UniTriSplat: A Unified 3D Gaussian Splatting Framework with Uniform Spherical Rasterization for Universal Cameras

— Supplementary Material —

7 Supplementary Experimental Details

This section provides additional details of the datasets and evaluation protocols used in the multi-FoV experiments. We first summarize the benchmark coverage, then describe the common HEALPix-based computation of HSSIM and the treatment of stitching artifacts in omnidirectional images.

7.1 Datasets and Evaluation Coverage

The selected datasets cover perspective, fisheye, and omnidirectional imaging, with FoVs ranging from conventional perspective views to complete omnidirectional coverage. They also combine real and synthetic scenes, indoor and outdoor environments, and different image resolutions. This diversity allows us to evaluate whether the same spherical rasterization framework remains effective under changes in camera geometry, scene content, and acquisition conditions. Mip-NeRF 360 contains both bounded indoor scenes and unbounded outdoor scenes. ScanNet++ provides calibrated indoor fisheye images, whereas FIORD contains challenging ultra-wide FoV indoor and outdoor scenes. Ricoh360, 360Roam, and OmniBlender complement these datasets with real outdoor omnidirectional scenes, high-resolution real indoor omnidirectional scenes, and synthetic omnidirectional scenes, respectively. Table 4 shows the summary of the datasets we use in the multi-FoV evaluation.

7.2 Fair HSSIM Evaluation

HSSIM serves two distinct roles in our framework: it is used as a spherical structural loss for UniTriSplat and as a common evaluation metric for all methods. Although optimizing the HSSIM loss improves UniTriSplat under this metric, the baselines similarly optimize standard SSIM in their native image domains. Moreover, the variant without the HSSIM loss still improves standard SSIM over the baselines on many scenes, indicating that the reconstruction gains do not solely result from optimizing the evaluation metric. HSSIM is included to measure structural fidelity on uniformly sampled spherical geometry, where planar SSIM is affected by projection-dependent pixel density.

For a camera model $\mathcal{C}$, let $\Pi_{\mathcal{C}}^{-1} : \mathbb{S}^2 \to \mathbb{R}^2$ map a spherical direction to its image coordinate. Each HEALPix pixel q corresponds to a unit direction $\mathbf{s}_q \in \mathbb{S}^2$. Given a rendered image $I_{\mathcal{C}}^m$ from method m and its ground truth

Table 4: Datasets used in the multi-FoV evaluation. Resolution is reported as width × height after preprocessing. FoV denotes horizontal × vertical field of view. The scene count refers to the subset used in our evaluation.

Camera	Dataset	Resolution	FoV	Scenes	Environment	Source
Perspective	Mip-NeRF 360 [1]	1237×822	$52^\circ \times 36^\circ$–$68^\circ \times 47^\circ$	9	Indoor/Outdoor	Real
Fisheye	ScanNet++ [6]	1752×1168	$127^\circ \times 85^\circ$	7	Indoor	Real
	FIORD [4]	3264×3264	$120^\circ \times 120^\circ$–$170^\circ \times 170^\circ$	7	Indoor/Outdoor	Real
Omnidirectional	Ricoh360 [2]	1920×960	$360^\circ \times 180^\circ$	12	Outdoor	Real
	360Roam [5]	6080×3040	$360^\circ \times 180^\circ$	11	Indoor	Real
	OmniBlender [2]	2000×1000	$360^\circ \times 180^\circ$	8	Indoor/Outdoor	Synthetic

$G_{\mathcal{C}}$, we construct their HEALPix representations using the same projection and interpolation:

$$I^m_{\mathrm{H}}(q) = I^m_{\mathcal{C}}\big(\Pi^{-1}_{\mathcal{C}}(\mathbf{s}_q)\big), \qquad G_{\mathrm{H}}(q) = G_{\mathcal{C}}\big(\Pi^{-1}_{\mathcal{C}}(\mathbf{s}_q)\big). \tag{8}$$

Only directions that project into the valid camera domain are retained through a visibility mask

$$M_{\mathcal{C}}(q) = \mathbf{1}\big[\Pi^{-1}_{\mathcal{C}}(\mathbf{s}_q) \in \mathcal{D}_{\mathcal{C}}\big], \tag{9}$$

where $\mathcal{D}_{\mathcal{C}}$ is the valid image region determined by the camera model and FoV. The local HEALPix-SSIM value h_q is computed over the spherical neighborhood $\mathcal{N}_{\mathrm{H}}(q)$ using the boundary-aware kernels described in the main paper. The final score is the masked average

$$\mathrm{HSSIM}(I^m_{\mathcal{C}}, G_{\mathcal{C}}) = \frac{\sum_q M_{\mathcal{C}}(q)\, h_q(I^m_{\mathrm{H}}, G_{\mathrm{H}})}{\sum_q M_{\mathcal{C}}(q)}. \tag{10}$$

All baseline and UniTriSplat outputs therefore undergo the same camera-to-sphere mapping, visibility masking, interpolation, and HSSIM computation. This protocol isolates structural quality on the sphere from the nonuniform sampling of the original image projection.

7.3 Preprocessing of Omnidirectional Images

Commercial 360° cameras commonly form an omnidirectional image by calibrating, warping, and blending images from multiple fisheye lenses. This upstream imaging process may introduce seam discontinuities, exposure differences, or local geometric misalignment. For datasets that provide processed omnidirectional images, we use the released stitched frames directly. For raw dual-fisheye captures, standard preprocessing first maps both lenses to a common spherical coordinate system using the calibrated camera parameters, applies photometric correction in overlapping regions, and blends the overlap before conversion to an equirectangular image. The resulting omnidirectional images are used consistently for pose estimation, training, and evaluation by all methods.

Table 5: Cross-resolution evaluation on 360Roam. Ours(H) and Ours(L) use the HEALPix subdivision levels immediately above and below the target input pixel count, respectively. Blue and light blue denote the best and second-best results within each resolution.

Resolution	Method	Reconstruction				Runtime		Resources		
		PSNR↑	SSIM↑	LPIPS↓	HSSIM↑	Train (min)↓	Render (ms)↓	Static GPU Mem. (MB)↓	Peak GPU Mem. (MB)↓	Avg. #Gaussians↓
6080 × 3040	ODGS	20.72	0.722	0.383	0.685	247.6	70.44	1740.2	17176.6	598872
	OP43DGS	21.04	0.732	0.376	0.686	330.7	83.21	1867.9	15560.9	683659
	OmniGS	21.48	0.741	0.369	0.710	122.6	**23.65**	**848.4**	**5983.3**	465993
	Ours(H)	**22.53**	**0.759**	**0.346**	**0.733**	324.4	92.01	1216.4	8926.4	521740
	Ours(L)	21.82	0.747	0.353	0.724	**102.1**	40.80	1035.5	7037.7	**454153**
3040 × 1520	ODGS	22.43	0.736	0.353	0.701	67.1	32.79	1092.1	10482.3	1025384
	OP43DGS	23.10	0.783	0.305	0.737	150.0	51.22	842.3	6158.2	779652
	OmniGS	23.61	0.793	0.299	0.752	**33.5**	**15.33**	**582.0**	**3724.8**	768071
	Ours(H)	**24.37**	**0.811**	**0.276**	**0.795**	97.8	38.25	1176.4	9826.1	932404
	Ours(L)	23.99	0.798	0.283	0.778	50.4	30.96	713.8	4812.5	**705181**
1520 × 760	ODGS	22.93	0.758	0.254	0.711	30.3	19.67	826.9	6217.1	1342318
	OP43DGS	23.51	0.774	0.230	0.756	65.1	31.87	635.2	3924.7	1063944
	OmniGS	25.05	0.809	0.187	0.763	**20.6**	11.09	**438.6**	**2478.5**	**791826**
	Ours(H)	**26.10**	**0.827**	**0.167**	**0.821**	46.7	25.50	895.5	5843.6	1101735
	Ours(L)	23.84	0.796	0.198	0.775	25.2	**10.46**	541.0	3186.9	943072

UniTriSplat does not contain a dedicated seam-removal module. Stitching artifacts arise from the physical multi-lens capture and preprocessing pipeline rather than the 3DGS rasterization model. Handling such hardware-specific artifacts is therefore outside the main scope of this work. Our objective is to provide a unified rasterizer for the processed perspective, fisheye, and omnidirectional images. More advanced calibration and seam-aware preprocessing can be incorporated independently in future work.

8 Additional Evaluation

We supplement the multi-FoV results with a controlled cross-resolution evaluation on 360Roam. This experiment analyzes how the discrete HEALPix hierarchy affects reconstruction quality, runtime, and resource consumption as the input resolution changes.

8.1 Cross-Resolution Protocol

We construct three image scales from the high-resolution 360Roam dataset: 6080×3040, 3040×1520, and 1520×760. All methods use the same train/test split, calibrated camera poses, SfM initialization, 30,000 optimization iterations, and a single NVIDIA GeForce RTX 4090 D GPU. The baseline methods retain their official optimization settings, while UniTriSplat uses the spherical-domain learning-rate and densification scaling described in Sec. 10.3. The final-pruning variant is not used in any reported result.

HEALPix supports only discrete subdivision levels. We therefore evaluate two adjacent configurations. **Ours(H)** selects the HEALPix level immediately above

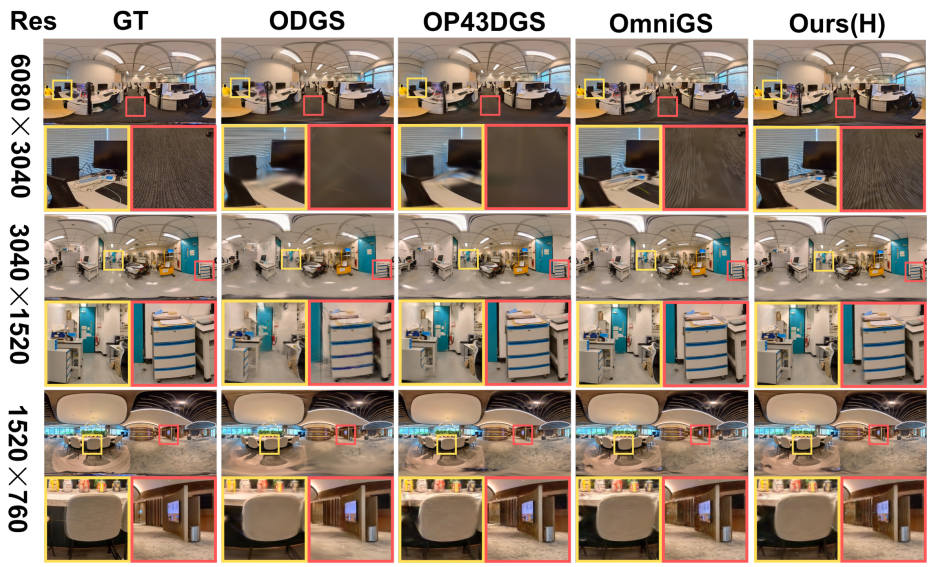

Fig. 9: Qualitative cross-resolution comparison. From top to bottom, the Office, Lab, and Cafe scenes from 360Roam are evaluated at decreasing input resolutions. The enlarged regions emphasize texture and polar details. Ours(H) preserves fine structures more consistently, whereas the lower HEALPix level may lose high-frequency details after sphere-to-image resampling.

the target input pixel count, while **Ours(L)** selects the level immediately below it. Ours(H) prioritizes sampling density and reconstruction quality; Ours(L) reduces the number of sampled spherical pixels and targets a better efficiency–quality trade-off. Training time measures the complete 30,000-iteration optimization, and rendering time is averaged over the test views. Static GPU memory is measured after model initialization, whereas peak memory records the maximum allocated GPU memory during training. The number of Gaussians is averaged over the evaluated scenes after training.

8.2 Cross-Resolution Results

As shown in Tab. 5, Ours(H) achieves the best reconstruction metrics at all three resolutions, confirming that a denser HEALPix level increases representation capacity. Ours(L) provides a different operating point: it remains competitive at medium and high resolutions while reducing the number of sampled spherical pixels. At 6080×3040, Ours(L) completes training in 102.1 minutes, compared with 122.6 minutes for OmniGS, and uses the fewest Gaussians among all methods. Its static and peak GPU memory are higher than OmniGS but substantially lower than ODGS and OP43DGS.

The efficiency gain of Ours(L) at the highest resolution is associated with approximately 31.9% fewer sampled pixels than the matched equirectangular grid,

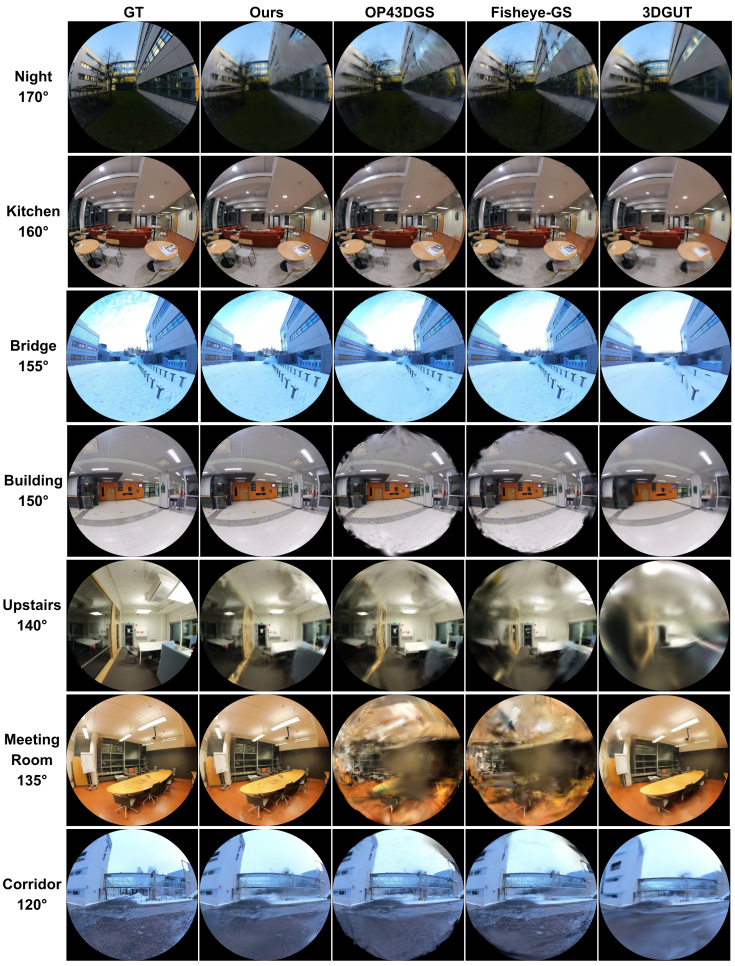

Fig. 10: Qualitative comparison on FIORD using fisheye training inputs. UniTriSplat is compared with OP43DGS, Fisheye-GS, and 3DGUT.

which offsets the CUDA overhead of HEALPix indexing and sphere-to-image sampling. This benefit is resolution-dependent rather than universal. At lower resolutions, the fixed indexing and resampling costs account for a larger fraction of runtime, and the lower HEALPix level may lose high-frequency information during sphere-to-image sampling. This explains why Ours(L) is less accurate than OmniGS at 1520×760, whereas Ours(H) recovers the missing detail by using the next subdivision level. Ours(L) nevertheless attains the lowest rendering latency at this scale.

The qualitative comparisons in Fig. 9 support the quantitative results. In the high-resolution Office scene, Ours(H) preserves carpet texture that is smoothed by the baseline. In the Cafe scene, it retains fine ceiling structures near the

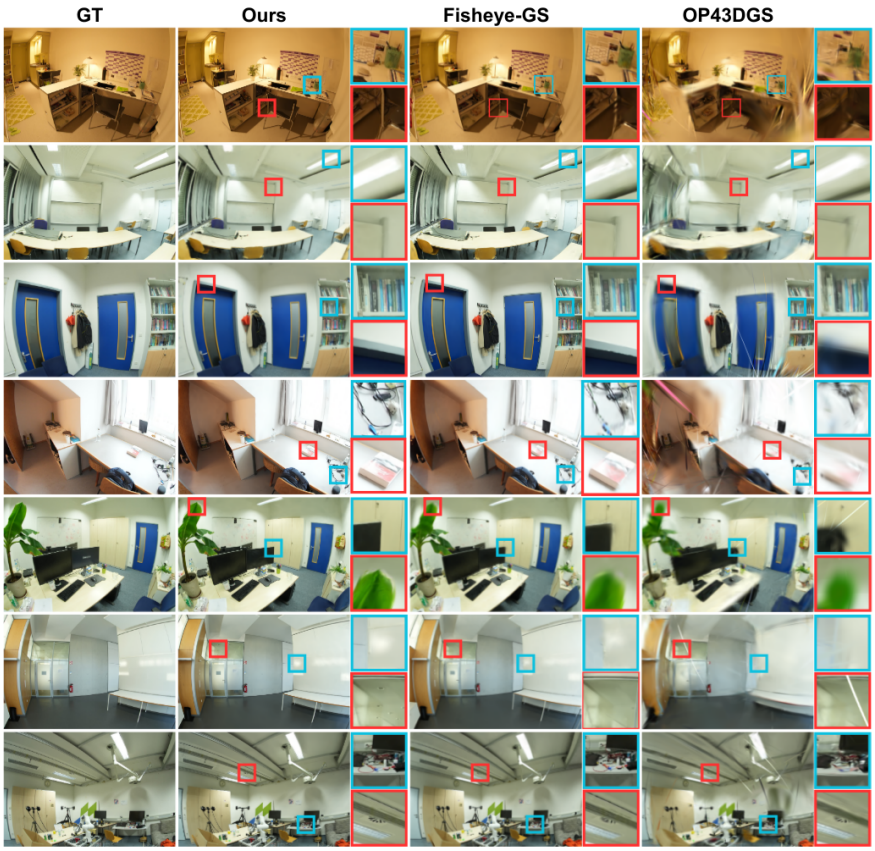

Fig. 11: Qualitative comparison on ScanNet++ using fisheye training inputs. Uni-TriSplat is compared with Fisheye-GS.

viewing poles even after downsampling. These observations are consistent with the equal-area sampling of HEALPix, which avoids the strong polar oversampling of equirectangular images. At the same time, the softer output of Ours(L) at low resolution exposes the remaining limitation of sphere-to-image interpolation.

8.3 Efficiency Limitations and Future Work

The generality of spherical rasterization introduces overhead that is absent from camera-specific planar pipelines. HEALPix indexing, spherical tile queries, and the final sphere-to-image sampling require additional CUDA operations. These costs are amortized for high-resolution omnidirectional inputs, where equal-area sampling removes substantial polar redundancy, but they are more visible for low-resolution or narrow-FoV images. Future work will simplify HEALPix indexing and search, improve sphere-to-planar reconstruction filters, and develop

Fig. 12: Qualitative comparison on Mip-NeRF 360 using perspective training inputs. UniTriSplat is compared with vanilla 3DGS and OP43DGS using the pinhole model.

specialized low-level GPU kernels. These optimizations target lower overhead without changing the unified camera-model formulation.

9 Additional Qualitative Results

This section provides additional qualitative comparisons for fisheye and perspective training inputs and for cross-camera rendering of Gaussian scenes reconstructed by vanilla 3DGS.

9.1 Fisheye Image Inputs

For fisheye training, we compare UniTriSplat with Fisheye-GS and OP43DGS on two datasets. Moreover, we also compare with 3DGUT on the FIORD dataset. Fisheye-GS and OP43DGS incorporate fisheye projection into EWA splatting, whereas UniTriSplat optimizes the scene on the common HEALPix sphere. We use the equidistant fisheye model for all methods. For datasets calibrated with polynomial distortion models, the input images are undistorted and reprojected using the provided calibration before training. We also modify the OP43DGS fisheye rasterizer to accept explicit camera intrinsics rather than deriving focal length only from a symmetric FoV.

As shown in Figs. 10 and 11, UniTriSplat produces fewer floating artifacts and better preserves local structures under strong fisheye distortion. The difference is most visible on FIORD, whose ultra-wide FoV covers regions close to and beyond the hemisphere boundary. ScanNet++ contains a smaller FoV and more accurate pose registration; on these indoor scenes, the improvements are concentrated around fine structures and image boundaries. The FIORD Night scene remains more challenging because low illumination and severe distortion reduce feature-matching accuracy and degrade the SfM poses.

9.2 Perspective Image Inputs

For perspective training, we compare UniTriSplat with vanilla 3DGS and OP43DGS on Mip-NeRF 360. The spherical representation provides stable Gaussian footprints near image boundaries, but the final resampling from the HEALPix sphere

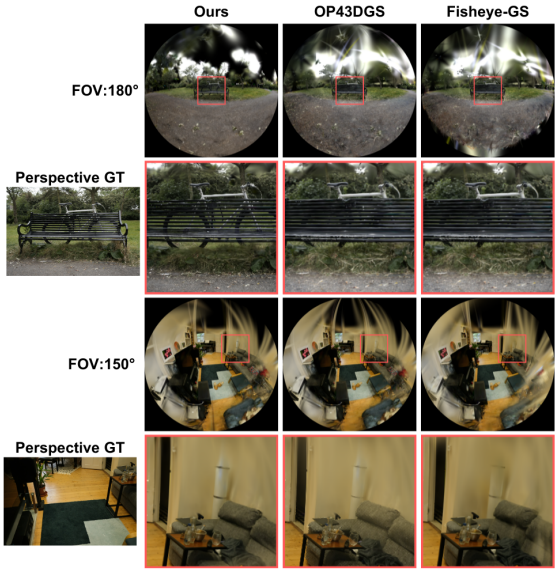

Fig. 13: Fisheye rendering of vanilla-3DGS models trained on Mip-NeRF 360, compared with OP43DGS and Fisheye-GS.

to a perspective image can reduce sharpness. This effect is more visible for narrow-FoV images because they have a higher angular resolution than panoramic inputs at the same image resolution.

In Fig. 12, UniTriSplat recovers scene geometry consistently and produces fewer thin, spike-shaped artifacts near image boundaries. In contrast, its output is slightly softer than vanilla 3DGS and OP43DGS in some high-frequency regions. The equal-area spherical representation avoids the strong stretching of projected Gaussian footprints near perspective-image boundaries, while the remaining blur results from interpolation between spherical and planar samples. Improving this resampling step is an important direction for future work.

9.3 Cross-Camera Validation on Vanilla 3DGS Models

We further evaluate cross-camera rendering on Gaussian scenes reconstructed by vanilla 3DGS from perspective images. Without retraining the Gaussian representation, we use different rasterizers to synthesize fisheye and omnidirectional views. This experiment separates the compatibility of the rasterizer from the optimization procedure used to obtain the Gaussian scene.

The first rows of Fig. 13 show that UniTriSplat preserves fine details when converting perspective-trained scenes to fisheye views. The indoor examples also contain fewer floating artifacts near image boundaries. In the omnidirectional comparisons of Fig. 14, UniTriSplat retains fine structures in the enlarged regions despite the incomplete spherical coverage of the original perspective training images. These results show that the HEALPix rasterizer can be applied

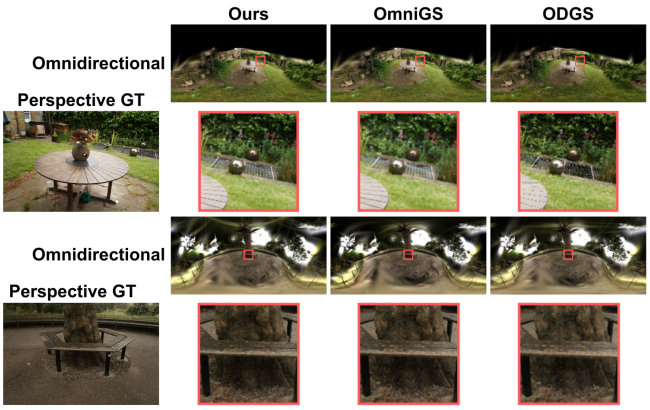

Fig. 14: Omnidirectional rendering of vanilla-3DGS models trained on Mip-NeRF 360, compared with OmniGS and ODGS.

to Gaussian scenes reconstructed by an external perspective pipeline and can synthesize camera models not used during their optimization.

10 Implementation Details

This section details the implementation of the UniTriSplat framework. Section 10.1 presents a unified camera modeling strategy that ensures compatibility across arbitrary FoV inputs, offering broader support for camera models than prior 3DGS methods. Section 10.2 elaborates on the HEALPix-based rasterization, covering grid resolution scaling, dynamic memory management, and algorithmic pseudocode. Finally, Section 10.3 discusses hyperparameter tuning for Gaussian optimization within the radian domain on the HEALPix grid.

10.1 Compatibility with Any FoV Input

Our approach accommodates perspective, fisheye, and omnidirectional projections by adopting a FoV-based camera modeling strategy, thereby enabling seamless adaptation to any input FoV. All camera models share the same Gaussian-to-HEALPix rasterization process. Only the camera-to-sphere mapping and the valid spherical region change during image synthesis. This design decouples spherical scene representation from camera-specific projection and enables perspective, fisheye, and omnidirectional rendering within a single rasterization framework.

For partial-FoV cameras, we restrict computation to HEALPix pixels inside the visible spherical region defined by the camera intrinsics and horizontal and vertical FoVs. The rendered spherical samples are then interpolated onto the target image plane. We next describe the camera-to-sphere mappings used for perspective, fisheye, and omnidirectional images.

Table 6: Resolution hierarchy in HEALPix rendering. Given an input image of size $W \times H$ covering a solid angle Ω_{in}, the HEALPix resolution is selected to match the average solid angle per input pixel. The default configuration uses the nearest HEALPix order, while k_{L} and k_{H} denote the adjacent lower and higher orders used in the cross-resolution evaluation.

Parameter	Symbol	Definition
Input resolution	$W \times H$	Image dimensions
Visible solid angle	Ω_{in}	Solid angle covered by the camera FoV
Continuous target	N^*_{side}	$\sqrt{\frac{4\pi WH}{12\Omega_{\text{in}}}}$
Default HEALPix order	k	$\text{round}(\log_2 N^*_{\text{side}})$
Lower / higher orders	k_{L}, k_{H}	$\lfloor \log_2 N^*_{\text{side}} \rfloor$, $\lceil \log_2 N^*_{\text{side}} \rceil$
HEALPix side resolution	N_{side}	2^k
Total spherical samples	N_{pix}	$12N^2_{\text{side}} = 12 \cdot 4^k$
Tile order	k_t	$k - \log_2 B$
Full-sphere tile count	N_{tile}	$12 \cdot 4^{k_t} = 12\,(N_{\text{side}}/B)^2$
Tile size	B	16 (16×16 HEALPix samples)

Perspective (Pinhole): We adopt the standard pinhole camera model used in vanilla 3DGS. Given image dimensions $W \times H$ and field of view FoV_x, the focal length is $f_x = W/(2\tan(\text{FoV}_x/2))$ (similarly for f_y), with principal point $(c_x, c_y) = (W/2, H/2)$.

For a pixel (u, v), we compute normalized coordinates $x = (u - c_x)/f_x$, $y = (v - c_y)/f_y$, then obtain the viewing direction $\mathbf{d} = (x, y, 1)^\top / \|(x, y, 1)\|$. The spherical coordinates are:

$$\omega = \arctan 2(d_x, d_z), \quad \phi = \arcsin(d_y) \tag{11}$$

where $\omega \in [-\pi, \pi)$ and $\phi \in [-\pi/2, \pi/2]$.

Fisheye (Anisotropic Equidistant): The classical equidistant fisheye model is isotropic, using a single FoV parameter w such that the incident angle $\psi = w \cdot r$ where r is the normalized radial distance. However, this restricts the valid region to a circle inscribed within the image, wasting pixels in rectangular sensors. We extend this to an *anisotropic* equidistant model with independent horizontal and vertical FoV parameters (w_x, w_y), where $w = \text{FoV}/180$. This allows the valid region to form an ellipse that better utilizes the full image area, enabling higher effective resolution for the same sensor size.

Given image dimensions $W \times H$ and requiring the valid region to be tangent to the image boundary, we set focal lengths $f_x = W/\pi$, $f_y = H/\pi$, with principal point $(c_x, c_y) = (W/2, H/2)$.

For a pixel (u, v), we first compute normalized coordinates $m_x = (u - c_x)/f_x$, $m_y = (v - c_y)/f_y$, then obtain the radial distance $r_m = \sqrt{m_x^2 + m_y^2}$ and azimuth angle $\gamma = \arctan 2(m_y, m_x)$, which measures the angular direction of the pixel relative to the principal point. The key to anisotropic projection is the direction-

dependent FoV factor:

$$w_{\text{eff}}(\gamma) = \sqrt{(w_x \cos\gamma)^2 + (w_y \sin\gamma)^2} \tag{12}$$

which interpolates between w_x (horizontal, $\gamma = 0$) and w_y (vertical, $\gamma = \pi/2$) based on the azimuthal direction. The incident angle from the optical axis is then $\psi = r_m \cdot w_{\text{eff}}(\gamma)$, yielding the 3D viewing direction $\mathbf{d} = (\sin\psi\cos\gamma, \sin\psi\sin\gamma, \cos\psi)^\top$ in camera coordinates. Finally, the spherical coordinates are:

$$\omega = \arctan 2(d_x, d_z), \quad \phi = \arcsin(d_y) \tag{13}$$

The valid imaging region is defined by the following elliptical constraint:

$$\frac{(u - c_x)^2}{a^2} + \frac{(v - c_y)^2}{b^2} \leq 1, \tag{14}$$

where the semi-axes a and b are parameterized as:

$$a = f_x \frac{\pi}{2}, \quad b = f_y \frac{\pi}{2}. \tag{15}$$

Specifically, when $w_x = w_y = 1$, Eq. (14) reduces to a standard 180° isotropic fisheye. For configurations where $w_x = w_y > 1$, the model accommodates super-hemispherical imaging exceeding 180°.

Omnidirectional: For a full 360° × 180° panoramic image of size $W \times H$, each pixel (u, v) maps directly to spherical coordinates $\omega = 2\pi u/W - \pi$ and $\phi = \pi/2 - \pi v/H$, yielding direction $\mathbf{d} = (\cos\phi\sin\omega, \sin\phi, \cos\phi\cos\omega)^\top$.

Unified Spherical Rendering: Regardless of the input camera model, all pixels are first projected to spherical coordinates (ω, ϕ) using the respective projection equations described above. Our method then renders directly in HEALPix space, which provides an equal-area tessellation of the sphere with $N_{\text{pix}} = 12 \times N_{\text{side}}^2$ pixels covering the full 4π steradians. This unified spherical representation enables seamless handling of arbitrary FoV inputs within a single rendering framework.

10.2 Implementation of the HEALPix-based Rasterization

Algorithm 1 shows the pipeline of the HEALPix-based 3D Gaussian rasterization. The forward pass follows the three-stage pipeline of standard 3DGS, adapted for HEALPix grids. Stage 1 (preprocessing) transforms Gaussians to camera space and computes arc-length covariances via Eq. (2) as described in Sec. 3.2. Stage 3 (rendering) performs per-pixel alpha compositing using great-circle distances per Eq. (6). We focus here on Stage 2: tile assignment strategies.

Stage 2: Tile Query Methods. We provide two tile query algorithms exploiting HEALPix's dual indexing schemes.

NESTED Quadtree Traversal. Algorithm 2 shows the pipeline of the NESTED quadtree traversal. We adapt the classical query-pixel algorithm from

Algorithm 1 HEALPix-Based 3D Gaussian Rasterization

Require: Gaussians $\mathcal{G} = \{(\boldsymbol{\mu}_i, \mathbf{s}_i, \mathbf{q}_i, o_i, \mathbf{c}_i)\}_{i=1}^{P}$, view $\mathbf{V} = [\mathbf{W} \mid \mathbf{b}]$, HEALPix pixel order k, tile size $B = 2^b$
Ensure: HEALPix image $\mathbf{I} \in \mathbb{R}^{N_{\text{pix}} \times 3}$, $N_{\text{pix}} = 12 \cdot 4^k$

$k_t \leftarrow k - b$ ▷ HEALPix order of the rendering tiles
// **Preprocessing (parallel over Gaussians)**
for $i = 1, \ldots, P$ **in parallel do**
  $\tilde{\boldsymbol{\mu}}_i \leftarrow \mathbf{W}\boldsymbol{\mu}_i + \mathbf{b}$; $(\omega_i, \phi_i) \leftarrow \Pi(\tilde{\boldsymbol{\mu}}_i)$; $\rho_i \leftarrow \|\tilde{\boldsymbol{\mu}}_i\|_2$
  $\boldsymbol{\Sigma}_{3D,i} \leftarrow \mathbf{R}_i \operatorname{diag}(\mathbf{s}_i^2)\, \mathbf{R}_i^\top$, where $\mathbf{R}_i = \textsc{QuatToMat}(\mathbf{q}_i)$
  $\mathbf{J}_i \leftarrow \left.\dfrac{\partial \Pi(\mathbf{t})}{\partial \mathbf{t}}\right|_{\mathbf{t}=\tilde{\boldsymbol{\mu}}_i}$; $\mathbf{S}_{\phi_i} \leftarrow \operatorname{diag}(\cos\phi_i, 1)$
  $\boldsymbol{\Sigma}_{\text{arc},i} \leftarrow \mathbf{S}_{\phi_i}\mathbf{J}_i\mathbf{W}\boldsymbol{\Sigma}_{3D,i}\mathbf{W}^\top\mathbf{J}_i^\top\mathbf{S}_{\phi_i}^\top$
  $r_{s,i} \leftarrow 3\sqrt{\lambda_{\max}(\boldsymbol{\Sigma}_{\text{arc},i})}$
end for
// **Tile assignment and front-to-back sorting**
for $i = 1, \ldots, P$ **in parallel do**
  $\mathcal{T}_i \leftarrow \textsc{QueryDisc}(\omega_i, \phi_i, r_{s,i}, k_t)$ ▷ NESTED or RING mode
  for each $t \in \mathcal{T}_i$ **do**
    Emit pair $((t, \rho_i), i)$ ▷ Tile-depth key and Gaussian index
  end for
end for
Sort pairs lexicographically by tile index and increasing radial depth
Identify per-tile ranges $[\texttt{start}_t, \texttt{end}_t)$ in the sorted Gaussian-index array
// **Parallel rendering**
for each pixel $p \in \{0, \ldots, N_{\text{pix}} - 1\}$ **in parallel do**
  $(\omega_p, \phi_p) \leftarrow \textsc{Nest2Lonlat}(p, k)$; $t \leftarrow \textsc{AncestorTile}(p, k_t)$
  $T \leftarrow 1$; $\mathbf{C} \leftarrow \mathbf{0}$
  for $j = \texttt{start}_t$ **to** $\texttt{end}_t - 1$ **do**
    $i \leftarrow \texttt{gaussian\_id}[j]$
    $\mathbf{d} \leftarrow \textsc{SphericalOffset}((\omega_p, \phi_p), (\omega_i, \phi_i))$
    $\alpha_i \leftarrow o_i \exp\left(-\frac{1}{2}\mathbf{d}^\top \boldsymbol{\Sigma}_{\text{arc},i}^{-1}\mathbf{d}\right)$
    $\mathbf{C} \leftarrow \mathbf{C} + T\alpha_i\mathbf{c}_i$; $T \leftarrow T(1 - \alpha_i)$
    if $T < \epsilon$ **then**
      break
    end if
  end for
  $\mathbf{I}[p] \leftarrow \mathbf{C}$
end for

the HEALPix [3], which performs depth-first search (DFS) through HEALPix's hierarchical NESTED structure. Starting from the 12 base quadrilaterals, each node is classified into zones based on its distance to the query disc center: zone 3 (fully inside), zone 2 (center inside), zone 1 (partially overlapping), or zone 0 (fully outside). Subtrees in zone 0 are pruned; zones 2–3 are added to results; zone 1 nodes are recursively subdivided.

To integrate this into our GPU rasterization pipeline, we search at a finer resolution $N_{\text{side}}^{\text{fact}} = \text{fact} \cdot N_{\text{side}}$ (where fact is typically 4 or 8) to handle boundary cases precisely, then map results back to tile resolution. This achieves $O(K + \log N_{\text{side}}^{\text{fact}})$ time per Gaussian but requires $O(\log N_{\text{side}}^{\text{fact}})$ stack memory per thread.

RING Sequential Scan. Algorithm 3 shows the pipeline of the RING sequential scan. This method exploits HEALPix's RING ordering, which arranges pixels along iso-latitude rings characterized by $z = \cos\vartheta$ where ϑ is the colatitude. Given a spherical disc with center (ω_c, ϕ_c) and radius r_s:

1. Compute z-bounds: $z_{\min} = \cos(\phi_c - r_s + \frac{\pi}{2})$, $z_{\max} = \cos(\phi_c + r_s + \frac{\pi}{2})$
2. Convert to ring indices $i_{r,\min}$, $i_{r,\max}$ using HEALPix ring formulas

Algorithm 2 QUERYDISCNESTED: Quadtree Traversal

Require: Disc center (ω, ϕ), radius r_s, tile order k, refinement f
Ensure: Overlapping tile indices $\mathcal{T}$
$k_{\max} \leftarrow k + \log_2 f$; **stack** $\leftarrow \{(p, 0) : p \in [0, 11]\}$; $\mathcal{T} \leftarrow \emptyset$
while stack $\neq \emptyset$ **do**
  Pop (p, o); $d \leftarrow \text{HAVERSINE}((\omega, \phi), \text{PIX2LONLAT}(p, o))$
  $z \leftarrow \text{CLASSIFY}(d, r_s, \text{PIXRAD}(o))$ ▷ 0:out, 1:partial, 2+:overlap
  if $z = 0$ **then continue** ▷ Prune
  else if $z \geq 2$ **and** $o \geq k$ **then** $\mathcal{T} \leftarrow \mathcal{T} \cup \{\lfloor p/4^{o-k} \rfloor\}$
  else if $o < k_{\max}$ **then** Push $(4p + c, o + 1)$ for $c \in \{0, 1, 2, 3\}$
  else $\mathcal{T} \leftarrow \mathcal{T} \cup \{\lfloor p/4^{o-k} \rfloor\}$
  end if
end while
return $\mathcal{T}$

3. For each ring i_r, analytically solve for the longitude half-width via Eq. (5)
4. Enumerate pixels within the $(\omega_c - \Delta\omega, \omega_c + \Delta\omega)$ range on each ring

This achieves $O(K)$ time with $O(1)$ memory and better GPU cache coherence due to sequential ring access. RING offers approximately 1.7× speedup over NESTED with marginal quality loss, as detailed in Sec. 5.3.

10.3 Training Parameter Scaling

The gradient dynamics of our method differ from the pinhole model due to the discrepancy in the projection domain. Specifically, the pinhole projection Jacobian scales with the focal length f as $\mathbf{J}_{\text{pixel}} \propto f/z$, whereas the spherical counterpart is governed by $\mathbf{J}_{\text{rad}} \propto 1/r$. Given that typical focal lengths f range from 500 to 1000, the spherical Jacobian magnitude is attenuated by several orders of magnitude. According to the chain rule, $\partial\mathcal{L}/\partial\boldsymbol{\mu}_{3D} = (\partial\mathcal{L}/\partial\boldsymbol{\mu}_{2D}) \cdot \mathbf{J}$, this reduction directly suppresses the 3D position gradients. To compensate for this gradient decay and ensure robust convergence, we scale the position learning rate upward by approximately one order of magnitude.

Conversely, the densification threshold—which governs adaptive Gaussian splitting—requires a downward recalibration to account for reduced gradient accumulation in the HEALPix domain. This phenomenon stems from three primary factors: (i) sparse effective sampling when the HEALPix resolution is coarser than the input imagery; (ii) diminished per-Gaussian activation frequency in scenarios where training views provide only partial spherical coverage; and (iii) the $\cos^2\phi$ metric distortion that attenuates gradient contributions near the poles. Consequently, we lower the densification threshold to maintain a splitting sensitivity consistent with vanilla 3DGS.

Supplementary References

1. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5470–5479 (2022)

Algorithm 3 QUERYDISCRING: RING Sequential Scan

Require: Disc center (ω, ϕ), radius r_s, tile order k
Ensure: Overlapping tile indices $\mathcal{T}$

$N_{\text{side}} \leftarrow 2^k$; $z_c \leftarrow \sin\phi$; $\mathcal{T} \leftarrow \emptyset$
$\phi_{\min} \leftarrow \max(-\pi/2, \phi - r_s)$; $\phi_{\max} \leftarrow \min(\pi/2, \phi + r_s)$
$z_{\min} \leftarrow \sin\phi_{\min}$; $z_{\max} \leftarrow \sin\phi_{\max}$
$(i_{r,\min}, i_{r,\max}) \leftarrow$ ZTORINGRANGE$(z_{\min}, z_{\max}, N_{\text{side}})$
for $i_r = i_{r,\min}$ **to** $i_{r,\max}$ **do**
  $(z_r, n_r, \omega_0) \leftarrow$ RINGINFO(i_r)
  $\delta \leftarrow \sqrt{\max(0, 1 - z_c^2)}\sqrt{\max(0, 1 - z_r^2)}$
  if $\delta < \epsilon$ **then** ▷ Disc center or ring at a pole
    if $z_c z_r \geq \cos r_s$ **then**
      Add all n_r pixels in ring i_r to $\mathcal{T}$
    end if
    continue
  end if
  $c_\omega \leftarrow (\cos r_s - z_c z_r)/\delta$
  if $c_\omega \leq -1$ **then**
    Add all n_r pixels in ring i_r to $\mathcal{T}$ ▷ Full ring
  else if $c_\omega < 1$ **then**
    $\Delta\omega \leftarrow \arccos(\text{clamp}(c_\omega, -1, 1))$
    $i_{p,\min} \leftarrow \left\lfloor \frac{(\omega - \Delta\omega - \omega_0) n_r}{2\pi} \right\rfloor$
    $i_{p,\max} \leftarrow \left\lceil \frac{(\omega + \Delta\omega - \omega_0) n_r}{2\pi} \right\rceil$
    for $i_p = i_{p,\min}$ **to** $i_{p,\max}$ **do**
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{$RING2NEST$(i_r, \text{mod}(i_p, n_r))\}$
    end for
  end if
end for
return $\mathcal{T}$

2. Choi, C., Kim, S.M., Kim, Y.M.: Balanced spherical grid for egocentric view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16590–16599 (2023)
3. Gorski, K.M., Hivon, E., Banday, A.J., Wandelt, B.D., Hansen, F.K., Reinecke, M., Bartelmann, M.: Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. The Astrophysical Journal **622**(2), 759–771 (2005)
4. Gunes, U., Turkulainen, M., Ren, X., Solin, A., Kannala, J., Rahtu, E.: Fiord: A fisheye indoor-outdoor dataset with lidar ground truth for 3d scene reconstruction and benchmarking. In: Scandinavian Conference on Image Analysis. pp. 3–17. Springer (2025)
5. Huang, H., Chen, Y., Zhang, T., Yeung, S.K.: 360roam: Real-time indoor roaming using geometry-aware 360° radiance fields. arXiv preprint arXiv:2208.02705 (2022)
6. Yeshwanth, C., Liu, Y.C., Nießner, M., Dai, A.: Scannet++: A high-fidelity dataset of 3d indoor scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12–22 (2023)