Vision Transformer for NeRF-Based View Synthesis from a Single Input Image
Supplementary Materials

1. Video Results

We include video results as a webpage (https://cseweb.ucsd.edu/~viscomp/projects/VisionNeRF/supplementary.html). In each video, we compare the input, SRN [11], PixelNeRF [13], ours and ground-truth. The renderings of SRN and PixelNeRF are provided by the authors of PixelNeRF on request.

2. Network Architecture

Transformer encoder. We adopt the pretrained ViT-b16 model from Wightman [12] as our transformer encoder. The transformer encoder has 12 layers ($J=12$), where each layer uses the LayerNorm [1] for normalization and GELU [4] for activation. The positional embedding is resized to the same size as the input image (e.g., $128 \times 128$ for the ShapeNet dataset, and $64 \times 64$ for the NMR dataset).

Convolutional decoder. We provide the architecture details of our convolutional decoder in Table 1. Our convolutional decoder takes the tokens $f^3$, $f^6$, $f^9$, and $f^{12}$ [9] as input and generate multi-level features $W_G^3$, $W_G^6$, $W_G^9$, $W_G^{12}$, as shown in Fig. 4 of the main paper. We then resize all feature maps $W_G$ to the size of $H/2 \times W/2$ via bilinear interpolation. Finally, we adopt a two-layer CNN module (see Table 2) to generate the global feature representation $W_G$.

2D CNN $G_L$. We use the ResBlocks in Fig. 1 to generate the local feature representation $W_L$. The local feature $W_L$ and global feature $W_G$ are then concatenated as our hybrid feature representation $W$.

NeRF MLP. We utilize the hierarchical rendering [8] to include detailed geometry and appearance. We use 6 ResNet blocks with width 512 for both coarse and fine stages to process the global and local features.

3. Implementation Details

Learning rate. We set the initial learning rate to be $10^{-4}$ for the MLP and $10^{-5}$ for ViT and the 2D CNN module.
Table 1. Architecture of the convolutional decoder. Conv denotes convolution layer. TransConv denotes transposed convolution layer.

<table>
<thead>
<tr>
<th>#</th>
<th>layer</th>
<th>kernel</th>
<th>stride</th>
<th>dilation</th>
<th>in</th>
<th>out</th>
<th>activation</th>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Conv0-0</td>
<td>1 × 1</td>
<td>1</td>
<td>1</td>
<td>768</td>
<td>96</td>
<td>N/A</td>
<td>f^3</td>
</tr>
<tr>
<td></td>
<td>TransConv0-1</td>
<td>4 × 4</td>
<td>4</td>
<td>1</td>
<td>96</td>
<td>96</td>
<td>N/A</td>
<td>Conv0-0</td>
</tr>
<tr>
<td></td>
<td>Conv0-2</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>96</td>
<td>512</td>
<td>N/A</td>
<td>TransConv0-1</td>
</tr>
<tr>
<td>6</td>
<td>Conv1-0</td>
<td>1 × 1</td>
<td>1</td>
<td>1</td>
<td>768</td>
<td>192</td>
<td>N/A</td>
<td>f^6</td>
</tr>
<tr>
<td></td>
<td>TransConv1-1</td>
<td>2 × 2</td>
<td>2</td>
<td>1</td>
<td>192</td>
<td>192</td>
<td>N/A</td>
<td>Conv1-0</td>
</tr>
<tr>
<td></td>
<td>Conv1-1</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>192</td>
<td>512</td>
<td>N/A</td>
<td>TransConv1-1</td>
</tr>
<tr>
<td>9</td>
<td>Conv2-0</td>
<td>1 × 1</td>
<td>1</td>
<td>1</td>
<td>768</td>
<td>384</td>
<td>N/A</td>
<td>f^9</td>
</tr>
<tr>
<td></td>
<td>Conv2-1</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>384</td>
<td>512</td>
<td>N/A</td>
<td>Conv2-0</td>
</tr>
<tr>
<td>12</td>
<td>Conv3-0</td>
<td>1 × 1</td>
<td>1</td>
<td>1</td>
<td>768</td>
<td>768</td>
<td>N/A</td>
<td>f^12</td>
</tr>
<tr>
<td></td>
<td>Conv3-1</td>
<td>3 × 3</td>
<td>2</td>
<td>1</td>
<td>768</td>
<td>768</td>
<td>N/A</td>
<td>Conv3-0</td>
</tr>
<tr>
<td></td>
<td>Conv3-2</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>768</td>
<td>512</td>
<td>N/A</td>
<td>Conv3-1</td>
</tr>
</tbody>
</table>

Table 2. Architecture of the last two layers for generating global feature representation $W_{G}$.

<table>
<thead>
<tr>
<th>Layer</th>
<th>kernel</th>
<th>stride</th>
<th>dilation</th>
<th>in</th>
<th>out</th>
<th>activation</th>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv0</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>1024</td>
<td>512</td>
<td>ReLU</td>
<td>$W_{G}$</td>
</tr>
<tr>
<td>Conv1</td>
<td>3 × 3</td>
<td>1</td>
<td>1</td>
<td>512</td>
<td>256</td>
<td>ReLU</td>
<td>conv0</td>
</tr>
</tbody>
</table>

To improve training stability, we use a warm-up schedule to increase the learning rate linearly from 0 for the first 10k steps. We decay the learning rate by a factor of 0.1 at 450k steps. Our learning rate schedule for the NeRF MLP is plotted in Fig. 2.

Inference speed. We ran 300 forward steps and averaged them to obtain the per-step inference time. PixelNeRF takes 1.35s and ours takes 1.7s. Our method has slightly more latency.

4. Experiment Configurations

We provide the detailed experiment setups in Sec. 4 of the main paper.


SRN [11] and PixelNeRF [13]: We obtain the PSNR and SSIM from Table 2 in [13]. We use the pre-generated results provided by the authors of [13] (on request) to calculate the LPIPS score.

CodeNeRF [5]: We obtain the PSNR and SSIM of the unposed input from Table 2 in [5]. As the pre-generated results are not available, and the source code of CodeNeRF does not provide optimization stages for unposed inputs, we are not able to calculate the LPIPS score.

CodeNeRF [5]: We obtain the PSNR and SSIM of the unposed input from Table 2 in [5]. As the pre-generated results are not available, and the source code of CodeNeRF does not provide optimization stages for unposed inputs, we are not able to calculate the LPIPS score.

4.2. Category-agnostic View Synthesis

Similar to category-specific view synthesis, we obtain the numbers of SRN [11] and PixelNeRF [13] from Table 4 of [13]. We obtain the numbers of FE-NVS [3] from Table 4 of [3]. Qualitative results for SRN and PixelNeRF are generated using the pre-generated results from [13]. The detailed categorical numbers of SRT [10] and FWD [2] are provided by their authors. For qualitative results, we obtain the images from SRT project website and from the author of FWD. Note that SRT does not provide the full renderings of the dataset.

4.3. View Synthesis on Real Images

We use the real car images provided by PixelNeRF and the Stanford Cars dataset [7] for evaluation. We remove the background using the PointRend [6] segmentation and resize the images to $128 \times 128$ resolution. We assume the input camera pose is an identity matrix and apply rotation matrices to simulate $360^\circ$ views. We use the source code and pre-trained model of PixelNeRF to generate the results for PixelNeRF, where we synthesize the renderings at the same camera poses.

5. Additional Results

5.1. Generalization on Unseen objects.

For unseen categories, we feed an image of a real mug to the category-agnostic model as shown in Fig. 3. While the training data do not include mugs, our method is able to predict reasonable novel views.
5.2. Results on Category-specific and Category-agnostic View Synthesis

We include extra results on category-specific view synthesis in Fig. 4 and 5, results on category-agnostic view synthesis in Fig. 6-12. Note that we choose the target views where most pixels are not visible in the input view to better compare the rendering quality of each method on occluded regions. The video results (in GIF format) are also provided on website (https://cseweb.ucsd.edu/~viscomp/projects/VisionNeRF/supplementary.html).

References

Figure 4. Category-specific view synthesis results on cars.
Figure 5. Category-specific view synthesis results on chairs.
Figure 6. Category-agnostic view synthesis results on the NMR dataset.
Figure 7. Category-agnostic view synthesis results on the NMR dataset.
<table>
<thead>
<tr>
<th>Chair</th>
<th>Input</th>
<th>SRN</th>
<th>PixelNeRF</th>
<th>FWD</th>
<th>Ours</th>
<th>GT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 8.** Category-agnostic view synthesis results on the NMR dataset.
Figure 9. Category-agnostic view synthesis results on the NMR dataset.
Figure 10. Category-agnostic view synthesis results on the NMR dataset.
Figure 11. Category-agnostic view synthesis results on the NMR dataset.
Figure 12. Category-agnostic view synthesis results on the NMR dataset.