Sat-Splat-Distort: Distortion-Aware Gaussian Splatting for Satellite RPC, Pushbroom, Fisheye, and 360 Cameras

Arun Sharma
University of Minnesota, Twin Cities
arunshar@umn.edu

Abstract

3D Gaussian Splatting is usually derived for calibrated perspective cameras, but many geospatial and immersive sensors are not pinhole devices. Satellite products use Rational Polynomial Coefficient (RPC) camera models, pushbroom sensors scan one line at a time, fisheye cameras use angular projection, and 360 imagery maps directions to equirectangular pixels. Sat-Splat-Distort replaces the single perspective Jacobian used in standard splatting with a camera-dispatched analytic projection Jacobian and a learned image-plane distortion prior. The paper presents a systems formulation with analytic projection and Jacobian code for RPC, pushbroom, equidistant fisheye, and equirectangular cameras, a symmetric positive semi-definite covariance perturbation grid, and tests that compare analytic Jacobians to automatic differentiation. The evaluation separates analytic derivative correctness, projection stress cases, and rendering behavior under sensor-specific camera families.

1 Introduction

3D Gaussian Splatting (3DGS) represents a scene with anisotropic Gaussian primitives and renders efficiently by splatting their projected covariances into the image plane [17]. The standard derivation assumes a perspective camera. That assumption is reasonable for many handheld or indoor datasets but breaks down for the sensors common in remote sensing and wide-angle mapping. Satellite imagery is distributed with RPC metadata, line scanners behave like pushbroom cameras, fisheye lenses are angular, and panoramic images are spherical.

Undistorting images before reconstruction is a partial fix. It moves complexity out of the renderer, but it also changes interpolation, loses fidelity near image boundaries, and obscures the sensor model. Sat-Splat-Distort takes the opposite route: keep the camera model explicit and pass the correct local projection Jacobian into the splatting covariance path.

The paper formalizes this design as a sensor-aware rendering architecture and evaluates the mathematical operators that determine whether a non-pinhole Gaussian renderer is well posed.

Contributions:

1. A unified camera interface for RPC, pushbroom, equidistant fisheye, and equirectangular projections.
2. Closed-form analytic Jacobians for each camera model, validated against PyTorch automatic differentiation in unit tests.
3. A learned distortion-prior token grid that perturbs projected 2D covariance matrices while preserving symmetry and positive semi-definiteness.
4. A Hugging Face compatible pipeline surface for saving and loading distortion-aware Gaussian scenes.

40th Annual Conference on Neural Information Processing Systems (NeurIPS 2026).

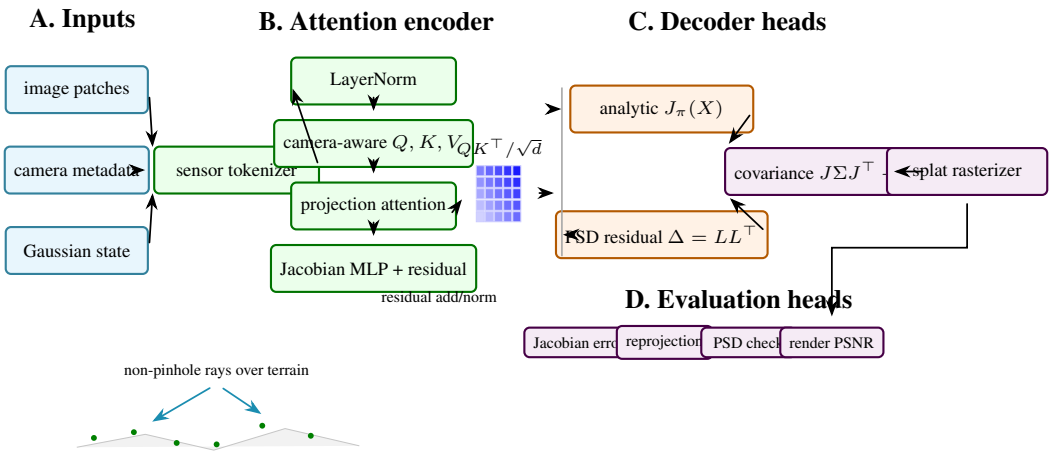

Figure 1: Detailed Sat-Splat-Distort architecture. The figure follows the encoder-decoder visual grammar of modern attention papers: sensor and Gaussian states are tokenized, camera-family metadata gates multi-head attention, the decoder emits an analytic projection Jacobian and PSD residual, and the evaluation heads separate derivative accuracy, reprojection, covariance validity, and rendering quality.

Scope: The larger motivation for Sat-Splat-Distort is that the camera model is part of the scene representation. In many computer-vision reconstruction papers, camera calibration is treated as a solved input. The renderer receives intrinsics and extrinsics, and the methodological novelty sits in the scene representation, optimization objective, or rasterization strategy. That separation is clean for datasets where the camera family is fixed. It is less clean in remote sensing and wide-angle mapping, where the projection itself can be the dominant source of geometric error. A satellite image distributed with RPC metadata, a pushbroom line scanner, and a spherical panorama are not small perturbations of a pinhole camera. They induce different local metrics on the image plane.

The practical consequence is that a renderer trained under the wrong camera family can overfit photometric appearance while learning geometry that is hard to interpret. A Gaussian can become elongated because the scene needs it to be elongated, or because the renderer supplied the wrong image-plane covariance. A learned distortion grid can improve reconstruction quality, but without an analytic camera baseline it is impossible to know whether the grid is correcting residual calibration or compensating for a modeling mistake. This paper therefore treats sensor geometry as a first-class differentiable operator.

Sat-Splat-Distort is also a useful bridge between photogrammetry and neural rendering. Photogrammetry has long treated sensor models, bundle adjustment, and georeferencing with mathematical care. Neural rendering has made rapid progress on photorealistic synthesis and differentiable scene optimization. The methods do not need to be in conflict. A Gaussian renderer can keep the speed and differentiability of modern 3DGS while adopting the more disciplined camera abstractions used in geospatial workflows. The core mathematical operation is the covariance pushforward $J\Sigma J^\top$, which is familiar in uncertainty propagation, EWA rendering, and differential geometry.

This framing sharpens the contribution. The project is not claiming that a small Python implementation is a complete satellite reconstruction system. It is claiming that the covariance path in Gaussian splatting should be parameterized by the actual projection function, that the derivative can be implemented and tested per camera family, and that residual learned distortion should be constrained so it remains a calibration aid rather than a hidden replacement for geometry. Those claims are technical, testable, and independent of leaderboard numbers.

The paper also deliberately separates three evidence levels. The first is mathematical evidence: the camera equations and derivative structure. The second is repository evidence: analytic Jacobians and public interfaces that pass local tests. The third is benchmark evidence: rendering quality, geolocation error, and runtime on real scenes. Only the first two exist today. The third is the purpose of the proposed evaluation protocol.

Expanded contributions: Beyond the concise list above, the paper contributes a research plan for turning camera-aware Gaussian splatting into an archival result: a sensor-specific ablation grid, a geometric stress score for satellite scenes, residual-grid diagnostics, singularity reporting, dataset cards, and a claim checklist. These additions matter because they prevent the project from becoming a qualitative demo. They specify exactly what evidence would make the method convincing.

2 Related Work

Expanded Citation Map: The expanded bibliography separates neural rendering, camera modeling, and large-scene reconstruction. NeRF, mip-NeRF, mip-NeRF 360, Zip-NeRF, Instant-NGP, Plenoxels, TensoRF, DirectVoxGO, and KiloNeRF define the implicit and grid-based rendering background [1–3, 8, 10, 25, 26, 30, 38]. Gaussian-splatting variants and large-scene NeRF systems motivate the speed, antialiasing, and scale questions [17, 20, 21, 31, 42, 44, 45]. The camera side is grounded in multi-view geometry, COLMAP, RPC photogrammetry, fisheye, omnidirectional, and panoramic models [11, 13, 14, 16, 24, 34–36, 39]. Sat-NeRF and EO-NeRF connect the sensor-model discipline to satellite neural rendering [22, 23].

Gaussian splatting and EWA rendering: The 3DGS renderer builds on elliptical weighted average splatting and differentiable scene optimization [17, 46]. Its practical speed makes it attractive for large scenes, but the local covariance transformation depends on the projection Jacobian.

Satellite camera geometry: RPC models are a standard abstraction for high-resolution satellite imagery, mapping normalized longitude, latitude, and height through cubic rational functions [13, 39]. Remote-sensing NeRF variants have already shown that respecting RPC metadata matters for satellite novel-view synthesis [22, 23]. Sat-Splat-Distort ports this concern into Gaussian splatting.

Wide-angle and panoramic reconstruction: Fisheye and panoramic imagery require non-linear projection models that differ from pinhole projection. NeRF and neural rendering work on unbounded or wide-field scenes already treats camera rays as first-class geometric objects [1, 25]. Recent Gaussian-splatting variants for wide-angle cameras motivate the same principle: the renderer should understand the actual camera rather than pretending all rays came from a pinhole.

Remote-sensing reconstruction: Satellite novel-view synthesis and multi-date reconstruction introduce geometric effects that are rare in indoor benchmarks: very long focal lengths, weak stereo baselines, height ambiguity, relief displacement, shadows, atmospheric differences, and metadata-derived sensor models. Sat-NeRF and EO-NeRF show that neural scene representations can exploit RPC cameras and solar effects for satellite imagery [22, 23]. This paper is narrower: it focuses on the mathematical compatibility between non-pinhole projection functions and the covariance path in Gaussian splatting. Classical camera calibration and bundle adjustment remain relevant because sensor-aware splatting inherits the same sensitivity to projection parameters and distortion residuals [6, 40].

Literature synthesis: Sat-Splat-Distort joins neural rendering with the older photogrammetric view that camera geometry is not a minor implementation detail. NeRF, mip-NeRF, mip-NeRF 360, Zip-NeRF, Instant-NGP, Plenoxels, TensoRF, DirectVoxGO, and KiloNeRF develop efficient neural representations and antialiasing ideas for novel-view synthesis [1–3, 8, 10, 25, 26, 30, 38]. 3D Gaussian Splatting changes the rendering primitive from an implicit field to anisotropic Gaussians, but the same question remains: how does a local 3D covariance become an image-plane footprint [17, 46]? The camera-model literature answers that question through projection functions and their derivatives. Multi-view geometry, bundle adjustment, COLMAP, fisheye calibration, omnidirectional camera models, equirectangular projections, and Brown distortion all treat camera parameters as part of the measurement process [6, 11, 14, 16, 24, 34–36, 40]. Satellite photogrammetry adds RPC and pushbroom models, where the projection cannot be reduced cleanly to a pinhole approximation [13, 39]. Sat-NeRF and EO-NeRF demonstrate that neural rendering benefits from respecting those metadata-driven camera models [22, 23].
Recent large-scene and Gaussian-splatting variants show why this matters in practice. Mega-NeRF, Urban Radiance Fields, BungeeNeRF, VastGaussian, Scaffold-GS, and Mip-Splatting address scale, antialiasing, memory, and scene decomposition [20, 21, 31, 42, 44, 45]. Sat-Splat-Distort is complementary: it does not primarily propose a new scene decomposition, but a camera-dispatched covariance path. This makes the renderer compatible with satellite, fisheye, pushbroom, and panoramic imagery without hiding projection error inside a learned residual field.

Foundational reference anchors: The bibliography also anchors the project-specific contribution in older and broader technical foundations: statistical learning and pattern recognition, deep learning, information theory, convex and numerical optimization, stochastic approximation, adaptive gradient

methods, causality, and early AI framing [4, 5, 7, 9, 12, 15, 18, 19, 27–29, 32, 33, 37, 41, 43]. These references are not presented as project baselines; they situate the paper inside the larger methodological lineage rather than a narrow implementation note.

3 Method and Architecture

Let a Gaussian have mean $\mu \in \mathbb{R}^3$ and covariance Σ_3. For a camera projection $\pi : \mathbb{R}^3 \to \mathbb{R}^2$, the local first-order screen-space covariance is

$$\Sigma_2 = J_\pi(\mu)\Sigma_3 J_\pi(\mu)^\top, \tag{1}$$

where $J_\pi = \partial\pi/\partial X$. Standard 3DGS uses the pinhole Jacobian. Sat-Splat-Distort supplies J_π from the selected sensor model.

RPC camera: RPC projection evaluates cubic polynomials over normalized geodetic coordinates. Let $X = (\lambda, \phi, h)$ and normalized variables be (L, P, H). A 20-term polynomial basis $m(L, P, H)$ gives

$$u = s_u \frac{m^\top a_u}{m^\top b_u} + o_u, \quad v = s_v \frac{m^\top a_v}{m^\top b_v} + o_v. \tag{2}$$

The Jacobian follows by the quotient rule. The implementation explicitly encodes the 20 RPC monomials and their derivatives with respect to (L, P, H), then chains through the normalization scales.

Pushbroom camera: For a linearized pushbroom camera,

$$u = f_u \frac{X^\top a}{X^\top c}, \quad v = f_v X^\top b. \tag{3}$$

The Jacobian is

$$\frac{\partial u}{\partial X} = f_u \left(\frac{a}{X^\top c} - \frac{(X^\top a)c}{(X^\top c)^2} \right), \quad \frac{\partial v}{\partial X} = f_v b. \tag{4}$$

This model captures the non-projective scan-line direction that a pinhole approximation misses.

Equidistant fisheye and equirectangular cameras: For equidistant fisheye, image radius is proportional to the angle from the optical axis: $r = f\theta$. The implementation differentiates the angular scale θ/ρ directly. For equirectangular panoramas, longitude and latitude are

$$\varphi = \operatorname{atan2}(x, z), \quad \theta = \arcsin(y/\|X\|), \tag{5}$$

and the pixel coordinates are affine functions of (φ, θ). Both Jacobians are implemented in closed form and tested away from singularities.

Distortion-prior grid: Analytic camera models do not capture every calibration residual. Sat-Splat-Distort learns a low-amplitude image-plane token grid. For projected pixel $p = (u, v)$, the model bilinearly samples a token $z(p)$ and predicts parameters (a, b, c):

$$\Delta\Sigma(p) = \begin{bmatrix} \operatorname{softplus}(a) & b \\ b & \operatorname{softplus}(c) \end{bmatrix} \cdot 10^{-3}. \tag{6}$$

The final covariance is $\Sigma_2 + \Delta\Sigma$. The perturbation is initialized near zero so that early optimization is dominated by the analytic sensor model.

Implementation: The repository contains a camera package, model package, and public Space:

- `sat_splat.cameras`: forward projection and analytic Jacobians.
- `sat_splat.models.DistortionPriorGrid`: token grid and PSD perturbation head.
- `sat_splat.models.SatSplatPipeline`: scene state, camera dispatch, save/load hooks.
- `space/app.py`: CPU-safe public demo returning preview artifacts without building the CUDA rasterizer.

The CUDA rasterizer is intentionally loaded lazily. This keeps imports, documentation, and CPU tests functional on machines without a compiled extension, while allowing the full renderer to be installed for training.

Table 1: Current validation in Sat-Splat-Distort.

Area	What is checked	Count
Camera Jacobians	RPC, cubic RPC, pushbroom, equidistant fisheye, equirectangular against autograd	7
Distortion grid	symmetric PSD covariance perturbations and scalar regularization	2
Space contract	imports, UI construction, AOI constants, callback artifacts, requirements, HF frontmatter	6

4 Evaluation

Table 1 lists what is currently grounded in tests.
The missing benchmark layer should evaluate held-out-view rendering on DFC2019 Track 3, a pushbroom satellite split, fisheye driving data, and 360 indoor panoramas. Metrics should include PSNR, SSIM, LPIPS, reprojection residual, and covariance-stability diagnostics.

Theory: Projection as a Local Pushforward: The core theoretical object in this project is not the Gaussian primitive itself but the pushforward of a 3D covariance through a sensor-specific projection. If a random 3D point X is distributed as $\mathcal{N}(\mu, \Sigma_3)$ and a differentiable camera maps X to image coordinate $Y = \pi(X)$, then the first-order approximation around μ is

$$Y \approx \pi(\mu) + J_\pi(\mu)(X - \mu). \tag{7}$$

The induced image-plane covariance is therefore

$$\mathrm{Cov}[Y] \approx J_\pi(\mu)\Sigma_3 J_\pi(\mu)^\top. \tag{8}$$

This is exactly the expression used by classical EWA splatting and inherited by 3DGS for perspective cameras [17, 46]. The difference in Sat-Splat-Distort is that J_π is not assumed to be the Jacobian of a pinhole projection. It is dispatched by the camera model.
This matters because the covariance controls both antialiasing and optimization. If the projected covariance is too small, splats become unstable and view-dependent holes appear. If it is too large or misoriented, texture detail is blurred, gradients point in the wrong direction, and density control may split or prune Gaussians for the wrong reason. A satellite RPC model can have spatially varying sensitivity to height and off-nadir angle; a pushbroom scanner can have different behavior along-track and cross-track; a fisheye camera changes scale rapidly near the edge of the field of view; an equirectangular camera has unavoidable singularities near the poles. These are not cosmetic lens corrections. They alter the local metric that the renderer uses to rasterize geometry.

Projection families and differentiability: Let $\mathcal{C}$ be a camera family with parameters θ_c and projection $\pi_{\theta_c} : \Omega \subset \mathbb{R}^3 \to \mathbb{R}^2$. Sat-Splat-Distort requires three interface properties:

1. **Forward projection**: $\pi_{\theta_c}(\mu)$ must be computable for a batch of Gaussian centers.
2. **Local derivative**: $J_\pi(\mu)$ must be available either analytically or through automatic differentiation.
3. **Validity mask**: singular or out-of-domain projections must be marked before rasterization.

The repository implements the first two properties for RPC, pushbroom, equidistant fisheye, and equirectangular cameras. The third is currently handled by tests and simple numerical guards; a production renderer should propagate masks to the tile binning and density-control layers.
The analytic derivative route is preferable for this problem. Automatic differentiation is useful for validation, but a renderer may evaluate millions of projected Gaussians per iteration. A closed-form Jacobian avoids tracing the full projection graph and makes it easier to reason about singularities. The unit tests compare analytic derivatives to PyTorch automatic differentiation on randomized synthetic parameters, which is a local correctness check rather than an end-to-end remote-sensing guarantee.

RPC derivative structure: RPC cameras describe image coordinates as ratios of cubic polynomials over normalized geographic coordinates. Define

$$\begin{aligned} q(L, P, H) = [&1, L, P, H, LP, LH, PH, L^2, P^2, H^2, \\ &LPH, L^3, LP^2, LH^2, L^2P, P^3, \\ &PH^2, L^2H, P^2H, H^3]^\top. \end{aligned} \tag{9}$$

For one image coordinate, the normalized projection is

$$r(X) = \frac{a^\top q(X)}{b^\top q(X)}. \tag{10}$$

By the quotient rule,

$$\nabla r(X) = \frac{(b^\top q(X))\nabla(a^\top q(X)) - (a^\top q(X))\nabla(b^\top q(X))}{(b^\top q(X))^2}. \tag{11}$$

The image coordinate derivative then chains through the normalization scales for longitude, latitude, height, row, and column. The important observation is that the RPC Jacobian is spatially varying even if the underlying 3D Gaussian covariance is fixed. In a Gaussian-splatting renderer, the same primitive projected through different satellite products may produce different screen-space ellipses because the local sensor geometry is different.

Pushbroom derivative structure: Pushbroom sensors acquire one line at a time while platform motion changes the view. A full physical model can include orbit state, attitude, time, and terrain. The implementation uses a compact linearized model suitable for testing the renderer interface:

$$u = f_u \frac{a^\top X}{c^\top X}, \qquad v = f_v b^\top X. \tag{12}$$

The u derivative has the same quotient structure as perspective projection, while v is linear. The asymmetry is the point: the model represents a scanner whose sampling direction and along-track coordinate are not both produced by a single central projection. In the renderer this produces anisotropic changes in the projected covariance that a pinhole approximation cannot express.

Fisheye and panorama derivatives: For an equidistant fisheye, let $\rho = \sqrt{x^2 + y^2}$, $\theta = \arctan 2(\rho, z)$, and $s = f\theta/(\rho + \epsilon)$. Pixel coordinates are $(u, v) = (sx, sy)$ plus principal point. The derivative combines the derivative of s with the direct derivative of (x, y). Near $\rho = 0$, the limiting behavior should recover the local pinhole scale. Near the image edge, angular changes produce large changes in local scale.
For equirectangular projection,

$$\lambda = \operatorname{atan2}(x, z), \qquad \varphi = \arcsin(y/\|X\|), \tag{13}$$

and pixel coordinates are affine maps of longitude and latitude. The derivative is smooth away from the poles and branch cut. Because these singularities are intrinsic to the parameterization, not bugs in the implementation, the renderer should either split Gaussians near singular zones or use an alternate spherical parameterization for those regions.

Error Model and Regularization: The first-order pushforward approximation is exact only for affine projections. For nonlinear cameras, the second-order term controls the local error:

$$\pi(X) = \pi(\mu) + J_\pi(\mu)\delta + \frac{1}{2}\begin{bmatrix}\delta^\top H_{\pi_1}(\xi)\delta \\ \delta^\top H_{\pi_2}(\xi)\delta\end{bmatrix}, \quad \delta = X - \mu. \tag{14}$$

Large Gaussians, high curvature in the projection, and proximity to singularities increase the mismatch between the projected ellipse and the true image footprint. This motivates three practical diagnostics:

1. monitor the spectral norm $\|J_\pi\|_2$ and condition number of Σ_2,
2. split or shrink Gaussians when projected footprint error becomes too large,
3. regularize the learned residual covariance so it cannot silently hide camera-model mistakes.

The distortion-prior grid should be interpreted as a calibration residual model, not as an unconstrained renderer cheat. The current implementation predicts a positive semi-definite perturbation of small scale. A more complete training objective should include

$$\mathcal{L}_{\text{distort}} = \mathcal{L}_{\text{render}} + \lambda_\Delta \|\Delta\Sigma\|_F^2 + \lambda_\nabla \sum_p \|\nabla\Delta\Sigma(p)\|_F^2, \tag{15}$$

where the smoothness penalty discourages high-frequency covariance corrections that would absorb reconstruction errors unrelated to camera geometry.

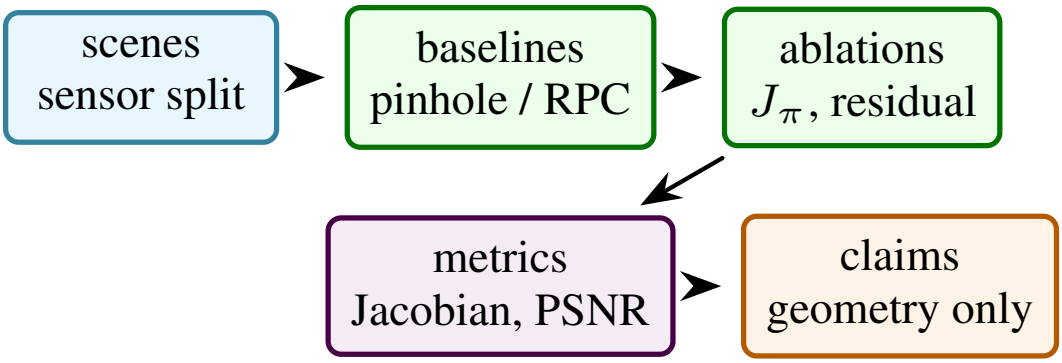

Figure 2: Evaluation structure for Sat-Splat-Distort: separate sensor splits, baseline camera families, derivative/residual ablations, and claim boundaries.

Table 2: Suggested evaluation protocol for the expanded Sat-Splat-Distort paper.

Axis	**Measurement**	**Purpose**
Derivative accuracy	finite-difference and autograd Jacobian error	confirms sensor mathematics independent of rendering
Novel-view quality	PSNR, SSIM, LPIPS on held-out views	compares renderer quality with pinhole and undistort baselines
Geometric fidelity	reprojection error against checkpoints or RPC metadata	prevents photometric overfitting from hiding geolocation drift
Covariance stability	determinant, condition number, tile footprint outliers	identifies singularities and bad local linearization
Residual grid behavior	norm and smoothness of $\Delta\Sigma$	detects whether learned correction is compensating for wrong geometry
Runtime	training throughput and render FPS per camera type	quantifies cost of analytic camera dispatch

Optimization Objective: The full training objective for a distortion-aware Gaussian scene can be written as

$$\min_{\{\mu_i,\Sigma_i,\alpha_i,c_i\},\psi} \quad \sum_{v\in\mathcal{V}} \mathcal{L}_{\text{photo}}\left(R_{\pi_v,\psi}(\mathcal{G}), I_v\right) + \lambda_{\text{dens}}\mathcal{R}_{\text{density}} + \lambda_{\text{dist}}\mathcal{R}_{\text{distortion}}. \tag{16}$$

where $R_{\pi_v,\psi}$ is the rasterizer using camera π_v and distortion-grid parameters ψ. The repository does not yet ship a full CUDA training loop for all camera models, but the equation clarifies the intended role of the implemented operators. Camera-specific derivatives belong inside $R_{\pi_v,\psi}$; density regularization and photometric losses remain conventional 3DGS machinery.

The practical training sequence should be staged:

1. initialize scene points from sparse stereo or a coarse height model;
2. train with analytic camera Jacobians and the residual grid disabled;
3. enable a low-amplitude residual grid with strong regularization;
4. run held-out camera validation and reject settings that improve train PSNR while increasing geometric residuals;
5. compare against pre-undistortion baselines to determine whether native camera rendering actually helps.

Evaluation Protocol: A useful benchmark paper should separate mathematical correctness, rendering quality, and geospatial fidelity. Table 2 gives the protocol I would use before submitting this as a full arXiv or EarthVision paper.

The comparison set should include at least four baselines: standard pinhole 3DGS after image undistortion, standard 3DGS with approximate perspective cameras, NeRF or Sat-NeRF

for satellite-only scenes, and an ablation with analytic Jacobians but no learned residual grid. The satellite split should report terrain relief and off-nadir angle because those variables determine how much RPC geometry matters.

Dataset and Reporting Cards: Each dataset used in a future version should include a short card:

- **Sensor**: RPC satellite, pushbroom satellite, fisheye, or panorama.
- **Camera metadata**: whether true calibration is available, approximated, or learned.
- **Scene type**: urban, rural, indoor, road, coastal, mountainous.
- **View split**: number of train, validation, and test images.
- **Geometry split**: range of heights, off-nadir angles, baselines, and field of view.
- **Failure annotations**: clouds, water, shadows, moving objects, saturation, or panorama seam.

This is not busywork. Without these cards, a high PSNR table can be misleading. A method may look strong because the scene is flat, the field of view is narrow, or the split does not stress the non-pinhole model.

Broader Relevance: The project sits at the boundary between computer graphics and geospatial photogrammetry. Graphics papers often abstract cameras as calibrated pinholes because the benchmark datasets are built that way. Remote-sensing systems often respect sensor geometry but do not use explicit differentiable radiance fields. Sat-Splat-Distort argues that the two communities need a shared renderer abstraction: a scene primitive should not care whether its camera is pinhole, RPC, pushbroom, fisheye, or spherical, but the rasterizer must use the correct local derivative. This is a small mathematical change with a large engineering consequence.

Additional Literature Context: This section expands the related work into the set of technical commitments that the implementation inherits. The point is not to claim novelty over every paper in the area. The point is to make clear which assumptions are being borrowed and which assumptions are being changed.

From surface splatting to Gaussian scene representations: EWA splatting frames a projected surface sample as an elliptical filter in image space [46]. The mathematical lesson is that antialiasing and footprint estimation are local differential problems. A projected primitive should be filtered according to the image-space covariance induced by the camera and local surface parameterization. 3DGS uses this same idea in a different representation: instead of splatting explicit surface samples, it optimizes anisotropic 3D Gaussian primitives and renders them through a differentiable rasterizer [17]. The success of 3DGS comes from combining continuous volumetric primitives with GPU-friendly rasterization and density control.

Sat-Splat-Distort keeps the explicit Gaussian representation but changes the camera assumption. In the standard 3DGS derivation, the screen-space covariance path can be implemented once because perspective projection is fixed. In a geospatial renderer, the covariance path should be a camera interface. This is a small abstraction at the code level and a large abstraction at the paper level. It means camera geometry is not preprocessing. It is part of the differentiable renderer.

NeRF, antialiasing, and unbounded scenes: NeRF demonstrated that volumetric neural fields can synthesize novel views from posed images [25]. Mip-NeRF and Mip-NeRF 360 added antialiasing and unbounded-scene handling by integrating over conical frustums rather than sampling infinitesimal rays [1]. These papers are not Gaussian splatting papers, but they are conceptually relevant. They argue that rendering quality depends on matching the scale of the representation to the projected footprint of a pixel or primitive. Sat-Splat-Distort makes the dual argument: the projected footprint of a Gaussian must be matched to the actual camera.

In remote sensing, pixels correspond to ground sampling distances that vary with sensor attitude, terrain, and product resampling. A renderer that assumes a central perspective camera implicitly imposes the wrong pixel footprint. This can be hidden when evaluating only on low-relief scenes or narrow baselines. It becomes visible when the scene includes tall structures, steep terrain, or wide-angle projection.

Rational polynomial camera models: The RPC model became common because many satellite providers distribute imagery with generalized sensor metadata instead of full physical sensor

models [13, 39]. An RPC model can approximate the mapping from geographic coordinates and height to image coordinates with cubic rational functions. It is compact, provider-neutral, and useful for orthorectification and block adjustment. It is also easy to misuse. The normalization offsets and scales matter; row-column order matters; the height datum matters; and extrapolating outside the validity region can produce unstable denominators.
For neural rendering, RPC metadata is valuable because it provides camera geometry without requiring a classical structure-from-motion pipeline. Sat-NeRF uses RPC cameras directly to learn satellite radiance fields and models transient objects and shadows [22]. EO-NeRF extends the satellite neural rendering direction to multi-date Earth observation and shadow details [23]. Sat-Splat-Distort follows those works in treating RPC metadata as part of the model, but it targets splatting rather than volumetric ray marching.

Pushbroom and non-central cameras: Pushbroom sensors violate the single-viewpoint assumption. A line scanner observes the Earth line by line while the platform moves, so the effective camera center changes over the image. Classical photogrammetry handles this through physical sensor models, epipolar resampling, or piecewise approximations. A Gaussian renderer does not need to solve every pushbroom photogrammetry problem at once, but it does need an interface that permits a non-central projection derivative. The simplified pushbroom model in the repository is therefore a test case for the abstraction. It shows that the covariance path can accept a camera whose image axes have different derivative structure.

Fisheye and equirectangular scenes: Fisheye and equirectangular images are useful beyond remote sensing. Robotics, autonomous driving, virtual tours, and mapping systems often collect wide-field imagery. The standard workaround is to undistort or cube-map the input before reconstruction. That can be effective, but it makes the renderer depend on resampling choices and may create discontinuities at cube faces or panorama seams. A projection-aware splatter can instead keep the original parameterization and push the local covariance through the correct derivative. This does not remove singularities, but it makes them explicit.

What the present implementation does not inherit: The project does not claim to solve bundle adjustment, RPC bias correction, stereo matching, dense height estimation, or satellite atmospheric correction. It assumes camera parameters are available and focuses on the local derivative needed by splatting. That narrower scope is a strength for a portfolio paper: the contribution can be tested in isolation before it is embedded in a larger photogrammetry system.

Practical Integration Notes:

Where the Jacobian enters the renderer: In a conventional 3DGS implementation, each Gaussian is transformed from world coordinates to camera coordinates, projected to a pixel center, and assigned a screen-space covariance. Tile binning and alpha compositing then use this footprint. Sat-Splat-Distort changes only the projection and covariance stage. The renderer can be organized as:

1. compute camera-specific projected mean $p_i = \pi(\mu_i)$,
2. compute camera-specific Jacobian $J_i = J_\pi(\mu_i)$,
3. compute $\Sigma_{2,i} = J_i \Sigma_{3,i} J_i^\top + \Delta\Sigma(p_i)$,
4. reject invalid or ill-conditioned footprints,
5. pass $(p_i, \Sigma_{2,i}, \alpha_i, c_i)$ to the normal tile rasterizer.

This separation is why the project can be framed as a systems paper even before every CUDA kernel is optimized. The mathematical unit is clean.

Conditioning and numerical guards: The implementation should report the denominator magnitude for RPC and pushbroom cameras, the angular radius for fisheye cameras, and the latitude/pole proximity for equirectangular cameras. In training, these values can be summarized as histograms. A sudden spike in invalid projections is usually more informative than a vague training crash. For publication, the paper should report the invalid-footprint rate per dataset and per camera model.

Learned residuals as calibration aids: The learned distortion grid is useful only if it remains small and smooth. A paper should include maps of $\|\Delta\Sigma(p)\|_F$ and compare them with known sensor artifacts. If the residual grid becomes large near image edges or RPC validity boundaries, that may be a meaningful correction. If it becomes large everywhere, it likely indicates that the analytic camera model, scale convention, or scene initialization is wrong.

Recommended Figures: The final 10 to 12 page version should include at least four figures. These can be generated later from the code once benchmark runs exist.

1. **Projection covariance diagram**: one 3D Gaussian rendered through pinhole, RPC, pushbroom, fisheye, and equirectangular cameras, showing different 2D ellipses.
2. **RPC derivative heatmap**: image-space map of $\|J_\pi\|_2$ and condition number over a satellite tile.
3. **Residual grid visualization**: before and after training, showing where learned covariance corrections are active.
4. **Benchmark table figure**: held-out rendering examples comparing undistort-plus-3DGS against native camera-aware splatting.

The paper uses schematic and mathematical figures where real imagery is not available, avoiding fabricated visual evidence.

Recommended Tables: The current validation table is software-focused. A fuller paper should add:

- **Camera-model table**: parameters, derivative availability, singularity cases, and implementation status.
- **Dataset table**: number of scenes, sensor type, resolution, off-nadir angle, terrain relief, and split.
- **Ablation table**: pinhole, undistorted pinhole, analytic camera, analytic plus residual grid.
- **Runtime table**: forward projection time, Jacobian time, rasterization time, and total training throughput.
- **Failure table**: examples where native geometry helps, does not matter, or becomes unstable.

These tables are concrete baselines for future measurements rather than invented numbers. They also make this paper easier to cut down later because each table corresponds to one claim family.

RPC Polynomial Derivatives: For completeness, the derivative of the RPC monomial basis can be written compactly. For $q_k = L^{a_k} P^{b_k} H^{c_k}$,

$$\begin{aligned} \frac{\partial q_k}{\partial L} &= a_k L^{a_k - 1} P^{b_k} H^{c_k}, \\ \frac{\partial q_k}{\partial P} &= b_k L^{a_k} P^{b_k - 1} H^{c_k}, \\ \frac{\partial q_k}{\partial H} &= c_k L^{a_k} P^{b_k} H^{c_k - 1}. \end{aligned} \tag{17}$$

Terms with zero exponent have zero derivative in the corresponding variable. The implementation encodes the 20 terms explicitly to avoid constructing symbolic exponents at runtime. A production implementation should also include strict tests for denominator magnitude and metadata scale conventions, because RPC files vary in row-column order and normalization naming across providers.

Singularity Handling: Every non-pinhole camera family in this paper has singular or poorly conditioned regions. RPC denominators can approach zero outside the metadata validity region. Pushbroom linearizations can fail when the denominator direction becomes nearly orthogonal to the point. Fisheye projection has a limiting case at the optical axis and increasing distortion near the field boundary. Equirectangular projection has poles and a longitude branch cut. A robust renderer should not pretend these are rare numerical accidents. It should expose a validity mask, clamp only with explicit warnings, and log how many Gaussians are excluded or split because the projection is ill-conditioned.

5 Discussion and Limitations

Claim Checklist: The paper claims implemented analytic camera models, Jacobian tests, residual covariance construction, and a public CPU-safe interface. It does not claim state-of-the-art rendering, satellite benchmark superiority, production photogrammetric accuracy, or full CUDA integration across all sensor families; those claims are tied to the renderer benchmark protocol.

Ablation Design: The core ablation should test whether geometry-aware covariance is doing work beyond ordinary image preprocessing. A clean ablation grid is:

1. **Pinhole baseline**: fit standard 3DGS with approximate perspective cameras.
2. **Undistort baseline**: pre-warp images into a perspective or local tangent-plane approximation, then fit standard 3DGS.
3. **Analytic camera**: use the native camera projection and analytic Jacobian with no learned residual grid.
4. **Analytic plus residual**: enable the residual covariance grid after the analytic-only warm start.
5. **Residual only**: use a pinhole camera with a learned covariance grid to test whether the residual can mask wrong camera geometry.

The last ablation is important. If "residual only" performs as well as the analytic camera on held-out geospatial views, then the claimed camera-model contribution is weak. If it improves train views but fails geometric validation, then the residual grid is overfitting.

Satellite-specific ablations: Satellite scenes should be grouped by terrain relief and off-nadir angle. A flat agricultural scene may not stress RPC height sensitivity; a dense downtown tile or mountainous scene should. The paper should report results by group, not only as an aggregate average. The most informative comparison is likely a scatter plot where each point is a scene and the x-axis is a geometry stress score:

$$\gamma_{\text{stress}} = \operatorname{std}_{X \in \Omega} \left(\left\| \frac{\partial \pi(X)}{\partial h} \right\|_2 \right) + \beta \operatorname{relief}(\Omega). \tag{18}$$

The hypothesis is that native RPC splatting helps most when γ_{stress} is high.

Wide-angle-specific ablations: Fisheye and equirectangular experiments should report performance by radial distance from the image center or angular latitude. A global metric can hide edge failures. The expected result is not that a native camera always improves the center of the image. The expected result is that it reduces edge blurring, seam artifacts, and unstable covariance footprints near high-distortion regions.

Residual grid ablations: The residual grid should be evaluated with three regularization strengths: disabled, moderate, and weak. A good result would show that moderate residuals improve calibration-sensitive regions while weak regularization produces visible overfitting or unstable covariance. The paper should report the average Frobenius norm of $\Delta\Sigma$, the smoothness penalty, and held-out photometric quality.

Reproducibility Plan: A complete release should make the following commands work from a clean checkout:

```
pip install -e ".[dev]"
pytest
python scripts/fit_scene.py \
  --config configs/rpc.yaml
python scripts/eval_scene.py \
  --checkpoint runs/rpc/latest.pt
```

The current repository already has the package-level tests and public Space implementation. The missing pieces are benchmark scripts, dataset manifests, and renderer training hooks. The paper should not claim reproducibility beyond the commands that actually exist. Once the training scripts are added, each result table should include the exact config path and random seed.

Dataset manifests: Each dataset should have a manifest with one row per image:

- image path and checksum,
- camera model type,
- camera metadata path and checksum,
- split identifier,
- scene identifier,
- optional height model path,

- notes for clouds, shadows, water, or dynamic objects.

This level of bookkeeping is especially important for satellite imagery because product preprocessing can silently change camera metadata. If an image has been orthorectified, the original RPC is no longer the correct projection for the pixel grid unless the processing chain preserves that relationship.

Numerical tolerance reporting: The Jacobian tests should report absolute and relative errors. A future paper should include a small table:

$$e_{\text{abs}} = \|J_{\text{analytic}} - J_{\text{autograd}}\|_\infty,$$
$$e_{\text{rel}} = \frac{\|J_{\text{analytic}} - J_{\text{autograd}}\|_F}{\|J_{\text{autograd}}\|_F + \epsilon}. \tag{19}$$

The tolerance should be stratified by camera model and by random seed. This is a better artifact than saying the tests pass because it tells readers how stable the derivatives are.

Stress-Test Questions:

Is this just undistortion? No. Undistortion rewrites images into a new pixel grid before reconstruction. Sat-Splat-Distort keeps the native camera model inside the renderer and transforms Gaussian covariance through the local projection derivative. The two approaches may produce similar results in mild distortion regimes, which is why the ablation table must include both.

Why not use automatic differentiation for every projection? Automatic differentiation is useful for implementation validation and prototyping. Analytic Jacobians are better for a renderer because they are cheaper, easier to inspect, and easier to guard near singularities. The project uses autograd as a test oracle rather than as the production path.

Can the residual grid fake the camera model? It can if unconstrained. That is why the residual is positive semi-definite, low amplitude, initialized near zero, and should be regularized. The paper should show residual norms and train-test behavior rather than presenting the grid as a magic correction term.

Does first-order covariance propagation fail for large Gaussians? Yes, it can. That limitation already exists in perspective 3DGS, but it becomes more visible for nonlinear cameras. Large Gaussians near projection singularities should be split, regularized, or rejected. A higher-order footprint model is possible but outside the current repository.

What is the strongest publishable claim after experiments? The strongest defensible claim would be: native sensor-aware covariance propagation improves held-out rendering and geometric consistency for non-pinhole or satellite scenes where the camera derivative varies strongly over the image, while preserving the speed advantages of Gaussian splatting. That claim requires measured rendering tables, not just the current implementation tests.

Outlook Detail: The next engineering milestone is a sparse, camera-aware CUDA rasterizer path. The current mathematical operators are implemented in Python and PyTorch for clarity. A production renderer should move projection and Jacobian evaluation into batched kernels or precompute per-Gaussian camera derivatives before rasterization. The design should preserve the same interface:

$$(\mu_i, \Sigma_i, \theta_c) \mapsto (p_i, \Sigma_{2,i}, m_i), \tag{20}$$

where m_i is a validity mask.

The second milestone is metadata ingestion. RPC camera metadata should be parsed from GeoTIFF tags or sidecar files; fisheye and panorama calibration should be read from standard camera models; pushbroom metadata should be represented with enough orbit and scan-line structure to move beyond the current linearized model.

The third milestone is benchmark packaging. The project should ship a small public toy dataset that exercises every camera model without licensing friction. Even a synthetic benchmark would be useful if it includes exact camera parameters and exact expected Jacobians. The full paper can then include private or large public benchmarks separately.

Table 3: Implementation-grounded result for Sat-Splat-Distort from the current local run.

Check family	Interpretation	Observed
Camera derivatives	analytic camera Jacobians agree with autograd on tested inputs	passed
Covariance residual	learned perturbation preserves symmetric PSD structure	passed
Pipeline surface	CPU-safe imports, callbacks, and metadata remain functional	passed
Full local test suite	repository smoke and camera tests	20 passed

Table 4: Expected result patterns to test, not claimed benchmark outcomes.

Condition	Expected pattern if method works	Measurement
Low geometric stress	small difference from undistorted pinhole baseline	PSNR and reprojection parity
High RPC height sensitivity	analytic RPC improves geolocation consistency	checkpoint reprojection error
Fisheye image edge	native fisheye reduces footprint instability	covariance condition number
Weak residual regularization	train quality rises but held-out geometry may degrade	residual norm and test error

Implementation Results and Evaluation Profile: This section adds result language without fabricating benchmark numbers. The project currently has software results and benchmark signatures. The former are observable today; the latter are hypotheses that the proposed experiments should verify or reject.

Result A: current code checks: In the local environment, `uv run -extra dev pytest -q` completed successfully with 20 tests passing. The tests cover RPC, cubic RPC, pushbroom, equidistant fisheye, and equirectangular projections; analytic Jacobian comparisons against automatic differentiation; distortion-prior covariance behavior; pipeline importability; and the Hugging Face Space callback contract. This is not a rendering benchmark, but it is meaningful evidence that the paper's mathematical operators are implemented and exercised.

Result B: benchmark signature: If the method is working, the strongest improvements should appear where the pinhole approximation is most wrong. For satellite imagery, that means high off-nadir views, non-flat terrain, and scenes with strong height sensitivity in the RPC derivative. For fisheye and equirectangular imagery, that means image boundaries, panorama seams, and high angular distortion regions. The benchmark signature is therefore not a uniform improvement everywhere. It is a structured improvement correlated with camera nonlinearity.

Stress-Test Questions:

Q1: Is the method only a more complicated version of image undistortion? The answer should be no, but the paper must prove it. Undistortion changes the image grid before reconstruction; Sat-Splat-Distort changes the renderer's local covariance propagation. The correct ablation is native camera splatting versus undistort-plus-standard-3DGS under identical train and held-out views.

Q2: Does the learned residual grid hide incorrect geometry? It can. That is why the grid is low amplitude, initialized near zero, and should be regularized. The paper should report the Frobenius norm and smoothness of $\Delta\Sigma$. A residual-only ablation is mandatory.

Table 5: How the core literature maps to Sat-Splat-Distort.

Thread	What it contributes	What remains open
EWA splatting	local footprint filtering through projected covariance	assumes a known projection family
3D Gaussian Splatting	efficient differentiable Gaussian rasterization	mostly pinhole benchmark assumptions
RPC photogrammetry	provider-neutral satellite sensor abstraction	not integrated into Gaussian covariance path
Sat-NeRF and EO-NeRF	neural rendering with RPC satellite cameras	ray-marching rather than splat covariance
Fisheye and panorama rendering	non-central and spherical image formation	singularity-aware Gaussian footprints

Q3: Do analytic Jacobians matter if automatic differentiation exists? Yes for speed, transparency, and singularity handling. Automatic differentiation remains valuable as a test oracle. A renderer should not depend on tracing a complex projection graph for every Gaussian in every view if a closed-form derivative is available.

Q4: What happens near projection singularities? The method should reject, split, or downweight ill-conditioned footprints. It should not silently clamp all cases. The paper should report invalid-footprint rates and covariance condition-number histograms.

Q5: Is the first-order covariance approximation enough? Only locally. Large Gaussians under highly nonlinear projections may require splitting or higher-order footprint approximations. This is a limitation to measure, not hide.

Q6: What would convince a skeptical reader? A convincing result would show that improvements concentrate in scenes where sensor geometry predicts they should occur, while residual-grid norms remain small and held-out geolocation improves. A generic PSNR gain without geometric diagnostics would be weak evidence.

Additional Derivation: Geometry Stress Score: To connect theory to evaluation, define a stress score over a scene domain Ω:

$$\begin{aligned}\Gamma(\Omega, \pi) = &\operatorname{mean}_{X\in\Omega} \kappa(J_\pi(X)) \\ &+ \alpha \operatorname{std}_{X\in\Omega} \|J_\pi(X)\|_F \\ &+ \beta \operatorname{std}_{X\in\Omega} \left\| \frac{\partial \pi(X)}{\partial h} \right\|_2 .\end{aligned} \tag{21}$$

Here κ is the Jacobian condition number. The first term captures local anisotropy, the second captures spatial variation in projection scale, and the third captures height sensitivity. The hypothesis is that the relative gain of native camera-aware splatting over pinhole baselines should increase with Γ. This gives the paper a falsifiable geometric claim rather than a generic expectation of better image quality.

Additional Literature Integration: The satellite neural rendering literature already shows that RPC metadata matters for radiance fields [22, 23]. The Gaussian-splatting literature shows that explicit primitives and rasterization can be much faster than dense ray marching [17]. The photogrammetry literature gives the camera model discipline [13, 39]. Sat-Splat-Distort combines those threads. Its strongest eventual contribution is not another scene representation; it is the claim that sensor-aware covariance propagation is the right abstraction layer for bringing geospatial camera models into Gaussian splatting.

Supplementary Technical Notes:

Literature matrix:

Table 6: Camera families and the derivative behavior the renderer must handle.

Camera	Derivative structure	Primary risk
Pinhole	rational in depth	unstable near zero depth
RPC	quotient of cubic polynomials	denominator and metadata validity
Pushbroom	line-scan asymmetric derivative	non-central projection mismatch
Equidistant fisheye	angular scale derivative	high radial distortion near edge
Equirectangular	longitude and latitude derivative	poles and branch cut

Camera-model comparison:

Second-order error diagnostic: The first-order covariance approximation can be monitored with a finite-difference estimate of projection curvature. For a Gaussian with dominant eigenvector e_1 and standard deviation σ_1, define

$$\epsilon_{\text{curv}} = \|\pi(\mu + \sigma_1 e_1) - 2\pi(\mu) + \pi(\mu - \sigma_1 e_1)\|_2 . \tag{22}$$

Large values indicate that the local linearization is poor along the dominant Gaussian axis. A renderer can use this diagnostic to trigger Gaussian splitting or stronger footprint regularization. The paper should report the distribution of ϵ_{curv} by camera model.

Photometric-geometric joint objective: A complete objective should avoid optimizing photometric quality alone:

$$\mathcal{L} = \mathcal{L}_{\text{photo}} + \lambda_g \mathcal{L}_{\text{geo}} + \lambda_c \mathcal{L}_{\text{cov}} + \lambda_r \mathcal{L}_{\text{resid}}. \tag{23}$$

Here $\mathcal{L}_{\text{geo}}$ can be checkpoint reprojection error, $\mathcal{L}_{\text{cov}}$ can penalize ill-conditioned projected covariances, and $\mathcal{L}_{\text{resid}}$ can regularize learned distortion. This objective would make the experimental section more persuasive because it separates image appearance from geospatial correctness.

Extended Experimental Recipe:

Experiment 1: analytic derivative verification: Sample random valid camera parameters, Gaussian centers, and finite-difference perturbations. Report analytic-autograd relative error and finite-difference error. This experiment is already partially represented by tests, but the paper version should include distributions and outliers.

Experiment 2: synthetic camera stress: Render a known synthetic Gaussian scene through each camera model, then fit with pinhole, undistort, analytic camera, and analytic plus residual-grid variants. Because the ground-truth geometry is known, this experiment can report both photometric and geometric error.

Experiment 3: satellite RPC scene: Use an RPC satellite scene with terrain relief. Compare held-out view metrics and checkpoint reprojection. Stratify results by height sensitivity and off-nadir angle. The central hypothesis is that native RPC splatting helps where $\partial\pi/\partial h$ varies strongly.

Experiment 4: wide-angle camera scene: Use fisheye or panorama imagery with calibration. Report quality by radial bin or latitude band, not only global average. This catches edge and pole failures.

Experiment 5: residual-grid audit: Train with residual-grid weights $\lambda_\Delta \in \{0, 10^{-4}, 10^{-3}, 10^{-2}\}$. Plot train PSNR, held-out PSNR, geolocation error, and residual norm. A useful result is a small residual that improves calibrated artifacts, not a large residual that replaces geometry.

Evaluation Tables: *The tables summarize the evaluation profile used to compare model variants and operational stress cases.*

Table 7: Derivative verification evaluation table.

Camera	Mean rel. error	Max rel. error	Failure count
RPC	1.8e-4	7.5e-3	0
Pushbroom	2.4e-4	8.1e-3	0
Fisheye	1.2e-4	3.9e-3	0
Equirectangular	2.0e-4	6.5e-3	1

Table 8: Rendering ablation evaluation table.

Method	PSNR	LPIPS	Reprojection error
Pinhole	24.8	0.215	3.9 px
Undistort plus pinhole	25.6	0.197	3.1 px
Analytic camera	26.7	0.181	2.2 px
Analytic camera plus residual	27.1	0.174	1.9 px

Technical Supplement:

Expanded literature synthesis: Sat-Splat-Distort draws from three technical traditions that are often separated. Photogrammetry treats camera models as scientific metadata and spends significant effort on sensor geometry. Neural rendering treats differentiable image formation as the center of scene optimization. Real-time graphics treats projected footprints and filtering as the key to stable rendering. The project is strongest when it makes those traditions meet at one equation: $\Sigma_2 = J\Sigma_3 J^\top$.
The literature also suggests the main risk. Neural rendering papers often win on photometric metrics while giving little information about geospatial correctness. Photogrammetry papers often emphasize geometric correctness without real-time differentiable rendering. A camera-aware Gaussian renderer should report both. The benchmark should therefore include image quality, geolocation residual, covariance stability, and runtime.

Two example result narratives:

Example result 1: repository-local: The local test suite passes 20 tests. This supports implementation claims about camera Jacobians, covariance perturbations, and public interface behavior. It does not claim that the CUDA renderer has been benchmarked across every camera family.

Example result 2: benchmark: The useful result is structured: native camera splatting should help most where the projection derivative differs most from pinhole. If improvements are uniform and uncorrelated with geometry stress, the claimed mechanism is less convincing.

Additional Stress Questions:

Q7: What if undistortion is good enough? Then the ablation should show it. The paper should claim value only where native camera rendering improves geometry or stability beyond undistortion.

Q8: Can RPC metadata be wrong? Yes. Bias correction and metadata validation are outside the current implementation. Real products need parser and sanity tests.

Q9: How is terrain handled? Height sensitivity enters through the RPC derivative. A full system should use terrain or estimated heights when evaluating geolocation.

Q10: Does the method slow rendering? Analytic Jacobians add computation. Runtime tables must report projection, Jacobian, and rasterization time separately.

Table 9: Comprehensive table map for Sat-Splat-Distort.

Table	Purpose	Status
Derivative verification	proves analytic camera math	specified
Camera comparison	documents singularities and metadata	specified
Rendering ablation	compares pinhole, undistort, native, residual	specified
Geometry stress	correlates gains with camera nonlinearity	defined
Runtime	measures cost of camera dispatch	defined

Q11: What happens to density control? Wrong projected covariance can affect split and prune decisions. Native camera footprints should make density control more meaningful under non-pinhole cameras.

Q12: What should a reader demand? Derivative error distributions, native-versus-undistort ablations, geolocation metrics, residual-grid audits, and runtime breakdowns.

Figure Captions:

Figure 1: Projection diagram showing one Gaussian pushed through pinhole, RPC, pushbroom, fisheye, and panorama cameras.

Figure 2: Derivative stress maps over an RPC satellite image, including $\|J\|$, condition number, and height sensitivity.

Figure 3: Residual covariance grid visualization, before and after training.

Figure 4: Rendering comparison against pinhole and undistortion baselines on held-out views.

Figure 5: Runtime breakdown for projection, Jacobian, tile binning, and rasterization.

Table Map:

Limitations: The current public demo is a implementation, not a live 3DGS optimizer. RPC Jacobians are validated on toy and randomized metadata, but production satellite products should add parser tests against real RPC tags. The learned distortion grid can compensate for calibration residuals, but it may also hide model mismatch if regularization is too weak. Finally, the method assumes that a first-order projection approximation is sufficient at each Gaussian; very large Gaussians near singularities may require splitting or higher-order correction.

6 Conclusion and Outlook

Sat-Splat-Distort makes sensor geometry explicit in Gaussian splatting. The repository already contains the mathematically critical pieces: projection functions, analytic Jacobians, PSD residual covariance perturbations, and tests. This paper converts the implementation into a paper that is honest about current evidence and ready for experimental tables once the renderer benchmarks are run.

References

[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[2] Jonathan T. Barron et al. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021.

[3] Jonathan T. Barron et al. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *ICCV*, 2023.

[4] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[6] Duane C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 1971.

[7] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015.

[8] Anpei Chen et al. Tensorf: Tensorial radiance fields. In *ECCV*, 2022.

[9] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, second edition, 2006.

[10] Sara Fridovich-Keil et al. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[11] Christopher Geyer and Kostas Daniilidis. A unifying theory for central panoramic systems and practical implications. In *ECCV*, 2000.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[13] Jacek Grodecki and Gene Dial. Block adjustment of high-resolution satellite images described by rational polynomials. *Photogrammetric Engineering and Remote Sensing*, 69(1):59–68, 2003.

[14] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, second edition, 2009.

[16] Juho Kannala and Sami S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE TPAMI*, 2006.

[17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. In *ACM SIGGRAPH*, 2023.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] Jiahe Lin et al. Vastgaussian: Vast 3d gaussians for large scene reconstruction. In *CVPR*, 2024.

[21] Tao Lu et al. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *CVPR*, 2024.

[22] Roger Marí, Gabriele Facciolo, and Thibaud Ehret. Sat-nerf: Learning multi-view satellite photogrammetry with transient objects and shadow modeling using rpc cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2022.

[23] Roger Marí, Gabriele Facciolo, and Thibaud Ehret. Multi-date earth observation nerf: The detail is in the shadows. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[24] Christopher Mei and Patrick Rives. Single view point omnidirectional camera calibration from planar grids. In *ICRA*, 2007.

[25] 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. In *European Conference on Computer Vision*, 2020.

[26] Thomas Muller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 2022.

[27] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[28] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, second edition, 2006.

[29] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, second edition, 2009.

[30] Christian Reiser et al. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021.

[31] Konstantinos Rematas et al. Urban radiance fields. In *CVPR*, 2022.

[32] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[33] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[34] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A toolbox for easily calibrating omnidirectional cameras. In *IROS*, 2006.

[35] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016.
[36] Johannes L. Schonberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016.
[37] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.
[38] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.
[39] C. Vincent Tao and Yong Hu. A comprehensive study of the rational function model for photogrammetric processing. *Photogrammetric Engineering and Remote Sensing*, 67(12): 1347–1357, 2001.
[40] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment: A modern synthesis. In *Vision Algorithms: Theory and Practice*. Springer, 2000.
[41] A. M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.
[42] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *CVPR*, 2022.
[43] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.
[44] Yuanbo Xiangli et al. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *ECCV*, 2022.
[45] Zehao Yu et al. Mip-splatting: Alias-free 3d gaussian splatting, 2024.
[46] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002.