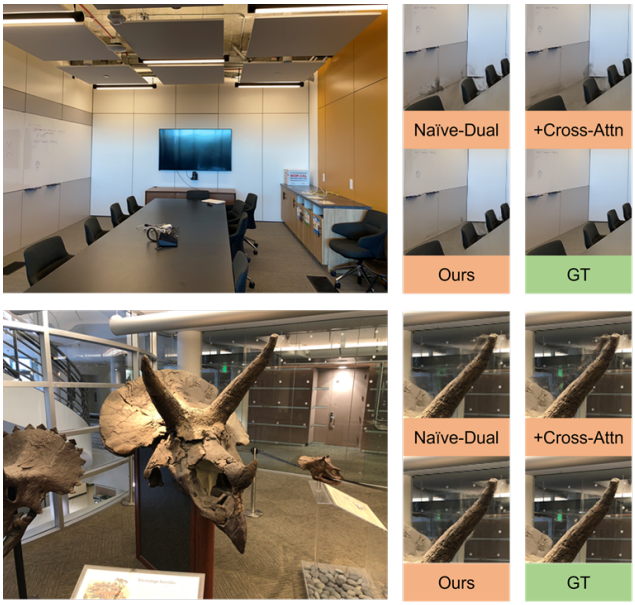
(a) Ground Truth (b) MVSNeRF (c) MatchNeRF (d) EVE-NeRF

Figure 2. Qualitative comparison of our generalizable GeoNeRF model with MVSNeRF [1] and MatchNeRF [3] in the few-shot setting. Our proposed method, EVE-NeRF, not only has higher rendering of new view pictures but also provides more accurate and detailed depth maps (without ground-truth depth supervision).This is due to the fact that EVE-NeRF provides accurate geometric and appearance a prior of multiple views for the model through the complementary structure of epipolar aggregation and view aggregation.

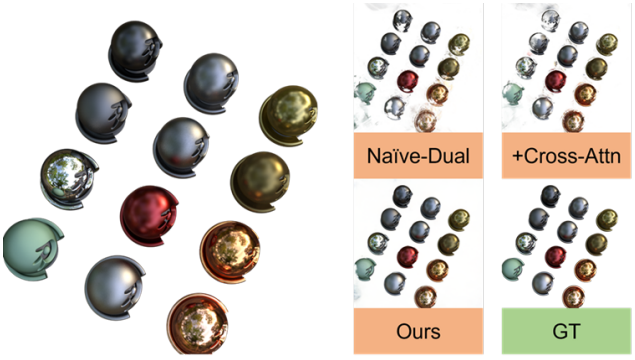
that view. Next, we obtain the convolution features $\boldsymbol{f}^c = \{\boldsymbol{f}^c_{i,j}\}^{N,M}_{i=1,j=1}$ in $\{\boldsymbol{F}^c_i\}^M_{i=1}$ for these projection points via bilinear interpolation. Therefore, for the target ray $\mathcal{R}$, we now have the multi-view convolution features $\boldsymbol{f}^c \in \mathbb{R}^{N\times M\times C}$ for $\mathcal{R}$, where C is the number of channels in the convolution features.

3. Feature Aggregation Network Proposed in Other Domains

Dual-branch network structures are commonly used in computer vision tasks [2, 5, 9, 13]. For instance, Simonyan [9] introduced a dual-stream network for action recognition in videos, consisting of a temporal stream for optical flow data and a spatial stream for RGB images, with the outputs from both branches being fused in the end. CrossVIT [2] is a vi-



(a) LLFF datasets



(b) Blender datasets

Figure 3. Qualitative comparison with naïve dual network architectures.

sual Transformer model based on dual branches, designed to enable the model to learn multi-scale feature information by processing different-sized image patches through the dual-branch network. DAT [5], on the other hand, is a transformer-based image super-resolution network that aggregates spatial and channel features through alternating spatial window self-attention and channel self-attention, enhancing representation capacity. Our approach does not follow the naive dual-branch structure. Instead, we introduce the along-epipolar perception and the multi-view calibration to compensate for the shortcomings in information interaction of the other branch. Besides, our dual-branch network demonstrates the efficient interplay between branches.

(a) Branch1 Rendering Results (b) Branch2 Rendering Results

Figure 4. Qualitative comparison with dual branches within the Naïve Dual Transformer

4. Additional Results

4.1. Qualitative Comparison for Setting 2

A qualitative comparison of our method with the few-shot generalizable neural rendering methods [1, 3] is shown in Figure 2. The novel view images rendered by our method produce minimal artifacts and can render the edge portion of the image and weakly textured regions. In addition, we generate a novel view depth map with 3 source views input through the volume rendering [8]. From Figure 2 we can observe that our generated depth map is more accurate and precise in terms of scene geometry prediction. This indicates that our proposed EVE-NeRF can extract high-quality aggregated features that imply the geometry and appearance of the scene, even in a few-shot setting.

4.2. Per-Scene Fine-Tuning Results

We fine-tune for $60,000$ iterations for each scene on the LLFF dataset. The quantitative comparison of our method with single-scene NeRF is demonstrated as shown in Table 4. We compare our method EVE-NeRF with NeRF [8], NeX [14], and NLF [10]. Our method outperforms baselines on the average metrics. The LPIPS of our method is lower than NLF by 13.4%, although NLF requires larger batchsize and longer iterations of training.

4.3. Qualitative Comparison With Naïve Dual Network Methods

As depicted in Figure 3a, we showcase a qualitative comparison of our approach with two other dual-branch methods on the Room and Horns scenes from the LLFF dataset. Our approach exhibits fewer artifacts and a more accurate geometric appearance. Specifically, in the Room scene, our method avoids the black floating artifacts seen in the chair and wall in the other two methods. In the Horns scene, our approach accurately reconstructs the sharp corners without causing ghosting effects. Figure 3b illustrates the qualitative comparison results in the Materials scene from the Blender dataset. It is evident that our method outperforms other dual-branch methods in rendering quality.

Models	Room	Fern	Leaves	Fortress	Orchids	Flower	T-Rex	Horns	Avg
NeRF [8]	32.70	25.17	20.92	31.16	20.36	27.40	26.80	27.45	26.50
NeX [14]	32.32	25.63	21.96	31.67	20.42	28.90	28.73	28.46	27.26
NLF [10]	**34.54**	24.86	22.47	**33.22**	21.05	**29.82**	**30.34**	29.78	28.26
EVE-NeRF	33.97	**25.73**	**23.78**	32.97	**21.27**	29.06	29.18	**30.53**	**28.31**

(a) PSNR↑

Models	Room	Fern	Leaves	Fortress	Orchids	Flower	T-Rex	Horns	Avg
NeRF [8]	0.948	0.792	0.690	0.881	0.641	0.827	0.880	0.828	0.811
NeX [14]	0.975	0.887	0.832	0.952	0.765	0.933	0.953	0.934	0.904
NLF [10]	**0.987**	0.886	0.856	**0.964**	**0.807**	**0.939**	**0.968**	0.957	0.921
EVE-NeRF	0.983	**0.894**	**0.891**	0.961	0.797	0.935	0.960	**0.961**	**0.923**

(b) SSIM↑

Models	Room	Fern	Leaves	Fortress	Orchids	Flower	T-Rex	Horns	Avg
NeRF [8]	0.178	0.280	0.316	0.171	0.321	0.219	0.249	0.268	0.250
NeX [14]	0.161	0.205	0.173	0.131	0.242	0.150	0.192	0.173	0.178
NLF [10]	0.104	**0.135**	**0.110**	0.119	**0.173**	0.107	0.143	0.121	0.127
EVE-NeRF	**0.060**	0.140	0.119	**0.089**	0.186	**0.103**	**0.095**	**0.086**	**0.110**

(c) LPIPS↓

Table 4. Single-scene fine-tuned comparison results for the LLFF dataset

While adding the cross-attention interaction mechanism can enhance the performance of generalizable novel view synthesis, it is apparent from Figure 3 that the rendered novel view images still exhibit artifacts and unnatural geometry. In some cases, the reconstruction quality of certain objects may even be inferior to the naïve dual transformer, as observed in the upper-left part of Figure 3b. This could be attributed to the limitation of the cross-attention interaction mechanism in aggregating features across both epipolar and view dimensions simultaneously.

Furthermore, we individually visualized the rendering results of each branch within the Naïve Dual Transformer, as depicted in Figure 4. It was observed that the second branch based on the epipolar transformer produced blurry rendering results. This is likely due to the absence of geometric priors, as interacting with epipolar information first can make it challenging for the model to acquire the geometry of objects. Therefore, aggregating view-epipolar feature naïvely may cause pattern conflict between view dimension and epipolar dimension. Instead of naïve feature aggregation, the dual network architecture of EVE-NeRF aims to compensate for the inadequacies in the first branch's interaction with information in the epipolar or view dimensions, providing the appearance continuity prior and the geometry consistency priors.

5. Limitation

Although our approach achieves superior performance in cross-scene novel view synthesis, it takes about 3 minutes to render a novel view image with a resolution of 1008×756, which is much longer than the vanilla scene-specific NeRF approach [4, 6, 8]. Nevertheless, we must admit that the simultaneous achievement of high-quality, real-time, and generalizable rendering poses a considerable challenge. In light of this, we posit that a potential avenue for further exploration is optimizing the speed of generalizable NeRF.

References

[1] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021. 3, 4

[2] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 357–366, 2021. 3

[3] Yuedong Chen, Haofei Xu, Qianyi Wu, Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Explicit correspondence matching for generalizable neural radiance fields. *arXiv preprint arXiv:2304.12294*, 2023. 3, 4

[4] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16569–16578, 2023. 5

[5] Zheng Chen, Yulun Zhang, Jinjin Gu, Linghe Kong, Xiaokang Yang, and Fisher Yu. Dual aggregation transformer for image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12312–12321, 2023. 3, 4

[6] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5875–5884, 2021. 5

[7] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18365–18375, 2022. 2

[8] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 4, 5

[9] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27, 2014. 3

[10] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8269–8279, 2022. 4, 5

[11] Mukund Varma, Peihao Wang, Xuxi Chen, Tianlong Chen, Subhashini Venugopalan, and Zhangyang Wang. Is attention all that nerf needs? In *The Eleventh International Conference on Learning Representations*, 2022. 2

[12] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 2

[13] Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Yu-Gang Jiang, Luowei Zhou, and Lu Yuan. Bevt: Bert pretraining of video transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14733–14743, 2022. 3

[14] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021. 4, 5