

ResVR: Joint Rescaling and Viewport Rendering of Omnidirectional Images

Supplementary Material

Anonymous Author(s)

Our main paper has outlined the core techniques of our proposed **ResVR** method for the joint Rescaling and Viewport Rendering of ODIs. It has also demonstrated the efficacy of our methodological contributions through experiments. This appendix offers further details on our ResVR in Sec. A, along with additional experimental results and analyses in Sec. B, which are not included in the main paper due to space constraints.

A MORE DETAILS OF PROPOSED METHOD

A.1 Preliminaries of Perspective projection and viewport rendering

Perspective projection. Perspective projection is the gnomonic projection [4] of a sphere surface onto a flat, rectangular plane surface. We first show the transformation between spherical coordinates and Cartesian coordinates. Any point on the unit sphere can be expressed in spherical coordinates with longitude $\phi \in [-\pi, \pi]$ and latitude $\theta \in [-\pi/2, \pi/2]$. We also define the corresponding Cartesian coordinate (X, Y, Z) . The transformation between spherical coordinate and Cartesian coordinate can be formulated as:

$$X = \cos \theta \sin \phi, Y = \sin \theta, Z = \cos \theta \cos \phi. \quad (1)$$

The corresponding inverse transformation is expressed as:

$$\phi = \tan^{-1} \frac{X}{Z}, \theta = \tan^{-1} \frac{Y}{\sqrt{X^2 + Z^2}}. \quad (2)$$

Here we consider the spherical coordinates of the perspective plane center to be $(\theta_c, \phi_c) = (0, 0)$ without loss of generality. The forward projection $F_{pp} : (X, Y, Z) \mapsto (x_p, y_p)$, which maps any point of the sphere to the front viewport plane is defined as [2]:

$$x_p = \frac{X}{Z}, y_p = \frac{Y}{Z}, \quad (3)$$

and the backward projection $F_{pp}^{-1} : (x_p, y_p) \mapsto (X, Y, Z)$ is formulated as:

$$X = qx_p, Y = qy_p, Z = q, \quad (4)$$

with

$$q = \frac{1 + \sqrt{x_p^2 + y_p^2 + 1}}{x_p^2 + y_p^2 + 1}. \quad (5)$$

Viewport rendering. Given the horizontal FoV F_h and the vertical FoV F_v , a viewport is actually a 2D image that is rendered through perspective projection from a region of the sphere. Denote the height and width of the viewport as h_v and w_v , respectively. Given the viewport pixel coordinates (m, n) , the coordinates of the viewport image plane (x_p, y_p) can be obtained by the following equations [3]:

$$x_p = 2 \tan \left(\frac{F_h}{2} \right) \left(\frac{m + 0.5}{w_v} - \frac{1}{2} \right), y_p = 2 \tan \left(\frac{F_v}{2} \right) \left(\frac{1}{2} - \frac{n + 0.5}{h_v} \right). \quad (6)$$

The required Cartesian coordinates (X, Y, Z) of the sphere can be obtained by Eqs. (4) and (5). Note that Eqs. (3)-(6) are under the assumption that the center of the viewport is on the Z-axis, i.e. $(\theta_c, \phi_c) = (0, 0)$. For any viewing direction (θ_c, ϕ_c) , the corresponding sphere coordinates (X', Y', Z') can be obtained by:

$$(X', Y', Z')^\top = R(X, Y, Z)^\top, \quad (7)$$

where R is the viewport rotation matrix defined by (θ_c, ϕ_c) .

Stage	Building Block	Output Size
Input Downsample	PixelUnshuffle 4×	$H/4 \times W/4 \times 64$
	$3 \times 3, 64$	
	LeakyReLU	
Downsampling Block1	ResidualDenseBlock-32	$H/8 \times W/8 \times 128$
	$\begin{bmatrix} 3 \times 3, 128 \\ \text{LeakyReLU} \end{bmatrix} \times 2$	
	ResidualDenseBlock-64	
Downsampling Block2	$\begin{bmatrix} 3 \times 3, 256 \\ \text{LeakyReLU} \end{bmatrix} \times 2$	$H/16 \times W/16 \times 256$
	ResidualDenseBlock-128	
	$3 \times 3, 512$	
Upsampling Block1	$\begin{bmatrix} 3 \times 3, 128 \\ \text{LeakyReLU} \end{bmatrix} \times 2$	$H/16 \times W/16 \times 128$
	ResidualDenseBlock-128	
	$3 \times 3, 256$	
Upsampling Block2	$\begin{bmatrix} 3 \times 3, 64 \\ \text{LeakyReLU} \end{bmatrix} \times 2$	$H/8 \times W/8 \times 64$
	ResidualDenseBlock-64	
	$3 \times 3, 3$	
Output layer	$3 \times 3, 3$	$H/4 \times W/4 \times 3$

Table 1: Architectural details of our downsampler.

A.2 Implementation details

Implementation details of ODI downscaling: Given an HR ERP image $\mathbf{I}^{\text{HR-ERP}} \in \mathbb{R}^{3 \times H \times W}$, an LR representation is firstly generated through our downsampler, where s is the rescaling factor. The downsampler is a U-Net [6] with dense blocks [1]. The details of our downsampler are shown in Tab. 1. "PixelUnshuffle 4×" stands for the rearrangement of elements [7] which downsamples the HR image by a factor of 4. " $3 \times 3, 64$, LeakyReLU" represents a 2D-convolution operation with a kernel size 3, output channel number

64, followed by a LeakyReLU operation. We follow [1] to use the implementation of the residual dense block, and "ResidualDenseBlock 32" refers to one residual dense block with a minimum channel 32.

Implementation details of JPEG compression: To further decrease the file size of transmitted LR ERP, according to JPEG algorithm, $\tilde{\mathbf{I}}^{\text{LR-ERP}}$ is firstly converted to the luma-chroma color space (YCbCr). Then the converted image is split into 8×8 pixel blocks as $\tilde{\mathbf{I}}^{\text{LR-ERP}} \in \mathbb{R}^{3 \times N \times 8 \times 8}$, where $N = \frac{H \times W}{64 \times s^2}$. Then we transform $\tilde{\mathbf{I}}^{\text{LR-ERP}}$ to Discrete-Cosine-Transform (DCT) coefficients $C \in \mathbb{R}^{3 \times \frac{H \times W}{64 \times s^2} \times 8 \times 8} = (C_Y, C_{Cb}, C_{Cr})$. Inspired by [5], the quantization tables Q for each jpeg image are predicted with a quantization prediction module (QPM). The QPM is implemented as two separate 8-layer multilayer perceptrons (MLPs), composed of the luma predictor MLP_L and MLP_C . For block $C_k = (C_{Y,k}, C_{C,k})$, we vectorize C_k into a 1D vector and produce its quantization table with QPM. Thus, we have

$$Q_L = \frac{\sum_k \text{MLP}_L(C_{Y,k})}{|C_{Y,k}|}, Q_C = \frac{\sum_k \text{MLP}_C(C_{C,k})}{|C_{C,k}|}, \quad (8)$$

where $|\cdot|$ calculates the number of blocks. The average Q is used as a quantization table for the whole image to facilitate the conventional JPEG codec. Then, the luma coefficient C_Y and chroma coefficient $C_C = (C_{Cb}, C_{Cr})$ are quantized separately by predicted Q_L and Q_C :

$$\widetilde{C}_Y = \frac{C_Y}{[Q_L]}, \widetilde{C}_C = \frac{C_C}{[Q_C]}, \quad (9)$$

where $[\cdot]$ represents the rounding and truncation function. Note that the quantization tables $[Q] = ([Q_L], [Q_C])$ and quantized DCT coefficients $[\widetilde{C}] = ([\widetilde{C}_Y], [\widetilde{C}_C])$ are also encoded into the JPEG file.

Following the training stage of learned compression, the additive uniform noise ϵ where each element follows $\mathcal{U}(-\frac{1}{2}, \frac{1}{2})$ is adopted to approximate the non-differentiable quantization noise in Eq. (9):

$$\widetilde{C}_Y = \frac{C_Y}{Q_L + \epsilon} + \epsilon, \widetilde{C}_C = \frac{C_C}{Q_C + \epsilon} + \epsilon, \quad (10)$$

In the testing stage, the standard quantization function in Eq. (9) is switched back to fit the JPEG API. In a practical scenario, the JPEG image facilitates faster transmission and costs less storage.

Implementation details of JPEG decompression: After receiving the JPEG ERP image, the JPEG decoder first extracts $[Q]$, $[\widetilde{C}]$ from the JPEG file for image reconstruction with inverted operations of encoding steps:

$$\widehat{C}_Y = [\widetilde{C}_Y][Q_L], \widehat{C}_C = [\widetilde{C}_C][Q_C], \quad (11)$$

and $\widehat{\mathbf{I}}^{\text{LR-ERP}} = \text{IDCT}([\widehat{C}_Y, \widehat{C}_C])$, where IDCT is the inverse operation of DCT.

Implementation details of VR module: VR module consists of an encoder \mathcal{E} , a local texture estimator h_ψ , and an MLP-based decoder \mathcal{D} . The architecture of the encoder \mathcal{E} is shown in Tab. 2.

The local texture estimator contains an amplitude estimator h_a ($\mathbb{R}^{24} \mapsto \mathbb{R}^{256}$), a frequency estimator h_f ($\mathbb{R}^{24} \mapsto \mathbb{R}^{2 \times 128}$), and a phase estimator h_p ($\mathbb{R}^{10} \mapsto \mathbb{R}^{128}$). h_a and h_f are implemented by 3×3 convolution layer with 256 channels, and h_p is a single linear layer with 128 channels. The decoder \mathcal{D} is a 4-layer MLP with ReLU activations, and its hidden dimension is 256.

Encoder \mathcal{E}	Building Block	Output Size
Input Convolution	$3 \times 3, 24$	$H/4 \times W/4 \times 24$
Residual Convolution Block	$\begin{bmatrix} 3 \times 3, 24 \\ 3 \times 3, 24 \\ \text{ReLU} \end{bmatrix} \times 16$	$H/4 \times W/4 \times 24$

Table 2: Architectural details of the encoder of VR module.



Figure 1: Visualization of an original ERP image and its ten rendered viewports for evaluation.

A.3 Evaluation Benchmark

For evaluation of the quality of rendered viewports, we choose ten different view directions, get the ground truth image using bicubic interpolation, and calculate the average PSNR, SSIM, and LPIPS [8]. Specifically, the view directions $\{(\theta, \phi)\}$ are set to $\{(0^\circ, 0^\circ), (0^\circ, 90^\circ), (0^\circ, 180^\circ), (30^\circ, 0^\circ), (30^\circ, 90^\circ), (30^\circ, 180^\circ), (45^\circ, 0^\circ), (45^\circ, 90^\circ), (45^\circ, 180^\circ), (-90^\circ, 0^\circ)\}$. An example is shown in Fig. 1.

B MORE EXPERIMENTS AND ANALYSES

B.1 Visualization of sampled pixels

Fig. 2 presents the visualization of sampled regions and pixels in three examples. Specifically, the left part of Fig. 2 shows randomly cropped $\mathbf{I}^{\text{HR-ERP-Patch}}$ (red rectangular), corresponding X_{view} (blue points), and randomly sampled X_{samp} (green points). The right part of Fig. 2 shows some of the sampled pixels through bicubic interpolation, which is reshaped to 96×96 for better visualization.

B.2 Visualization of LR JPEG images

The file size and visual quality of LR JPEG images are also important for efficient transmission and user preview. We provide some visualizations of transmitted JPEG LR ERP images in Fig. 3, together with their bpp and WS-PSNR. It can be seen that ResVR generates LR images without compromising HR reconstruction quality.

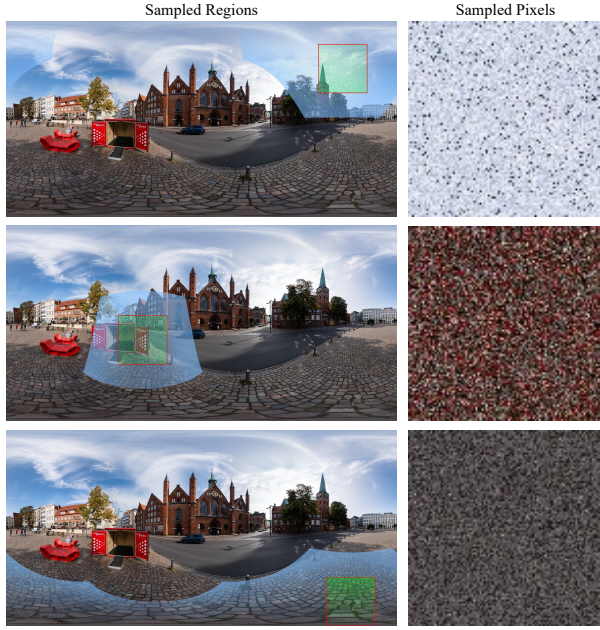


Figure 2: Visualization of sampled regions and pixels through our proposed discrete pixel sampling strategy in Sec. 3.3 of our main paper. **Left:** The red rectangular denotes the area of cropped patch $I^{\text{HR-ERP-Patch}}$, the blue points denote the elements in X_{view} , and the green points denote some elements in X_{samp} . **Right:** Corresponding sampled pixels in S_{pix} , which are reshaped to 96×96 for better visualization.

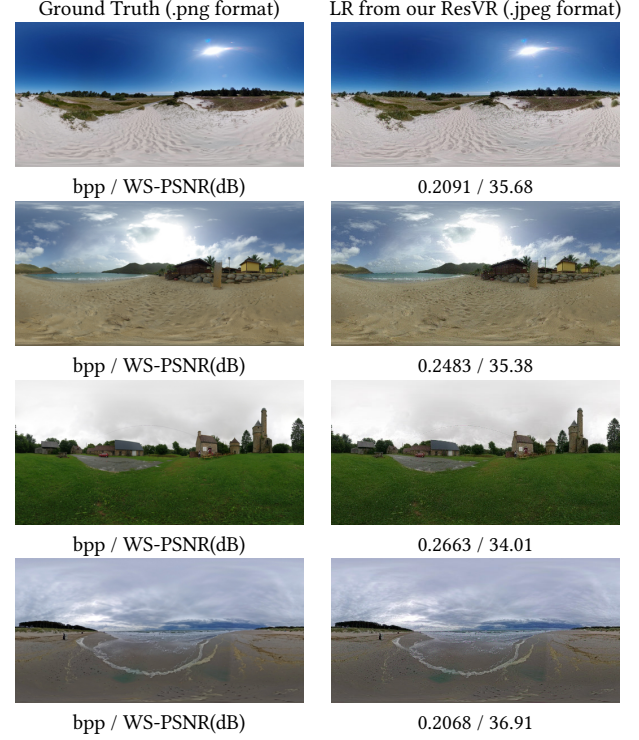


Figure 3: Visualizations of the ground truth images and corresponding LR JPEG ones downscaled by our ResVR model.

REFERENCES

[1] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4700–4708.

[2] Falah Jabar, João Ascenso, and Maria Paula Queluz. 2017. Perceptual analysis of perspective projection for viewport rendering in 360° images. In *2017 IEEE International Symposium on Multimedia (ISM)*. IEEE, 53–60.

[3] Falah Jabar, Joao Ascenso, and Maria Paula Queluz. 2019. Objective assessment of perceived geometric distortions in viewport rendering of 360° images. *IEEE Journal of Selected Topics in Signal Processing* 14, 1 (2019), 49–63.

[4] Max D. Larsen and James L. Fejfar. 1974. *INTRODUCTION TO GEOMETRY*. 225.

[5] Chenyang Qi, Xin Yang, Ka Leong Cheng, Ying-Cong Chen, and Qifeng Chen. 2023. Real-time 6K Image Rescaling with Rate-distortion Optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14092–14101.

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5–9, 2015, proceedings, part III 18*. Springer, 234–241.

[7] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1874–1883.

[8] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.